University of Economics, Prague

Faculty of Informatics and Statistics

Department of Information and Knowledge Engineering

Habilitation Thesis

Interpretable Data Analysis with Entity-based Text Representations and Rule-based Models

Tomáš Kliegr

September 2018

Acknowledgments

I would like to thank my current and former colleagues and collaborators at the Department of Information and Knowledge Engineering who either directly contributed to the work presented in the thesis or helped to inspire or guide the work. For comments that helped to shape the introduction of the thesis, I would like to thank prof. Jan Rauch and prof. Vojtěch Svátek. I would especially like to thank my entire family.

Contents

List of Acronyms

ANN Artificial Neural Network

ASR Automatic Speech Recognition

BOE Bag of Entities

BOW Bag of Words

BRMS Business Rule Management Systems

CBA Classification By Associations (classification algorithm)

EDL Entity Discovery and Linking

ESA Explicit Semantic Analysis

GDPR General Data Protection Regulation

GUHA General Unary Hypotheses Automaton

iML interactive Machine Learning

JAPE Java Annotation Patterns Engine

LHD Linked Hypernyms Dataset

MFS Most Frequent Sense

NIST National Institute for Standardization (U.S.)

PMML Predictive Model Markup Language

SVM Support Vector Machine

WEKA Waikato Environment for Knowledge Analysis

WIN Word INterchangeability dataset

WSC Word Similarity Computation

Part I.

Introduction

1. Preface

The research outlined in this thesis covers two different aspects of interpretable data analytics: entity-based representation, which allows to describe text in an unambiguous and machine readable way, and association rules, which present patterns appearing in data in a way naturally comprehensible to humans. These areas fall within the scope of the topics pursued at the Department of Information and Knowledge Engineering (DIKE), where I have been working as an assistant professor.

Following in the research focus of my dissertation thesis defended at DIKE in 2012 [59], I continued to work in the area of entity-based text representation. This field combines natural language processing with semantic web technologies, to provide a machine understandable layer over text documents. The main research problem I addressed was entity classification – identification of entities (people, places, things, etc.) in the text. The results can be used to enrich and improve accuracy of knowledge graphs, which are an important source of information for many general artificial intelligence applications. In my work, I focused on one of the largest open knowledge graphs, DBpedia [64], the Czech version of which is hosted at DIKE.

Historically, an important research track in DIKE was machine learning, specifically learning of rules and various rule-like patterns from data. The origins date back to 1960's, when the the GUHA method (General Unary Hypotheses Automaton) was conceived to "describe all the possible assertions which might be hypotheses" [39]. The GUHA method thus predates the discovery of the backpropagation algorithm for training artificial neural networks (ANNs) [100, 101], which made training of multi-layer networks feasible and efficient. While currently ANNs overtake many symbolic machine learning frameworks thanks to their versatility and high accuracy on a range of problems, the rule-like representations remain an important subject of research, largely owing to their good comprehensibility. In this thesis, I report on several papers that relate to comprehensibility of rule models, studying various approaches that can reduce the number of rules in rule models.

An important element in research at UEP is the emphasis on applications, which can be readily used by IT practitioners as well as entrepreneurs. To respond to this, I initiated the development of an easy to use web-based framework for rule learning. The research started around 2010, when a predecessor of what is now called EasyMiner succeeded in the international RuleML Challenge, a contest organized within the RuleML conference to promote new applications of rule systems. The first release of EasyMiner incorporating on-line mining capabilities, presented at the ECML/PKDD conference in 2012, used as a mining backend the LISp-Miner system [90], developed since 1996 at DIKE.[1] EasyMiner was used in teaching at UEP since 2013, which helped to expose hundreds of students to Machine Learning as a Service (MLaaS) principles already at the time when only few MLaaS systems were available.

With the Interest Beat (InBeat) project we combined results from both research tracks with a recommender system, which creates user models composed of rules learnt from entity-based

[1] https://lispminer.vse.cz/people.html

description of the content the user has interacted with. While InBeat was initially conceived as an application providing the end-users with useful suggestions, in our future work we plan to use it to improve understanding of psychological factors that affect interpretability of rules.

Thesis organization. This thesis is divided into three parts. Part I (Introduction) walks the reader through the individual contributions following this structure: a short motivation, definition of challenges, followed by an account of how these were addressed. The introduction is split into two chapters by the target area – Chapter 2 covers entity-based representations and Chapter 3 rule learning. Chapter 4 gives an overview of papers presenting the contributions, stating also the author's approximate share on each of the seven most significant conference papers and journal articles, the reprints of which were selected for inclusion in Part II and Part III. Chapter 5 presents a summary of contribution and an outlook for future work. Part II contains the reprints of four selected papers from the area of entity-based description of text, and Part III the reprints of the three selected papers from the rule learning domain.

2. Entity-based text representation

2.1. Motivation and State-of-the-art

The term “entity” is defined by the U.S. National Institute for Standardization (NIST) within the 2017 Entity Discovery and Linking (EDL) task as a specific individual person, organization, geopolitical entity, location, or facility.[1] Entities can be used for a *knowledge-based representation of text*, as opposed to the previously adopted approach describing text using bag-of-words [104] or word vectors. The advantage of the entity-based text representation is incorporation of high quality domain knowledge available in knowledge bases, the use of structured representation and deeper understanding of text [104]. Representing text with entities corresponds to number of research tasks, which are demonstrated in the following example.

Example. Assume that a user is watching a sport broadcast, and in the speech transcript the commentator says, “Maradona scored goal of the century”. As the first step, *mention detection* is performed – the word “Maradona” is identified as a candidate entity. As the second step, the entity is *disambiguated* to a knowledge base resource such as `dbpedia.org/page/Diego_Maradona`. These two operations are often performed within one *entity linking* system.
For the knowledge base to contain the information on the entity, it need to have been previously *populated*, typically using algorithms analysing semistructured documents describing the entities, such as a Wikipedia article on Diego Maradona.
Let us assume that the knowledge base contains for `dbpedia.org/page/Diego_Maradona` one entity type (“footballer”), but our target application aims to categorize entities either as “midfielder” or “goalkeeper”. A number of *entity classification* algorithms can be used for such purpose. Some of these can use machine learning techniques to create supervised classifiers from labelled data. Another approach is based on application of *semantic word similarity* algorithms, which analyse the relative position of the words in some thesaurus, such as WordNet [17].
If the target application requires information on how important the entities are in given context, *entity salience* algorithms can be used, in this case maybe assigning “Maradona” and “goal” the highest salience level, and “century” a lower salience level.

The importance of entity linking for competitive intelligence and corporate decision making is widely recognized [10]. For example, one of the first commercially available entity linking systems, Open Calais[2], is operated by Thomson Reuters, which is an important provider of information to enterprises worldwide. To this end, for design or testing models in our research, we also aimed to use data based on Reuters content.[3] We used Reuters-128 in [14],

[1] `https://tac.nist.gov/2017/KBP/`
[2] `http://www.opencalais.com/`
[3] `https://trec.nist.gov/data/reuters/reuters.html`

and Reuters-21578 news corpora in [61]. However, the examples used in this thesis are mostly constructed for the football domain, which is traditionally used to illustrate entity linking research (cf. recently e.g. [95, 6]).

2.1.1. Entity Linking

Entity linking largely evolved from Named Entity Recognition (NER). In the NER task, the goal typically is to identify and classify CONLL-based named entity types [93]: persons, locations, organizations and names of miscellaneous entities that do not belong to the previous three groups. In entity linking, the goal is to link the entity to the correct entry in the knowledge base. Once this is performed, the knowledge base already provides a machine-readable entity type. An authoritative current definition of the entity linking task is provided in [89].

The first step in entity linking is called mention detection. The purpose of mention detection is to identify spans in the input text that correspond to an entity. Applicable methods can be classified into two high level categories: rule-based methods and statistical named-entity recognition, these two approaches can also be combined [26]. An overview of algorithms for mention detection is provided, for example, by summary reports from the NIST TAC entity discovery and linking tasks [44]. Once mentions have been identified the next step is their linking to the knowledge base. This process is also sometimes called “wikification” [70], because multiple commonly used knowledge bases are derived from Wikipedia.

A benchmark of selected approaches focused specifically on disambiguation to Wikipedia has been published in [36]. For entity linking, new algorithms were proposed, but also approaches previously developed for word sense disambiguation can be reused [72]. Always assigning the most frequent sense of the given word has been widely adopted as a base line in word sense disambiguation research [74]. The challenge addressed in our work was establishing the equivalent of the “most frequent sense” baseline for linking entities to a knowledge base. **(Challenge 1.1)**

2.1.2. Entity Classification

The types the entity has in the knowledge base may not suit all applications. In some cases, they are not sufficiently specific, or not all types of the entity are available (referring to the example above, Maradona was not only a football player, but also a manager).

The field of *fine grained entity typing* (e.g. [105]) aims to assign a specific type to an entity. When the set of target types is not known in advance, the task is sometimes called *open named entity typing* [107]. The challenge that we addressed was design of multi-lingual approach for extraction and disambiguation of hypernyms (as open entity types) from free text. **(Challenge 1.2)**

Another possible approach to entity classification is perceiving it as a document categorization task, when the entity has been previously disambiguated to a knowledge base entry providing a semi-structured description of the entity, which is used to construct a feature vector for classification. The challenge that we address is adapting existing research on document categorization to handle the multi-layer class hierarchies used with semantic knowledge bases. **(Challenge 1.3)**

2.1.3. Word Similarity Computation

With Word Similarity Computation (WSC) algorithms we can directly compute the similarity between the entity and a list of possible target classes. This approach can be best suited for more general entities, such as computing similarity between "mug" and "container".

WSC algorithms can be coarsely divided into the following two groups: *thesaurus-based measures*, such as Lin [65] or Resnik [81], and *distributional-based measures* trained on large corpora, such as Explicit Semantic Analysis (ESA) [27] and Neural Network Language Models [71]. Understanding of the performance of individual algorithms is a prerequisite of selection of the right WSC approach for a given entity classification problem. For example, the performance of thesaurus and corpora-based measures may substantially differ, depending on whether the entity in question is a general frequent word or noun phrase, or a rarely occurring name.

The performance of best known word similarity algorithms was evaluated on several benchmark datasets, the most well-known of which is WordSim353 [21]. However, it is currently widely acknowledged that this dataset is conceptually obsolete [1]. SimLex-999 [41] is a recently proposed dataset that addresses the shortcomings of WordSim353. The corresponding challenge is a critical analysis of SimLex-999, and design of complementary resources addressing any research gap in benchmarking word similarity algorithms not covered by SimLex-999. **(Challenge 1.4)**

2.1.4. Entity Salience

When representing text with entities, it is important not only to identify the entity and to disambiguate it, but also to correctly determine the level of salience (importance) of the entity in the given context. There are supervised entity algorithms, which are trained on a number of features derived from the entity mention, as well as from the local context and global context [29]. However, most research on entity salience focuses on entities that are not linked to knowledge bases. The corresponding research challenge is a design of a lexical resource for training and evaluation of entity salience algorithms that would also link the entities to a knowledge base. **(Challenge 1.5)**

2.1.5. Systems and Applications

Entity detection, disambiguation, classification and salience algorithms need to be implemented within a software system as a prerequisite to practical applications. Starting around 2011, the *DBpedia Spotlight system* [68] became one of the first complex approaches for performing *fine-grained* entity typing with Wikipedia. Many other entity typing algorithms and systems followed (cf. [86, 83] for a review).

The DBpedia Spotlight system had to rely on machine readable types in the DBpedia knowledge graph [64]. The availability of this information in DBpedia depended, at the time, on the presence of semistructured information in so called infoboxes and article categories. However, many articles in Wikipedia, from which DBpedia is populated, lack an infobox, in many cases the infobox also lacks the most precise type definition. The second limitation of DBPedia Spotlight was that the system relied on the periodic DBpedia exports (dumps). Since generation of DBpedia from Wikipedia is a computation-intensive task, a machine-readable information in DBpedia is often available with considerable delay compared to when that information has been added to Wikipedia. To improve the utility of entity-based representation of text, we envisaged a new entity classification system that would be able to use types extracted from free text of articles from live Wikipedia. **(Challenge 1.6)**

The availability of the structured information is an important limiting factor not only for applicability of entity classification, but also for general-purpose artificial intelligence systems. For example, in winning Jeopardy in the 2011 IBM Grand Challenge [19], the IBM Watson system relied, in part, on structured information amalgamated from a number of sources [31]. One of the largest of these resources was DBpedia. As noted above, the availability of types in DBpedia was largely limited to what could be extracted from the semistructured information in Wikipedia, affecting the quality of results provided by systems like IBM Watson. To improve the coverage of DBPedia, we decided to develop a complementary knowledge base populated from types extracted from free text of Wikipedia articles. **(Challenge 1.7)**

Entity-based representation is used in a number of applications. For example, analysis of social media streams such as Twitter has received considerable attention [4]. In our work, we initially focused on the use of algorithm ESA, which does not yet represent text in terms of machine readable entities, but rather as a vector of concepts that correspond to entries in a knowledge base. The first use case aimed at evaluation of the contribution of ESA to concept detection in video. **(Challenge 1.8)** In subsequent research, we aimed to evaluate the utility of the full entity-based representation of text for document categorization. **(Challenge 1.9)**

2.1.6. List of Challenges

Challenge no.	Description
1.1	Most frequent sense baseline for linking entities to a knowledge base.
1.2	Multi-lingual approach for type extraction and linking from free text
1.3	Algorithm based for supervised hierarchical classification of entities
1.4	Analysis of SimLex-999 and design of complementary benchmarking resources
1.5	Lexical resource for analyzing salience of entities linked to knowledge bases
1.6	Entity linking and classification system using live Wikipedia
1.7	Knowledge base population framework based on analysis of free text
1.8	Evaluation of the ESA algorithm for multimodal concept detection in video
1.9	Evaluation of bag-of-entities representation for document categorization

Table 2.1.: List of challenges (Entity-based text representation).

2.2. Contribution

2.2.1. Entity Linking

To address **Challenge 1.1**, we considered various modifications and combinations of Most-Frequent-Sense (MFS) based linking, entity co-occurrence-based linking, and the ESA-based linking. These algorithms were evaluated in the 2013 and 2014 editions of the NIST EDL task [44, 13]. The MFS method based on the Wikipedia Search has obtained the best B-cubed+ F1 score from our submissions, outperforming multiple algorithm runs submitted by other teams.

This confirmed our intuition that the open-source and widely deployed Wikipedia Search algorithm provides good balance between complexity, availability of implementation and quality of results. This finding drove the selection of Wikipedia-based MFS as a disambiguation and entity linking algorithm in our Linked Hypernyms Dataset and EntityClassifier frameworks outlined in the following (Section 2.2.5).

2.2.2. Entity Classification

To address **Challenge 1.2**, in [45] we introduced a complete unsupervised approach for extracting types for a given entity from the free text of a document describing the entity. As the type we used the first hypernym identified in the first paragraph of the document. To extract the hypernym, we used a custom-developed pattern-based approach relying on Java Annotation Patterns Engine (JAPE) grammars [9], a regular-expression like approach allowing to reference higher-level linguistic annotations.

The advantage of JAPE grammars over supervised algorithms is that JAPE grammars do not require labelled training data, which makes this approach portable to many languages.[4]

The estimated accuracy of raw plain text hypernyms exceeded 0.90 for all three languages for which the JAPE grammars were developed (German, English, Dutch), and the estimated accuracy of disambiguated hypernyms was between 0.77 for German and 0.88 for Dutch [45]. To our knowledge, paper [45] was the first study to report on accuracy of multilingual hypernym extraction from free text of Wikipedia articles.

The original THD approach suffered from two limitations. First, adding support for a new language requires input from a linguist, who needs to design the JAPE grammar. Second, the success of the extraction relies on the presence of a definition in the start of the document. For this reason, we set out to develop a complementary approach building upon statistical processing, rather than lexico-syntactic patterns. In further development, published in 2014 [54], we introduced a co-occurrence-based algorithm, which we named Statistical Type Inference. This enhanced the results of type extraction but did not completely eliminate the dependence on lexico-syntactic patterns. To address **Challenge 1.3**, in [55] we proposed an algorithm based on hierarchical Support Vector Machines (SVMs) [67]. The contribution was in adapting the algorithm to the case of hierarchies containing hundreds of classes in several layers, which is the case in the DBpedia ontology, typically used to assign machine-readable types to Wikipedia articles. Paper [45] (J1) is included in the Appendix A, and paper [55] (J2) is included in the Appendix B.

2.2.3. Word Similarity Computation

To address **Challenge 1.4**, we set out to contribute to benchmarking resources used for evaluation of word similarity algorithms, basing our work on the widely used SimLex-999 and WordSim-353 datasets. While analysing SimLex-999, the state-of-the-art word similarity resource, we concluded that it defines word similarity possibly overly narrowly for some applications. Simlex-999 essentially equates similarity with hyponymy-hyperonymy and synonymy, while some research has shown that antonyms are also highly similar – in fact, they are similar in all but one aspect, in which they are maximally opposed [102, 87].

[4] Certain limitation is that JAPE grammars are most effective, when they can refer language annotations, such as Part of Speech (POS) tags. This implies the need for availability of language parsers for the target language.

Our contribution [56] was not only a critical analysis of SimLex-999, but a development of several new benchmarking resources addressing its limitations. Most importantly, we developed two versions of annotation guidelines that consider antonymy as a similarity relation. In our first attempt called Explicit Similarity guidelines, we directly included antonymy as a similarity relation, in the refined Word INterchangeability (WIN) guidelines, the antonymy was contained implicitly. As part of this research, we also created several benchmarking resources for Czech. For example, Czech version of WordSim353 was reannotated using the WIN guidelines. The translated pairs were adopted from [8], WordSim-353 translation and reannotation performed at the Charles University. In addition to new lexical resources, in [56] we also provide benchmarking results for several WSC algorithms.

Paper [56] (J3) is included in Appendix C.

2.2.4. Entity Salience

There is a number of resources for evaluation of entity linking systems, but few contain information on salience of the entities. To address **Challenge 1.5**, we extended the list of entities in the Reuters-128 dataset with “common” entities (i.e. entities, which are not named entities) and with salience information [14]. Crowdsourcing system CrowdFlower[5] was used to collect the entity salience judgments on the scale: most salient, less salient and not salient. The advantage of this dataset compared to most existing resources, such as [16], is that the entities in Reuters-128 have been previously linked to the DBpedia knowledge graph. This allows to use the interconnected information in the knowledge graph to compute a range of additional features, such as PageRank [75], as demonstrated in [14].

2.2.5. Systems and Applications

Entity Classification Framework

To address **Challenge 1.6**, the EntityClassifier system for entity-based representation of text was designed [12]. This system used the main principles of the THD approach originally proposed in my dissertation thesis [59], and additionally integrates several other entity-based algorithms.

EntityClassifier system was added to Gerbil 1.2.5 [96, 86], which is possibly the most comprehensive entity benchmarking framework integrating 20 entity annotation systems. Another frequently used entity classification framework with which our system was integrated is NERD [83].[6]

One of the unique features of EntityClassifier was its ability to extract types on-the-fly from documents describing the entities (Wikipedia article text). The on-the-fly type extraction is particularly useful for newly emerged entities, which just had an article describing them added to Wikipedia. By focusing on the free text modality, this approach is complementary to DBPedia-live [73], a framework for extracting types from structured data in Wikipedia.

Paper [12] (C1) describing the EntityClassifier system is present in Appendix D.

[5] `http://figure-eight.com/` (CrowdFlower was rebranded as FigureEight)

[6] EntityClassifier was integrated with a later version of NERD (appearing as “THD” in [84])

Knowledge Base Population

To address **Challenge 1.7**, the entity classification algorithms developed to address Challenge 1.2 and Challenge 1.3 were integrated into an extraction framework, which can be used to generate datasets comprised of entity-type pairs extracted from free text of Wikipedia articles. The initial releases of the Linked Hypernyms Dataset (LHD) generated by the framework contained nearly five million entity-type assignments [45].

Soon after the first version was published, the German LHD dataset was imported to German DBpedia. The dump of the English LHD dataset is made available in the DBpedia release current as of writing, as well as in several previous releases (2016-04, 2016-10).[7] The contribution of the LHD datasets to DBpedia was recognized by the DBpedia TextExt Challenge Prize awarded to the LHD team in 2017.[8]

As shown in [45], by exploiting the textual modality, the types extracted with the proposed approach are complementary to types generated by the state-of-the-art DBPedia type enrichment algorithm SDtype [76], the results of which are also available in DBpedia.

The initial version of the implementation of the Linked Hypernyms Dataset Framework was described in [57].

Multimodal Video Classification

To address **Challenge 1.8**, in [28] we evaluated suitability of Wikipedia-based word similarity algorithm ESA for performing multimodal fusion. The data used were sourced from TRECVID 2012 Semantic Indexing (SIN) task dataset. The transcripts of Automatic Speech Recognition (ASR) were used as input for ESA. For processing of the visual modality, linear SVMs were used, which output a degree of confidence for each of the 125 target concepts. The main contribution of the paper was exploration of multiple fusion strategies for combining the feature vectors generated from text and video.

Text Categorization

To address **Challenge 1.9**, in [61], we evaluated whether the entity-based representation can improve text classification when used as a replacement for the Bag-of-Words (BoW) approach. The experiments were performed on the Reuters-21578 collection, which is frequently used for benchmarking of information retrieval algorithms. As part of the preprocessing, a "Bag of Entities" (BoE) feature set was created from the underlying dataset, using EntityClassifier to detect entities. Entities with a low Term Frequency-Inverse Document Frequency (TF-IDF) were dropped, decreasing the length of the input vector. As the classification algorithm, we used a modified version of CBA [50], which is an interpretable rule-based classifier discussed in the following chapter.

We compared the performance of the standard (at the time) BoW approach with the proposed BoE approach. The evaluation suggests that BoE results in a small improvement in F-Measure over BoW when the length of the input vector is small.

We see the contribution of this research in the following: to our knowledge, [61] was the first approach to use entity-based representation for document categorization. Through rules learnt over semantically interpretable entities, we effectively created an "interpretable" document

[7] `http://downloads.dbpedia.org/2016-04/core-i18n/en/`,`http://downloads.dbpedia.org/2016-10/core/`,`http://downloads.dbpedia.org/current/core/`

[8] `https://wiki.dbpedia.org/textext`

classifier. By using a standard dataset for which many previous results have been published, we made it easy to compare the performance of previous “black box” models with the proposed interpretable model. Overall, we concluded that the Bag-of-Entities approach has properties that make it favourable for use in preference learning (recommender) systems applications, but also that more research is needed to close the performance gap between the proposed approach and state-of-the-art black-box models.

3. Association Rule Learning and its Applications

3.1. Motivation and State-of-the-art

Rule-based methods belong between popular techniques in machine learning and data mining, with the discovered rules corresponding to regularities in data that can be expressed in the form of an IF-THEN rule [23]. Rule learning typically either serves a descriptive or predictive purpose. *Descriptive rule learning* aims to discover interesting patterns in data and present these in the form of human understandable rules. *Predictive rule learning* aims to create a collection of rules, which covers the entire instance space. The resulting classifier composed of rules can be used to assign a class to new instances. The advantages of rule-based classifiers as opposed to other commonly used classifier representations, such as random forests or Artificial Neural Networks (ANNs), include typically faster learning times (particularly as opposed to neural networks), and better interpretability. Unlike rules, ANNs lack a straightforward natural explanation, which requires application of additional algorithms [85, 82] to explain these models to users. The disadvantage of rules is lower reported average accuracy [18].

One of the first rule learning approaches was the AQ algorithm proposed by Ryszard Michalski in the 1960's [69]. We provide a brief overview of AQ and selected succeeding rule learning algorithms in [24]. Detailed account of the foundations of the most commonly used rule learning algorithms is presented in [23]. In this thesis, we focus on association rule learning, which is one of the subfields of rule learning. The concept of association rules is attributed to Rakesh Agarwall, who invented the Apriori algorithm [2] for fast discovery of association rules.[1]

The Apriori algorithm operates in two phases. First, all *frequent itemsets* (i.e., conditions that cover a certain minimum number of examples) are found. The minimum number of examples (instances) is specified by the user and is called a *minimum support* threshold. In a second phase, frequent itemsets are converted into association rules. The association rules need to meet a second threshold specified by the user, which is called *minimum confidence*, and corresponds to minimum observed conditional probability of the consequent of the rule (prediction) given the antecedent (set of conditions).

While association rule learning was initially conceived as a method of descriptive rule learning, it was later adapted for a range of tasks, such as clustering [22], anomaly detection [40] and classification [66]. In our research, we dealt with the problem of reducing the size of the model, which is applicable both to descriptive rule learning and classifier building. We also explored some of the other uses for association rules.

[1] The attribution is made, for example, by the Encyclopedia of Machine Learning and Data Mining [94]. According to [38], the notion of association rules was introduced already in mid 1960's by Petr Hájek [37].

3.1.1. Reducing the Number of Rules

Following the recent introduction of legal requirements, such as the General Data Protection Regulation (GDPR), some machine learning models, including those used for recommendation, need to be comprehensible. In particular, Articles 13–15 of GDPR provide rights to "meaningful information about the logic involved" in automated decisions [88].

While rules in general are well suited to provide such meaningful explanation to the end user, one of the limiting factors for the use of association rules is a high number of frequent itemsets that can often be discovered even for very small datasets due to combinatorial explosion [15]. The high number of discovered itemsets, and consequently rules, does not only have computational costs, but also severely impedes interpretability of the model. While a single association rule is typically easily interpretable, how does one interpret one million rules?

Number of approaches for addressing this problem has been proposed. These initially focused on the development of quantifiable interest measures [30], which can be used to select the rules that are supposed to be interesting for the human user. Another research direction aims at pruning the discovered rules – removing redundancies. The main limitation of rule interest measures and pruning algorithms is that they generally work only with information available in the analysed data.

When the user possesses domain knowledge not contained in the data, different selection of interesting rules needs to be applied. For this purpose, approaches have been developed that first elicit knowledge from the user, and then use this knowledge to filter out rules that are logical consequences of the items of this knowledge [79]. The corresponding research challenge is to devise a system following some of the earlier proposed principles for elicitation of domain knowledge from experts [78], advancing the subsequent use of the collected knowledge for identification of subjectively interesting rules. **(Challenge 2.1)**

Decoupling elicitation of the background knowledge from the user from its use during mining or filtering of the discovered rules can be inefficient and ineffective. For some domains, the body of potentially relevant knowledge is large, leading to high demands on the time of the domain expert. Furthermore, it cannot be ruled out that the important pieces of knowledge would not be elicited, or that the elicited knowledge would not be used.

To address this problem, interactive software has been developed that lets the users express their domain knowledge during mining as constraints on the search space. Making Interactive Mining Easy (MIME) [32] is an example of a desktop-based system for frequent pattern and rule mining. In our research, we aimed to advance the field of the interactive "user-in-the-loop" systems by exploring the possibilities provided by the web environment for restricting the search space **(Challenge 2.2)** and editing the resulting model **(Challenge 2.3)**.

In subsequent research, we aimed to identify suitable previously proposed algorithms that could be applied to the "too many discovered rules" problem without access to domain knowledge or user feedback. **(Challenge 2.4)** As discussed in detail in the contributions section below, our survey pointed to the CBA algorithm [66] as a potential base approach for reduction of rules discovered by association rule learning.[2] In addition to pruning, CBA also converts the discovered rules to a classifier.

CBA can be used to prune also results of mining with the GUHA ASSOC procedure [37], which continues to be developed at DIKE [80]. In contrast to the Apriori algorithm, GUHA ASSOC mines for generalized association rules, which can also contain the negation and disjunction connectives. As part of **Challenge 2.4**, we aimed to perform a preliminary study

[2] CBA is applicable only if the consequent of the rules is constrained to one specific attribute

of the hypothesis that higher expressiveness of GUHA ASSOC rules might result in smaller size of the models.

3.1.2. Design of Data Formats for Machine Learning Systems

The typical machine learning environment consists of a machine learning software that is used to create a model, and a scoring engine, which applies the model to new use cases. When domain knowledge from experts is to be utilized, additional components are needed for collection and preprocessing to a form that is usable in further steps in the machine learning process.

The industry standard data format used to exchange models between machine learning software and scorers is Predictive Model Markup Language (PMML) [34]. As of writing, the most up-to-date version of PMML is version 4.3.[3] PMML focuses on support for mainstream machine learning models, such as decision trees or neural networks. The challenge being addressed included development of PMML-based formats for a) support of background (domain) knowledge, b) association rule-based models with extended expressivity (GUHA-ASSOC), c) anomaly models based on frequent itemsets. **(Challenge 2.5a–c)**

In some cases, machine learning models can be exchanged also with existing systems in other domains. This is the case of Business Rule Management Systems (BRMS). The corresponding problem to address is the possibility to transform discovered association rules to business rules. **(Challenge 2.5d)**

3.1.3. Systems and Applications

Self-service Rule-based Machine Learning Framework

There are several general purpose platforms for machine learning and classification. In particular, WEKA (Waikato Environment for Knowledge Analysis) was one of the first systems that adopted the philosophy to "move away from supporting a computer science or machine learning researcher, and towards supporting the end user of machine learning" [42, 103].

Systems like WEKA provide generic interfaces that allow the machine learning researcher to add their algorithm. This is suitable for addressing common types of tasks with standardized requirements on inputs and outputs. LISp-Miner [90], developed at DIKE, is a representative of a more focused system from knowledge discovery from databases, which provides specialized user interfaces for several GUHA procedures, including GUHA ASSOC.

Most mainstream machine learning systems have been developed as desktop-based solutions. To reduce maintenance costs and improve availability to users, these systems are now complemented by Machine Learning as a Service (MLaaS) platforms [106]. The challenge to be addressed was design of a self-service web-based MLaaS system based on rules. **(Challenge 2.6)**

Recommender system using entities and rules

Content-based recommendation relies either on explicit information on user interest, or on those user actions (implicit feedback), which could be interpreted as a manifestation of user (dis)interest in a certain object. While the latter does not require the user to perform any extra activity, the information obtainable in this way on a particular content item is often restricted

[3] `http://dmg.org/pmml/v4-3/GeneralStructure.html`

to several discrete actions (e.g. user opening a web page) and the duration between them (the time spent on a web page). Eye tracking was relatively recently proposed as an effective way of obtaining highly detailed user feedback [7]. Processing streams of data from sensors, such as eye trackers, provides new opportunities for improving the quality recommendations, but also a technical challenge to which recommender systems need to adapt.

Another impetus for change in the design of recommender system comes from the description of content. Many early recommender approaches adopt collaborative filtering, which provide suggestions based on co-occurrences of items in user histories. Subsequent research yielded variety of algorithms that also take advantage of the description of the individual items, which is often available in the form of a textual documents [3]. While multiple recommender systems try to derive semantics from such information using techniques like Latent Semantic Indexing (LSI) [11], few can work with machine-readable semantics sourced from external knowledge bases [62].

Not only technological advances in other fields, but also regulatory requirements drive advancement in recommender systems. Following Article 22 of GDPR models should allow for “human intervention”. In some applications the choice of the algorithm may thus be influenced by the amenability of the model representation to user changes.

The challenge addressed relates to design of a proof-of-concept framework integrating support for streaming input, with semantic description of content being interacted with sourced from a knowledge base. The user models should be interpretable, and the knowledge representation used should give the user the possibility to edit the model. **(Challenge 2.7)**

3.1.4. List of Challenges

Challenge no.	Description
2.1	Design of proof-of-concept of domain knowledge-facilitated pruning
2.2	Design of proof-of-concept system for interactive state-space restrictions
2.3	Design of proof-of-concept system for editable rule models
2.4	Analysis of the CBA algorithm
2.5a	Extension to PMML supporting generalized association rules (GUHA-ASSOC)
2.5b	PMML-based model for exchange of domain knowledge
2.5c	Extension to PMML for anomaly detection
2.5d	Interoperability of GUHA-ASSOC models with BRMS systems
2.6	Design of proof-of-concept system for self-service rule learning
2.7	Design of proof-of-concept “interpretable” recommender system

Table 3.1.: List of challenges (Association rule learning).

3.2. Contribution

3.2.1. Reducing the Number of Rules

In the following, we briefly describe our contributions to approaches for pruning discovered rules, referencing individual detailed studies.

Domain-knowledge Facilitated Pruning

To address **Challenge 2.1**, with the SEWEBAR-CMS system [58] we attempted to tackle the problem of too many discovered rules by facilitating the exchange of information between domain experts and the rule learning process. The proposed system allowed the users to define what they know about the relations between pairs of attributes in the given domain using the formalisms developed in [78, 77]. This information was then used to filter out uninteresting rules. For the filtering purposes, a data mining ontology queried by the Tolog language was developed [51] using the ISO/IEC 13250:2003 standard.

Interactive State-space Restrictions

To address **Challenge 2.2**, in [91] we presented a prototype system that made the formulation of the mining task look somewhat similar to web search. The user interface put emphasis on interactivity and user-defined constraints. The user could gradually refine the search space to avoid the combinatorial explosion and overload by the discovered rules.

Manual Edits of the Model

To address the **Challenge 2.3**, in [97], we presented a proof-of-concept human-in-the-loop machine learning system. By providing the edit capability, the users had the option to manually reduce the size of the model by dropping rules or conditions within rules. The system, introduced in 2014, falls within the scope of interactive Machine Learning (iML), a field which has only recently received increased attention [35, 43]. The standard iML approach typically requires a specific type of feedback from the user to use in further learning. In contrast, our system first learns a model, and then gives the user the possibility to edit the model.

Automated Pruning

To address **Challenge 2.4**, in [50] we analysed the CBA algorithm, which is possibly the most widely used algorithm for classification with association rules. The results confirmed the suitability of CBA for pruning, as the accuracy of the models produced after pruning was virtually the same as model accuracy obtained with the full list of class association rules. We also inspected the effect of using a learning algorithm outputting more expressive association rules such as containing a negation. In our preliminary experiments, this led to substantial increase in computational demands and a negligible positive effect on accuracy and rule count. Finally, we also investigated the effect of various settings of confidence and support thresholds on the performance of CBA models. This preliminary investigation on several UCI datasets, reported on in [50], was followed-up by two more focused studies.

In [49], we performed empirical evaluation of CBA and several other algorithms on real world data from the CLEF NewsReel 2014 Challenge [5]. The results indicate that CBA provided recommendations competitive to other symbolical approaches. Following this preliminary study on the offline data, the CBA algorithm was adopted for a subsequent submission to the CLEF NewsReel 2017 Challenge by a team from the Czech Technical University [33]. In [49], we also provided analysis of the sensitivity of the results of pruning to the amount of input data. The results could be used to improve performance of CBA-based solutions in environments where fast classifier building is required.

In [20], we investigate the two variations of the CBA algorithm known as M1 and M2, which were originally proposed in [66]. Following the recommendation of the CBA authors, most software implementations of CBA adopt M2 as the only or default version. In this paper, we empirically show that M1 is better suited to take advantage of optimized vectorized operations, supported in modern environments focused on data science such as SciKit learn.[4]

Paper [50] (C2) is included in the Appendix E.

3.2.2. Design of Data Formats for Machine Learning Systems

To address **Challenge 2.5a-d**, we devised four data formats described in the following:

(a) The `Association Rules` model in PMML is restricted to rules composed of conjunctions of items (attribute-value pairs). Our proposal [52], inspired by GUHA-ASSOC, supports models containing disjunctions and negations in association rules, along with some other extensions.

(b) PMML does not contain any mechanism for capturing user's knowledge of the problem. In [53], we proposed a PMML-based model focused on exchanging domain knowledge relating to given dataset, covering both individual attributes and their known pair-wise interactions.

(c) The latest PMML specification as of writing does not contain a dedicated model for anomaly detection. In [60], we proposed an anomaly detection model based on a frequent pattern-based anomaly detection algorithm [40]. The proposal is included in the PMML RoadMap for PMML 4.4.

(d) The rule learning task can yield a set of rules that can be used instead of manually specified business rules. To improve interoperability between the two kinds of rule-based systems, in [98] we cover the problem of transforming GUHA-based association rules discovered by LISp-Miner to the DRL format, used in the leading open source BRMS system Drools.[5]

3.2.3. Systems and Applications

Self-service Rule-based Machine Learning Framework

To address **Challenge 2.6**, EasyMiner[6] system was conceived as a self-service platform for association rule learning. In the initial prototype [91], the LISp-Miner desktop system was used to find association rules, which allowed EasyMiner to provide multiple features not present in standard association rule learning implementations, such as support for binning wild cards (called coefficients in LISp-Miner), negation and disjunction logical connectives, and variety of interest measures. The user interaction was confined to a single "screen", which allowed to define preprocessing, formulate constraints for mining and inspect the discovered rules.

In a later EasyMiner development [99], the research on EasyMiner refocused on association rule classification based on CBA. The addition of CBA allowed the users to activate pruning, reducing the size of the produced models, as well as to create classification models – CBA

[4] `http://scikit-learn.org`

[5] `http://drools.org/`

[6] Originally written as "I:ZI Miner', following the Czech acronym of DIKE ("KIZI").

adds a default rule to the end of the model, resulting in the model covering all conceivable instances.

Paper [48] summarizes the EasyMiner development, and paper [99] (J4), covering the current version, is presented in the Appendix F.

Recommender system using entities and rules

To address **Challenge 2.7**, we designed InBeat as a generic framework for user tracking and preference learning. Its "SMART-TV use case" was first introduced at ACM RecSys'13 [63], and in [62] we presented an extended version of the system, which also performed physical behaviour tracking using Microsoft Kinect, a widely accessible commodity hardware. The documents describing the items interacted with can be described with entities. In our experiments, we used our entity recognition system EntityClassifier (see Section 2.2.5). Entities are assigned a type from several knowledge bases, including LHD (described also in the previous chapter and Appendix A–B). InBeat was evaluated in a video recommender setting, described in the following example.

Example. (InBeat in SmartTV use case) The supplementary material to the InBeat system [62] presents a football fan watching a video stream composed of clips from football videos, general news and commercials. Subtitles are used to detect "pseudo-shots", fragments of the video between the start and end offsets of a subtitle. Consider subtitle fragment: *Luiz Felipe Scolari wants to combine the experience of the former Barcelona, AC Milan and Paris Saint-Germain star with young talent like Neymar.* Entities in this fragment are disambiguated to DBpedia resources. For example, for Neymar the following types are retrieved: SoccerPlayer, Athlete, Agent and Person. As illustrated in Table 3.2, appearance of an entity in a subtitle activates one or more types in the ontology. The activation is spread up to the root class.
After applying association rule learning on the semantic description of content combined with the level of user interest derived from the implicit tracking, the system arrived (among others) at the following preference rules:

- `SoccerPlayer` **then** `interest=positive`
- `Global_City=YES` **and** `Science=YES` **then** `interest=negative`

Note that the level of interest in Table 3.2, used as input to association rule learning, was derived from whether the user was watching the screen or not.

Identification			Description			
userId	sessionId	pseudo shotId	dbp: Neymar	dbo:SoccerPlayer	...	Interest
user1	1124541	125	yes	yes	...	positive

Table 3.2.: Example training instance. *dbo:* and *dbp:* are prefixes in the DBpedia knowledge graph.

Presenting the user model in the form of rules provides the user with the possibility to edit the model, as demonstrated in [97] and discussed in Section 3.2.1. Software support for such functionality exposed to the end-user is a matter of future work.

Paper J5 describing the InBeat system is presented in Appendix G.

4. Overview of Contribution

In this chapter, we provide an overview of papers comprising the contributions described in this thesis. Section 4.1 covers the most significant publications reprinted in Appendix A–G. Section 4.2 contains a list of selected other papers, which are referenced from Chapters 2–3.

4.1. Overview of Publications Reprinted in Appendices

Figure 4.1 specifies the relation of the selected papers to the topic of the thesis. This figure also roughly categorizes the papers as either algorithm research or applied research / software systems. The following tables provide further details for the journal (Table 4.1) and conference (Table 4.2) papers included in the thesis. The meaning of columns in these tables is as follows:

- **ID:** Paper identifier.
- **Impact factor** (only Table 4.1): value of Thomson Reuters impact factor for the year preceding the publication date.
- **CORE Rank** (only Table 4.2): CORE Rank[1] of the conference valid for the year when the paper was published.[2]
- **Contribution**: Author's share/contribution on the paper. For journal articles the contribution shares were confirmed by the co-authors. For conference papers the stated contribution is proportional to the number of authors of the paper.

Papers included in these two tables are in full reprinted in the Appendix. Permission to include articles J1–J5, published in Elsevier journals, is not required, since as the author I retain the right to include journal articles published with Elsevier in a thesis or dissertation, provided it is not published commercially.[3] Springer Nature granted a license to include paper C1 in this thesis under number 4423620262951, and a license to include paper C2 in this thesis under number 4423621260470.

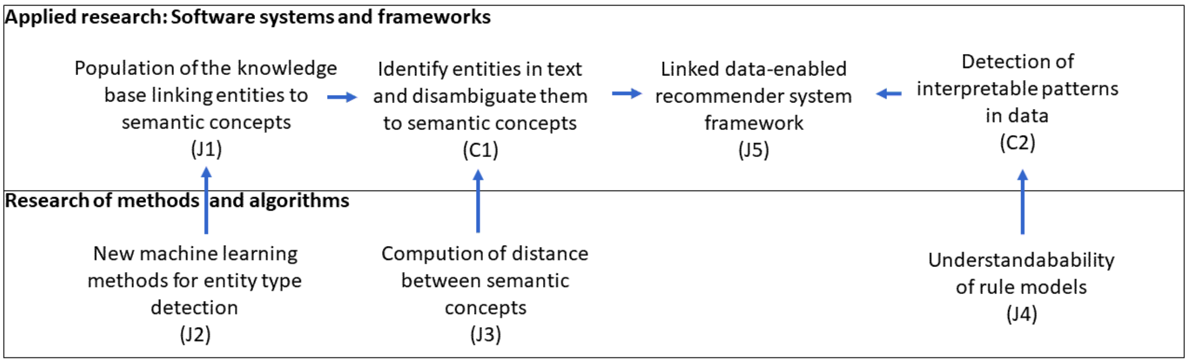
Statement of Contribution Research in computer science is typically a collaborative endeavour, resulting in multi-author publications. For this thesis, it is desirable to generally qualify my contribution with respect to the works covered.

I either wrote or significantly contributed to writing of all included papers. In papers J1-J3, C2, I was responsible for or significantly contributed to the technical work, including implementation of the software, and execution of experiments. In papers C1, J4, J5, my primary role was of the initiator of the research – I formulated the research problem, and steered the

[1] http://www.core.edu.au/conference-portal

[2] Unlike the journal impact factor, which is published after the year has ended, CORE Rankings are published in advance. For example, CORE 2018 Ranking were available already on 16th Dec 2017 (http://www.core.edu.au/conference-portal/2018-conference-rankings-1).

[3] https://www.elsevier.com/about/policies/copyright#Author-rights


Applied research: Software systems and frameworks

Population of the knowledge base linking entities to semantic concepts (J1) → Identify entities in text and disambiguate them to semantic concepts (C1) → Linked data-enabled recommender system framework (J5) ← Detection of interpretable patterns in data (C2)

Research of methods and algorithms

New machine learning methods for entity type detection (J2)

Compution of distance between semantic concepts (J3)

Understandabability of rule models (J4)

Figure 4.1.: Relation between the five journal papers (J1–J5) and the two conference papers (C1–C2) included in the Appendix

projects, defining functional requirements and contributing to the design of software architecture. My contribution to implementation for these three papers was roughly as follows: for C1, I provided initial THD implementation, for J4 I performed proof-of-concept implementation of CBA and some visualizations. For J5, I provided a proof-of-concept implementation of the GAIN module, which was extended and rewritten in `node.js` by the main author of J5.

Paper J1 is a single author paper. Papers J2–J4 contain a joint declaration of contribution made by the authors in the acknowledgment section. For article J5, where the declaration is not directly included, the contribution statement approved by both authors, Jaroslav Kuchař (JK) and Tomáš Kliegr (TK), follows: “JK wrote the software code. Both JK and TK designed the algorithms and system architecture. JK authored the figures and tables. TK and JK wrote the text of the manuscript. TK conceived the research idea.”

Table 4.1.: Journal articles

ID	Title	Impact factor	Publisher	Contribution
J1	Kliegr, Tomáš. Linked hypernyms: Enriching DBpedia with targeted hypernym discovery. Web Semantics: Science, Services and Agents on the World Wide Web 31 (2015): 59-69.	2.55 (2014)	Elsevier	100%
J2	Kliegr, Tomáš, and Ondřej Zamazal. LHD 2.0: A text mining approach to typing entities in knowledge graphs. Web Semantics: Science, Services and Agents on the World Wide Web 39 (2016): 47-61.	1.277 (2015)	Elsevier	50%
J3	Kliegr, Tomáš, and Ondřej Zamazal. "Antonyms are similar: paradigmatic association approach to rating similarity in SimLex-999 and WordSim-353". Data and Knowledge Engineering (2018)	1.467 (2017)	Elsevier	70%
J4	Stanislav Vojíř, Václav Zeman, Jaroslav Kuchař, Tomáš Kliegr, EasyMiner.eu: Web framework for interpretable machine learning based on rules and frequent itemsets, Knowledge-Based Systems, 2018,	4.396 (2017)	Elsevier	25%
J5	Jaroslav Kuchař, Tomáš Kliegr, InBeat: JavaScript recommender system supporting sensor input and linked data, Knowledge-Based Systems, Volume 135, 2017.	4.529 (2016)	Elsevier	40%

Table 4.2.: Conference papers

ID	Title	CORE	Publisher	Contribution
C1	Milan Dojchinovski, and Tomáš Kliegr. Entity-Classifier.eu: real-time classification of entities in text with Wikipedia. Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Springer, Berlin, Heidelberg, 2013.	A (2013)	Springer	50%
C2	Kliegr, Tomáš, Jaroslav Kuchař, Davide Sottara, and Stanislav Vojíř. Learning business rules with association rule classifiers. In International Workshop on Rules and Rule Markup Languages for the Semantic Web, pp. 236-250. Springer, Cham, 2014..	B (2014)	Springer	25%

4.2. Overview of Other Selected Publications

Entity-based text representation

- Kliegr, T., Chandramouli, K., Nemrava, J., Svatek, V., & Izquierdo, E. (2008, August). Combining image captions and visual analysis for image concept classification. In Proceedings of the 9th International Workshop on Multimedia Data Mining: held in conjunction with the ACM SIGKDD 2008 (pp. 8-17). ACM.
- Kliegr, T., & Zamazal, O. (2014). Towards Linked Hypernyms Dataset 2.0: complementing DBpedia with hypernym discovery and statistical type inferrence. In Proceedings of The Ninth International Conference on Language Resources and Evaluation, LREC.
- Galanopoulos, D., Dojchinovski, M., Chandramouli, K., Kliegr, T., & Mezaris, V. (2015). Multimodal Fusion: Combining Visual and Textual Cues for Concept Detection in Video. In Multimedia Data Mining and Analytics (pp. 295-310). Springer, Cham.
- Dojchinovski, M., Reddy, D., Kliegr, T., Vitvar, T., & Sack, H. (2016). Crowdsourced Corpus with Entity Salience Annotations. In Proceedings of The Tenth International Conference on Language Resources and Evaluation, LREC .

Rule learning

- Kliegr, T., Ralbovský, M., Svátek, V., Šimůnek, M., Jirkovský, V., Nemrava, J., & Zemánek, J. (2009, September). Semantic analytical reports: A framework for post-processing data mining results. In International Symposium on Methodologies for Intelligent Systems (pp. 88-98). Springer, Berlin, Heidelberg.
- Kliegr, T., & Rauch, J. (2010, October). An XML format for association rule models based on the GUHA method. In International Workshop on Rules and Rule Markup Languages for the Semantic Web (pp. 273-288). Springer, Berlin, Heidelberg.
- Kliegr, T., Hazucha, A., & Marek, T. (2011, August). Instant feedback on discovered association rules with PMML-based query-by-example. In International Conference on Web Reasoning and Rule Systems (pp. 257-262). Springer, Berlin, Heidelberg.
- Škrabal, R., Šimůnek, M., Vojíř, S., Hazucha, A., Marek, T., Chudán, D., & Kliegr, T. (2012, September). Association rule mining following the web search paradigm. In Joint European Conference on Machine Learning and Knowledge Discovery in Databases (pp. 808-811). Springer, Berlin, Heidelberg.
- Kuchař, J., & Kliegr, T. (2013, October). GAIN: web service for user tracking and preference learning-a SMART TV use case. In Proceedings of the 7th ACM conference on Recommender systems (pp. 467-468). ACM.
- Kliegr, T., & Kuchař, J. (2014, August). Orwellian Eye: Video Recommendation with Microsoft Kinect. In Frontiers in Artificial Intelligence and Applications, ECAI. IOS PRESS
- Kliegr, T., & Kuchař, J. (2015, September). Benchmark of rule-based classifiers in the news recommendation task. In International Conference of the Cross-Language Evaluation Forum for European Languages (pp. 130-141). Springer, Cham.

- Fürnkranz, J., & Kliegr, T. (2015, August). A brief overview of rule learning. In International Symposium on Rules and Rule Markup Languages for the Semantic Web (pp. 54-69). Springer, Cham.

5. Conclusion and Future Work

In this thesis, we described several methods for improving the performance of entity classification and rule learning algorithms. These resulted in several new algorithms for hierarchical classification of entities described by free-text documents according to prespecified taxonomy: a lexico-syntactic pattern-based approach, an ontology-aware co-occurrence algorithm, and a hierarchical SVM approach. The algorithms were embedded into a type extraction framework, which used Wikipedia as the input, and created enrichments for the DBpedia knowledge graph. The EntityClassifier system was designed as a proof-of-concept application demonstrating the ability of the framework to adapt to change in the input text. We also contributed to benchmarking resources, for example, through the collection of datasets for evaluating word similarity computation algorithms according to paradigmatic association.

The second track of the research focused predominantly on improving interpretability of association rule-based classifiers by reducing the size of the created models. A supplementary contribution was made by making rule-based models more comprehensible by enhancing the interconnection between domain knowledge and models through proposals of new data formats. Throughout our research, EasyMiner, a web-based rule learning framework, served as a source of new ideas, as well as a testbed for some of the solutions.

Building upon research on understanding of the CBA algorithm used in EasyMiner, we work on a quantitative version of the CBA algorithm [46], which allows to use numerical attributes within the rule mining process to reduce the size of the produced models and improve accuracy.

With InBeat, we combine entity classification and rule learning with the intent to generate semantic user profiles. The system used sensors to determine the level of user interest, hinting at one possible direction of future work, which is to align proxies for various mental states, such as the level of interest, with specific semantic concepts. This could be useful in analysing the comprehensibility of hypotheses learnt from data, such as rules. In our research on rule pruning, we assumed that smaller models are more comprehensible. While such assumption is common in the literature, there is little evidence that this is so [92]. In our current work [25, 47], we aim to analyse individual factors that affect comprehensibility of rules, including model size.

Bibliography

[1] Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor Soroa. A study on similarity and relatedness using distributional and WordNet-based approaches. In *NAACL '09*, NAACL '09, pages 19–27, Stroudsburg, PA, USA, 2009. ACL.

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[3] Jesús Bobadilla, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. Recommender systems survey. *Knowledge-based systems*, 46:109–132, 2013.

[4] Kalina Bontcheva and Dominic Rout. Making sense of social media streams through semantics: a survey. *Semantic Web*, 5(5):373–403, 2014.

[5] Torben Brodt and Frank Hopfgartner. Shedding light on a living lab: the clef newsreel open recommendation platform. In *Proceedings of the 5th Information Interaction in Context Symposium*, pages 223–226. ACM, 2014.

[6] Volha Bryl, Christian Bizer, and Heiko Paulheim. Gathering alternative surface forms for dbpedia entities. In *NLP-DBPEDIA@ ISWC*, pages 13–24, 2015.

[7] Georg Buscher, Andreas Dengel, and Ludger van Elst. Eye movements as implicit relevance feedback. In *CHI '08 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '08, pages 2991–2996, New York, NY, USA, 2008. ACM.

[8] Silvie Cinkova. Wordsim353 for Czech. In *International Conference on Text, Speech, and Dialogue*, pages 190–197. Springer, 2016.

[9] Hamish Cunningham, Diana Maynard, and Valentin Tablan. JAPE - a Java Annotation Patterns Engine (Second edition), Department of Computer Science, University of Sheffield, 2000. Technical report, 2000. Technical Report.

[10] Edward Curry, Andre Freitas, and Sean O'Riain. The role of community-driven data curation for enterprises. In *Linking enterprise data*, pages 25–47. Springer, 2010.

[11] Scott Deerwester, Susan T Dumais, George W Furnas, Thomas K Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[12] Milan Dojchinovski and Tomas Kliegr. Entityclassifier. eu: real-time classification of entities in text with Wikipedia. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 654–658. Springer, 2013.

[13] Milan Dojchinovski, Ivo Lasek, Tomas Kliegr, and Ondrej Zamazal. Wikipedia search as effective entity linking algorithm. In *TAC*, 2013.

[14] Milan Dojchinovski, Dinesh Reddy, Tomas Kliegr, Tomas Vitvar, and Harald Sack. Crowdsourced corpus with entity salience annotations. In *LREC*, 2016.

[15] Jugendra Dongre, Gend Lal Prajapati, and SV Tokekar. The role of apriori algorithm for finding the association rules in data mining. In *Issues and Challenges in Intelligent Computing Techniques (ICICT), 2014 International Conference on*, pages 657–660. IEEE, 2014.

[16] Jesse Dunietz and Daniel Gillick. A new entity salience task with millions of training examples. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, volume 2: Short Papers*, pages 205–209, 2014.

[17] Christiane Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010.

[18] Manuel Fernandez-Delgado, Eva Cernadas, Senén Barro, and Dinani Amorim. Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1):3133–3181, 2014.

[19] David A Ferrucci. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3.4):1–1, 2012.

[20] Jiri Filip and Tomas Kliegr. Classification based on associations (cba) – a performance analysis. RuleML Challenge 2018, 2018. To appear.

[21] Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: the concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131, January 2002.

[22] Benjamin CM Fung, Ke Wang, and Martin Ester. Hierarchical document clustering using frequent itemsets. In *Proceedings of the 2003 SIAM international conference on data mining*, pages 59–70. SIAM, 2003.

[23] Johannes Furnkranz, Dragan Gamberger, and Nada Lavrač. *Foundations of rule learning*. Springer Science & Business Media, 2012.

[24] Johannes Furnkranz and Tomas Kliegr. A brief overview of rule learning. In *International Symposium on Rules and Rule Markup Languages for the Semantic Web*, pages 54–69. Springer, 2015.

[25] Johannes Furnkranz, Tomas Kliegr, and Heiko Paulheim. On cognitive preferences and the interpretability of rule-based models. *arXiv preprint arXiv:1803.01316*, 2018.

[26] Ryan Gabbard, Jay DeYoung, Constantine Lignos, Marjorie Freedman, and Ralph Weischedel. Combining rule-based and statistical mechanisms for low-resource named entity recognition. *Machine Translation*, 32(1-2):31–43, 2018.

[27] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[28] Damianos Galanopoulos, Milan Dojchinovski, Krishna Chandramouli, Tomas Kliegr, and Vasileios Mezaris. Multimodal fusion: Combining visual and textual cues for concept detection in video. In *Multimedia Data Mining and Analytics*, pages 295–310. Springer, 2015.

[29] Michael Gamon, Tae Yano, Xinying Song, Johnson Apacible, and Patrick Pantel. Identifying salient entities in web pages. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2375–2380. ACM, 2013.

[30] Liqiang Geng and Howard J Hamilton. Interestingness measures for data mining: A survey. *ACM Computing Surveys (CSUR)*, 38(3):9, 2006.

[31] Alfio Gliozzo, Or Biran, Siddharth Patwardhan, and Kathleen McKeown. Semantic technologies in IBM watson. In *Proceedings of the fourth workshop on teaching Natural Language Processing*, pages 85–92, 2013.

[32] Bart Goethals, Sandy Moens, and Jilles Vreeken. MIME: a framework for interactive visual pattern mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 757–760. ACM, 2011.

[33] Christian Golian and Jaroslav Kuchar. News recommender system based on association rules at clef newsreel 2017. In *Working Notes of the 8th International Conference of the CLEF Initiative, Dublin, Ireland. CEUR Workshop Proceedings*, 2017.

[34] Alex Guazzelli, Michael Zeller, Wen-Ching Lin, Graham Williams, et al. PMML: An open standard for sharing models. *The R Journal*, 1(1):60–65, 2009.

[35] Iryna Gurevych, Christian M. Meyer, Carsten Binnig, Johannes Fürnkranz, Stefan Roth, and Edwin Simpson. Interactive Data Analytics for the Humanities. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing: Proceedings of the 18th International Conference*, Lecture Notes in Computer Science. Berlin/Heidelberg: Springer.

[36] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R Curran. Evaluating entity linking with Wikipedia. *Artificial intelligence*, 194:130–150, 2013.

[37] Petr Hajek, Ivan Havel, and Metodej Chytil. The GUHA method of automatic hypotheses determination. *Computing*, 1(4):293–308, 1966.

[38] Petr Hajek, Martin Holena, and Jan Rauch. The GUHA method and its meaning for data mining. *Journal of Computer and System Sciences*, 76(1):34 – 48, 2010. Special Issue on Intelligent Data Analysis.

[39] Petr Hajek, Martin Holeňa, and Jan Rauch. The GUHA method and its meaning for data mining. *Journal of Computer and System Science*, pages 34–38, 2010.

[40] Zengyou He, Xiaofei Xu, Zhexue Joshua Huang, and Shengchun Deng. Fp-outlier: Frequent pattern based outlier detection. *Computer Science and Information Systems*, 2(1):103–118, 2005.

[41] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695, 2015.

[42] Geoffrey Holmes, Andrew Donkin, and Ian H Witten. Weka: A machine learning workbench. In *Intelligent Information Systems, 1994. Proceedings of the 1994 Second Australian and New Zealand Conference on*, pages 357–361. IEEE, 1994.

[43] Andreas Holzinger. Interactive machine learning for health informatics: when do we need the human-in-the-loop? *Brain Informatics*, 3(2):119–131, 2016.

[44] Heng Ji, Joel Nothman, Ben Hachey, et al. Overview of TAC-KBP2014 entity discovery and linking tasks. In *Proc. Text Analysis Conference (TAC2014)*, pages 1333–1339, 2014.

[45] Tomas Kliegr. Linked hypernyms: Enriching DBpedia with targeted hypernym discovery. *Web Semantics: Science, Services and Agents on the World Wide Web*, 31:59–69, 2015.

[46] Tomas Kliegr. Quantitative cba: Small and comprehensible association rule classification models. *arXiv preprint arXiv:1711.10166*, 2017.

[47] Tomas Kliegr, Stepan Bahnik, and Johannes Furnkranz. A review of possible effects of cognitive biases on interpretation of rule-based machine learning models. *arXiv preprint arXiv:1804.02969*, 2018.

[48] Tomas Kliegr, J Kuchar, S Vojir, and Vaclav Zeman. Easyminer-short history of research and current development. In *Proceedings of the 17th Conference on Information Technologies-Applications and Theory*, pages 235–239, 2017.

[49] Tomas Kliegr and Jaroslav Kuchar. Benchmark of rule-based classifiers in the news recommendation task. In *International Conference of the Cross-Language Evaluation Forum for European Languages*, pages 130–141. Springer, 2015.

[50] Tomas Kliegr, Jaroslav Kuchar, Davide Sottara, and Stanislav Vojír. Learning business rules with association rule classifiers. In Antonis Bikakis, Paul Fodor, and Dumitru Roman, editors, *8th International Symposium, RuleML 2014, Co-located with the 21st European Conference on Artificial Intelligence, ECAI 2014, Prague, Czech Republic, August 18-20, 2014. Proceedings*, pages 236–250, Cham, 2014. Springer.

[51] Tomas Kliegr, Marek Ovecka, and Jan Zemanek. Topic maps for association rule mining. In *Proceedings of TMRA 2009*. University of Leipzig, 2009.

[52] Tomas Kliegr and Jan Rauch. An XML format for association rule models based on the GUHA method. In *International Workshop on Rules and Rule Markup Languages for the Semantic Web*, pages 273–288. Springer, 2010.

[53] Tomas Kliegr, Stanislav Vojir, and Jan Rauch. Background knowledge and PMML: first considerations. In *Proceedings of the 2011 workshop on Predictive markup language modeling*, pages 54–62. ACM, 2011.

[54] Tomas Kliegr and Ondrej Zamazal. Towards linked hypernyms dataset 2.0: complementing dbpedia with hypernym discovery and statistical type inferrence. In *Proceedings of The Ninth International Conference on Language Resources and Evaluation, LREC*, 2014.

[55] Tomas Kliegr and Ondrej Zamazal. LHD 2.0: A text mining approach to typing entities in knowledge graphs. *Web Semantics: Science, Services and Agents on the World Wide Web*, 39:47–61, 2016.

[56] Tomas Kliegr and Ondrej Zamazal. Antonyms are similar: Towards paradigmatic association approach to rating similarity in SimLex-999 and WordSim-353. *Data & Knowledge Engineering*, 115:174–193, 2018.

[57] Tomas Kliegr, Vaclav Zeman, and Milan Dojchinovski. Linked hypernyms dataset-generation framework and use cases. In *3rd Workshop on Linked Data in Linguistics: Multilingual Knowledge Resources and Natural Language Processing*, page 82, 2014.

[58] Tomaš Kliegr, Vojtěch Svatek, Milan Šimunek, and Martin Ralbovský. Semantic analytical reports: A framework for post-processing of data mining results. *Journal of Intelligent Information Systems*, 37(3):371–395, 2011.

[59] Tomáš Kliegr. *Unsupervised Entity Classification with Wikipedia and WordNet*. Dissertation thesis, University of Economics, Prague, 2012.

[60] Jaroslav Kuchar, Adam Ashenfelter, and Tomas Kliegr. Outlier (anomaly) detection modelling in PMML. RuleML, CEUR-WS, 2017.

[61] Jaroslav Kuchar and Tomas Kliegr. Bag-of-entities text representation for client-side (video) recommender systems. *Proceedings of the RecSysTV,a KDD Workshop on Recommendation Systems for Television and Online Video*, 2014.

[62] Jaroslav Kuchar and Tomas Kliegr. Inbeat: Javascript recommender system supporting sensor input and linked data. *Knowledge-Based Systems*, 135:40–43, 2017.

[63] Jaroslav Kuchař and Tomaš Kliegr. Gain: Web service for user tracking and preference learning - a smart TV use case. In *Proceedings of the 7th ACM Conference on Recommender Systems*, RecSys '13, pages 467–468, New York, NY, USA, 2013. ACM.

[64] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[65] Dekang Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[66] Bing Liu, Wynne Hsu, and Yiming Ma. Integrating classification and association rule mining. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, pages 80–86. AAAI Press, 1998.

[67] Tie-Yan Liu, Yiming Yang, Hao Wan, Hua-Jun Zeng, Zheng Chen, and Wei-Ying Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36–43, June 2005.

[68] Pablo N Mendes, Max Jakob, Andrés Garcia-Silva, and Christian Bizer. DBpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th international conference on semantic systems*, pages 1–8. ACM, 2011.

[69] Ryszard S Michalski. On the quasi-minimal solution of the general covering problem. 1969.

[70] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM, 2007.

[71] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[72] Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. *Transactions of the Association for Computational Linguistics*, 2:231–244, 2014.

[73] Mohamed Morsey, Jens Lehmann, Sören Auer, Claus Stadler, and Sebastian Hellmann. DBpedia and the live extraction of structured data from Wikipedia. *Program*, 46(2):157–181, 2012.

[74] Roberto Navigli. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):10, 2009.

[75] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia, 1998.

[76] Heiko Paulheim and Christian Bizer. Type inference on noisy RDF data. In *The Semantic Web–ISWC 2013*, pages 510–525. Springer, 2013.

[77] Jan Rauch. Considerations on logical calculi for dealing with knowledge in data mining. In *Advances in Data Management*, pages 177–199. Springer, 2009.

[78] Jan Rauch and Milan Simunek. Dealing with background knowledge in the SEWEBAR project. In *Knowledge discovery enhanced with semantic and social information*, pages 89–106. Springer, 2009.

[79] Jan Rauch and Milan Simunek. Applying domain knowledge in association rules mining process–first experience. In *International Symposium on Methodologies for Intelligent Systems*, pages 113–122. Springer, 2011.

[80] Jan Rauch and Milan Simunek. Apriori and guha–comparing two approaches to data mining with association rules. *Intelligent Data Analysis*, 21(4):981–1013, 2017.

[81] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, pages 448–453, 1995.

[82] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144. ACM, 2016.

[83] Giuseppe Rizzo and Raphaël Troncy. NERD: a framework for unifying named entity recognition and disambiguation extraction tools. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 73–76. Association for Computational Linguistics, 2012.

[84] Giuseppe Rizzo, Marieke van Erp, and Raphaël Troncy. Benchmarking the extraction and disambiguation of named entities on the semantic web. In *Proceedings of The Ninth International Conference on Language Resources and Evaluation*, pages 4593–4600. LREC, 2014.

[85] Marko Robnik-Sikonja and Igor Kononenko. Explaining classifications for individual instances. *IEEE Transactions on Knowledge and Data Engineering*, 20(5):589–600, 2008.

[86] Michael Röder, Ricardo Usbeck, and Axel-Cyrille Ngonga Ngomo. Gerbil–benchmarking named entity recognition and linking consistently. *Semantic Web*, pages 1–21, 2017.

[87] Silke Scheible, Sabine Schulte im Walde, and Sylvia Springorum. Uncovering distributional differences between synonyms and antonyms in a word space model. In *Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pages 489–497, 2013.

[88] Andrew D Selbst and Julia Powles. Meaningful information and the right to explanation. *International Data Privacy Law*, 7(4):233–242, 2017.

[89] Wei Shen, Jianyong Wang, and Jiawei Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460, 2015.

[90] Milan Simunek. Academic kdd project lisp-miner. In *Intelligent Systems Design and Applications*, pages 263–272. Springer, 2003.

[91] Radek Skrabal, Milan Simunek, Stanislav Vojir, Andrej Hazucha, Tomas Marek, David Chudan, and Tomas Kliegr. Association rule mining following the web search paradigm. In *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *LNCS*, pages 808–811. Springer Berlin Heidelberg, 2012.

[92] Julius Stecher, Frederik Janssen, and Johannes Furnkranz. Shorter rules are better, aren't they? In *International Conference on Discovery Science*, pages 279–294. Springer, 2016.

[93] Erik F Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics, 2003.

[94] Hannu Toivonen. *Association Rule*, pages 70–71. Springer US, Boston, MA, 2017.

[95] Salvatore Trani, Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, and Raffaele Perego. Sel: A unified algorithm for entity linking and saliency detection. In *Proceedings of the 2016 ACM Symposium on Document Engineering*, pages 85–94. ACM, 2016.

[96] Ricardo Usbeck, Michael Röder, Axel-Cyrille Ngonga Ngomo, Ciro Baron, Andreas Both, Martin Brummer, Diego Ceccarelli, Marco Cornolti, Didier Cherix, Bernd Eickmann, et al. GERBIL: general entity annotator benchmarking framework. In *Proceedings of the 24th International Conference on World Wide Web*, pages 1133–1143, 2015.

[97] Stanislav Vojir, Premysl Vaclav Duben, and Tomas Kliegr. Business rule learning with interactive selection of association rules. *RuleML Challenge*, 2014, 2014.

[98] Stanislav Vojir, Tomas Kliegr, Andrej Hazucha, Radek Skrabal, and Milan Simunek. Transforming association rules to business rules: Easyminer meets drools. *RuleML-2013 Challenge. CEUR-WS. org*, 49, 2013.

[99] Stanislav Vojir, Vaclav Zeman, Jaroslav Kuchar, and Tomas Kliegr. Easyminer.eu: Web framework for interpretable machine learning based on rules and frequent itemsets. *Knowledge-Based Systems*, 150:111–115, 2018.

[100] Paul Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, 1974.

[101] Paul John Werbos. *The roots of backpropagation: from ordered derivatives to neural networks and political forecasting*, volume 1. John Wiley & Sons, 1994.

[102] Caroline Willners. *Antonyms in Context : A Corpus-Based Semantic Analysis of Swedish Descriptive Adjectives*. PhD thesis, Lund University, 2001.

[103] Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

[104] Chenyan Xiong. *Knowledge Based Text Representations for Information Retrieval*. PhD thesis, University of Massachusetts, Amherst, 2016.

[105] Yadollah Yaghoobzadeh, Heike Adel, and Hinrich Schuetze. Corpus-level fine-grained entity typing. *Journal of Artificial Intelligence Research*, 61:835–862, 2018.

[106] Yuanshun Yao, Zhujun Xiao, Bolun Wang, Bimal Viswanath, Haitao Zheng, and Ben Y Zhao. Complexity vs. performance: empirical analysis of machine learning as a service. In *Proc. of the 2017 Internet Measurement Conference*, pages 384–397. ACM, 2017.

[107] Zheng Yuan and Doug Downey. Otyper: A neural architecture for open named entity typing. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018.* AAAI Press, 2018.

Part II.

Entity-based Text Representation

Appendix A: Linked hypernyms: Enriching DBpedia with targeted hypernym discovery

J1 Tomáš Kliegr, Linked hypernyms: Enriching DBpedia with Targeted Hypernym Discovery, Web Semantics: Science, Services and Agents on the World Wide Web, Volume 31, 2015, Pages 59-69, ISSN 1570-8268, https://doi.org/10.1016/j.websem.2014.11.001.

Web Semantics: Science, Services and Agents on the World Wide Web 31 (2015) 59–69

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

Linked hypernyms: Enriching DBpedia with Targeted Hypernym Discovery

Tomáš Kliegr *

Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, nám. W Churchilla 4, 13067, Prague, Czech Republic
Multimedia and Vision Research Group, Queen Mary, University of London, 327 Mile End Road, London E1 4NS, United Kingdom

ARTICLE INFO

Article history:
Received 1 October 2014
Accepted 4 November 2014
Available online 15 November 2014

Keywords:
DBpedia
Hearst patterns
Hypernym
Linked data
YAGO
Wikipedia
Type inference

ABSTRACT

The Linked Hypernyms Dataset (LHD) provides entities described by Dutch, English and German Wikipedia articles with types in the DBpedia namespace. The types are extracted from the first sentences of Wikipedia articles using Hearst pattern matching over part-of-speech annotated text and disambiguated to DBpedia concepts. The dataset covers 1.3 million RDF type triples from English Wikipedia, out of which 1 million RDF type triples were found not to overlap with DBpedia, and 0.4 million with YAGO2s. There are about 770 thousand German and 650 thousand Dutch Wikipedia entities assigned a novel type, which exceeds the number of entities in the localized DBpedia for the respective language. RDF type triples from the German dataset have been incorporated to the German DBpedia. Quality assessment was performed altogether based on 16.500 human ratings and annotations. For the English dataset, the average accuracy is 0.86, for German 0.77 and for Dutch 0.88. The accuracy of raw plain text hypernyms exceeds 0.90 for all languages. The LHD release described and evaluated in this article targets DBpedia 3.8, LHD version for the DBpedia 3.9 containing approximately 4.5 million RDF type triples is also available.

1. Introduction

The Linked Hypernyms Dataset (LHD) provides entities described by Dutch, English and German Wikipedia articles with types taken from the DBpedia namespace. The types are derived from the free-text content of Wikipedia articles, rather than from the semistructured data, infoboxes and article categories, used to populate DBpedia [1] and YAGO [2]. The dataset contains only one type per entity, but the type has stable and predictable granularity. These favorable properties are due to the fact that the types are sourced from the first sentences of Wikipedia articles, which are carefully crafted by the Wikipedia editors to contain the most important information.

To illustrate the LHD generation process, consider the first sentence of the Wikipedia article entitled “Karel Čapek”: *Karel Čapek (...) was a Czech writer of the early* 20th *century best known for his science fiction, including his novel War with the Newts and the play R.U.R. that introduced the word robot.* This text is first processed with a part of speech (POS) tagger. Consequently, using a *JAPE grammar*, a regular expressions language referencing the underlying text as well as the assigned POS tags, the hypernym “writer” is extracted. This hypernym is then disambiguated to a DBpedia Ontology class `dbo:Writer`. The resulting entry in LHD is the RDF type triple[1]:

```
dbp:Karel_Čapek rdf:type dbo:Writer .
```

The LHD dataset was subject to extensive evaluation, which confirms the following hypotheses:

- high quality types for DBpedia entities can be extracted from the first sentences of Wikipedia articles,
- resulting set of types provides a substantial complement to types obtained by the analysis of Wikipedia infoboxes and categories.

This dataset can thus be used to “fill the gaps” in DBpedia and YAGO, the two largest semantic knowledge bases derived

* Correspondence to: Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, nám. W Churchilla 4, 13067, Prague, Czech Republic.
E-mail address: tomas.kliegr@vse.cz.

[1] `dbo:` standing for http://dbpedia.org/ontology/ and `dbp:` for http://dbpedia.org/resource/.

http://dx.doi.org/10.1016/j.websem.2014.11.001
1570-8268/© 2014 The Author. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/3.0/).

from the semistructured information in Wikipedia. To illustrate the individual types of complementarity, consider the following examples.

- LHD can provide a more *specific* type than DBpedia or YAGO. This is typically the case for less prolific entities, for which the semistructured information in Wikipedia is limited. The most specific type provided by DBpedia or YAGO for the "HMS Prince Albert (1984)" entity is `dbo:Ship`, while LHD assigns the type `dbp:Warship` (as a subclass of `dbo:Ship`).
- LHD can provide a more *precise* type. An asteroid named "1840 Hus" is assigned type `dbo:Asteroid` in LHD, while DBpedia assigns it the imprecise type `dbo:Planet` (asteroid is not a subclass of planet).
- LHD is in some cases the only knowledge base providing *any type information*. For example, for asteroid "1994 Shane", neither DBpedia nor YAGO provide a type, while LHD does.
- LHD helps to choose the primary most specific type for an entity. DBpedia assigns Karel Čapek, a famous Czech writer, `dbo:Person` as the most specific DBpedia type, YAGO assigns `yago:CzechScience FictionWriters`, but also several other less commonly sought for types, such as `yago: PeopleFromTrutnov District`. Following the choice of Wikipedia editors for the first article's sentence, LHD assigns a single type: `dbo:Writer`. This can help to identify `yago:CzechScienceFictionWriters` as the primary most specific type for Karel Čapek (as opposed to `yago: PeopleFromTrutnovDistrict`).

The last bullet point shows that even if the LHD provided type is less specific than the type provided in YAGO or DBpedia, it may not be completely redundant. The LHD dataset for German, English and Dutch is provided under a free license. Additionally, this paper along with the complementary resources[2] describes the LHD design process in detail sufficient to allow for generation of the dataset also for other language versions of Wikipedia.

This paper is organized as follows. Section 2 gives a survey of related work. Section 3 describes the text-mining algorithm, Section 4 the procedure for disambiguating the hypernyms with a DBpedia URI and the resulting datasets. Section 5 describes the alignment of the linked hypernyms with DBpedia and YAGO2s ontologies. Human evaluation of accuracy is presented in Section 7. The following two sections discuss LHD extensions. Section 8 presents LHD 2.0 draft, which uses statistical type inference to increase the number of types mapped to the DBpedia Ontology. Steps required to extend LHD to other languages are covered in Section 9. The conclusion in Section 10 summarizes the key statistics, gives dataset license, availability and discusses possible applications, including a named entity recognition system based on the Linked Hypernyms Dataset.

2. Related work

The use of methods from computational linguistics on extraction of machine-readable knowledge from electronic dictionary-like resources has long been studied (cf. Wilks et al. [3]) with research specifically on extraction of hyponymy–hypernymy relation from lexical resources using patterns dating back to at least 1984 [4]. The hypernym discovery approach proposed here is based on the application of a special type of hand-crafted lexico-syntactic patterns often referred to as Hearst patterns [5]. The prototypical Hearst pattern goes along the sentence frame H0:

"An L_0 is a (kind of) L_1" (H0).

Hearst patterns were so far used primarily on large text corpora with the intent to discover all word-hypernym pairs in a collection. The extracted pairs can serve e.g. for taxonomy induction [6,7] or ontology learning [8]. This effort was undermined by the relatively poor performance of syntactic patterns in the task of extracting *all* candidate hypernym/hyponym word pairs from a *generic* corpus. The recall–precision graph for the seminal hypernym classifier introduced by [6] indicates precision 0.85 at recall 0.10 and precision 0.25 at recall of 0.30.

Utilization of hypernyms discovered from textual content of *Wikipedia articles* was investigated in a number of works. Strube and Ponzetto [9] built a large scale taxonomy from relation candidates extracted from English Wikipedia categories. One of the sources of evidence for a relation being classified as a *subsumption* or not is obtained by applying Hearst patterns (and corresponding anti-patterns) on Wikipedia and the Tipster corpus. The result of the classification was determined based on whether a majority of the matches are accounted for the patterns or the anti-patterns. Detection of hypernyms in the free text of Wikipedia articles was used as one of the methods to classify relation candidates extracted from the categories and as such had only a marginal influence on the overall results (0.04 precision improvement).

To the best of my knowledge, [10] were first to implement a system that extracts a hypernym for the Wikipedia *article subject* with high precision from *the first sentence* of the article text with the help of Part of Speech (POS) tagger. The discovered hypernyms were used as features in a Conditional-Random-Fields-based named entity tagger yielding again only a moderate improvement in accuracy.

HypernymFinder [11] is an algorithm that searches a hypernym for a specific noun phrase. It identifies a number of candidates by searching for occurrences of Hearst patterns featuring the query hyponym and then uses the frequency of the matches to determine the best hypernyms. The Hearst patterns were matched against a large 117 million web page corpus. The authors record an improvement over the results reported earlier by [6] for lexicosyntactic patterns with baseline precision at 0.90 and recall at 0.11.

The 2007 paper [10] laid foundations to the use of Hearst patterns over Wikipedia that is called *Targeted Hypernym Discovery* task (THD) in this paper. To get hypernym for a particular entity, THD applies Hearst patterns on a document describing the entity. In earlier work using English Wikipedia, we obtained accuracy of 87% when extracting hypernyms from articles describing named entities [12]. To the extent of my knowledge, this 2008 paper presented the first evaluation of the quality of hypernyms discovered from Wikipedia. Similar results for extracting hypernyms from articles describing people in German Wikipedia were later reported by [13] (also refer to Section 7).

Contrasted to HypernymFinder, which uses a set of randomly selected noun phrases as query hyponyms, the set of query hyponyms in THD is limited to Wikipedia article titles. With this constraint, the first Hearst pattern match in the first sentence of the respective article yields hypernyms with higher precision and substantially higher recall of 0.94 and 0.88 respectively for English Wikipedia (cf. Section 7.1). Note that the results for THD, HypernymFinder [11], and the algorithm of Snow et al. [6] cannot be directly mutually compared, since the latter evaluates precision and recall over candidate hypernym/hyponym word pairs (the input is a large corpus), while HypernymFinder is concerned with whether or not a good hypernym for a given noun phrase can be retrieved (the input is again a large corpus), and eventually THD evaluates whether a good hypernym for Wikipedia article subject can be retrieved (the input is that article's first sentence).

Tipalo [14] is the most closely related system to the workflow used to generate LHD. Similarly to approach presented in this paper, Tipalo covers the complete process of generating types for

[2] http://ner.vse.cz/datasets/linkedhypernyms.

Table 1
Tipalo output for the "Kanai Anzen" entity. Retrieved using on-line service at http://wit.istc.cnr.it/stlab-tools/tipalo/ on 23/09/14.

Subject	Predicate	Object
dbpedia:Kanai_Anzen	rdf:type	domain:Omamorous
dbpedia:Kanai_Anzen	rdf:type	domain:Religion
dbpedia:Kanai_Anzen	rdf:type	domain:JapaneseAmulet
domain:JapaneseAmulet	rdfs:subClassOf	domain:Amulet
dbpedia:Amulet	owl:equivalentClass	dbpedia:Amulet

DBpedia entities from the free text of Wikipedia articles. However, while LHD generation process uses THD to extract the hypernym directly from the POS-tagged first sentence, the extraction process in Tipalo is more complex. The algorithm starts with identifying the first sentence in the abstract which contains the definition of the entity. In case a coreference is detected, a concatenation of two sentences from the article abstract is returned. The resulting natural language fragment is deep parsed for entity definitions using the FRED tool [15] for ontology learning. FRED uses methods based on frame semantics for deriving RDF and OWL representations of natural language sentences.

The result of analyzing the entity definition is maximum one type for THD, while Tipalo may output multiple types. If there are multiple candidate hypernyms in the definition, Tipalo uses all of them. Also, if a hypernym is composed of a multi-word noun phrase Tipalo outputs multiple types formed by gradually stripping the modifiers (cf. example below).

To illustrate the differences, consider the Wikipedia page "Kanai Anzen". Using the first sentence of the Wikipedia entry: *Kanai Anzen is a type of omamori, or Japanese amulet of the Shinto religion.*, THD outputs just the head noun of the first candidate hypernym[3] ("omamori"). Tipalo result for this Wikipedia page is presented in Table 1. Tipalo outputs four types ("JapaneseAmulet", "Amulet", "Religion" and "Omamorous"). Similarly to steps subsequent to the THD execution, Tipalo detects whether the entity is a class or instance and correspondingly selects the relation (`rdfs:subClassOf` or `rdf:type`) with which the entity will be linked to the assigned types. Another interesting aspect common to both systems is their use of DBpedia resources as classes.

In this specific example, the results of both tools are comparable and somewhat complementary: LHD provides a more precise DBpedia mapping (omamori is a type of Japanese amulet), while Tipalo output contains supplemental taxonomic information (JapaneseAmulet as a subclass of Amulet). While in LHD all types are represented with DBpedia concepts, Tipalo also outputs concepts in the FRED namespace.[4]

Tipalo uses a context-based disambiguation algorithm which links the concepts to WordNet synsets. Consequently, OntoWordnet 2012, an OWL version of WordNet, is used to align the synsets with types from the Dolce Ultra Lite Plus[5] (DULplus) and the Dolce Zero (D0)[6] ontologies. The latter being an ontology defined by the authors which generalizes a number of DULplus classes in OntoWordnet. In contrast, LHD aims at providing types suitable for DBpedia and YAGO enrichment. To this end, the types assigned to entities are from the DBpedia namespace, preferably DBpedia Ontology classes.

To illustrate the differences in ontology mapping results, consider the types returned for "Lupercal" (an example listed on the Tipalo homepage). Tipalo assigns type `dbp:Cave`, which is mapped via the `owl:equivalentClass` to `wn30:synset-cave-noun-1` and is marked as a subclass of `d0:Location`.[7] In contrast, LHD assigns this entity with `dbo:Cave`, a class from the DBpedia ontology.

As could be seen, there are multiple dissimilarities between the LHD generation process and Tipalo both on the algorithmic and conceptual level. The scale of the resources is also different. Tipalo is demonstrated with a proof of concept ontology constructed from analyzing 800 randomly selected English Wikipedia pages and evaluated on 100 articles. However, its online demo service is able to process any Wikipedia article. LHD was generated for three complete Wikipedia languages and is supplemented by evaluation performed on two orders of magnitude larger scale. A limited comparison with Tipalo in terms of hypernym extraction results is covered in Section 7.

The Linked Hypernyms Dataset described in this paper is a comprehensive attempt to extract types for DBpedia entities from the free text of Wikipedia articles. The dataset is generated using adaptations of previously published algorithms, approaches and systems: Hearst patterns are used to extract hypernyms from the plain text, Wikipedia search for disambiguation, and string-based ontology matching techniques for alignment with DBpedia and YAGO ontologies.

By providing results not only for the English Wikipedia, but also for the entire Dutch and German Wikipedias, it is demonstrated that the presented approach can effectively be extended to other languages. The retrieval of new types for entities from the free-text can provide a complementary information to other recent DBpedia enrichment efforts [16,17], which derive new types either from data already in the Linked Open Data (LOD) cloud (as in [16]), or from the semistructured information (cross-language links in [17]).

3. Targeted hypernym discovery

The Targeted Hypernym Discovery implementation used to perform the linguistic analysis for Linked Hypernyms Dataset is an extended and reworked version of the algorithm presented in [12]. The precision and recall of the grammars was improved. Also, the workflow was changed to support multilingual setting and grammars for German and Dutch were added. The largest conceptual deviation from the original algorithm as well as from the prototypical H0 pattern is that the occurrence of the subject ($L0$) is not checked. According to empirical observation this change increases recall with negligible effect on precision.[8]

The schematic grammar used is

"* is a (kind of) L_1" (H1).

where * denotes a (possibly empty) sequence of any tokens. This modification increased recall. Restricting the extraction to the first match in the article's first sentence helped to improve the precision. The grammars were manually developed using a set of 600 randomly selected articles per language.

The main features of the THD implementation used to generate the presented datasets include:

[3] As discussed in Section 3 this is the most reliable choice according to empirical observation.

[4] `http://www.ontologydesignpatterns.org/ont/fred/domain.owl#`.

[5] http://www.ontologydesignpatterns.org/ont/wn/dulplus.owl.

[6] `http://www.ontologydesignpatterns.org/ont/d0.owl`.

[7] `wn30syn:` `http://purl.org/vocabularies/princeton/wn30/instances/`.

[8] Validating whether the subject of the article's first sentence matches the article title is an unnecessary check, which sometimes causes false negative matches due to differences between the first sentence's subject and the article title. For example, the article entitled "ERAP" starts with: *Entreprise de recherches et d'activités pétrolières is a French petroleum company*.... Checking the occurrence of "ERAP" in the first sentence would result in no match.

- **Only the first sentence of the article is processed**. More text (first paragraph, section) introduces noise according to empirical observation.
- **Only the first hypernym is extracted**.

> *Example.*
> Consider sentence: *Évelyne Lever is a contemporary French historian and writer.* The result of THD is one hypernym *historian*, the word *writer* is ignored. German articles are more likely to contain multiple hypernyms in the first sentence, while this is less common for English and Dutch.

- **Some Wikipedia article types are excluded**. Programmatically identifiable articles that do not describe a single entity are omitted. This applies to lists, disambiguation articles and redirects.
- **For multi-word hypernyms, the result is the last noun.**

> *Example.*
> Consider sentence: *Bukit Timah Railway Station was a railway station.* The THD result is "station", rather than "railway station". Extracting the complete multi-word sequence would yield a more specific hypernym in many cases, but a straightforward implementation would also negatively impact precision.

Multi-word hypernyms were left for future work.

- **Hypernym contained in the entity name or article title is ignored.**

> *Example.*
> While for a human it may be obvious that if something is named "Bukit Timah Railway Station" then it is a (railway) station, it follows from the nature of Hearst patterns that the hypernym in the entity name is ignored. Likewise, hypernyms contained in article title such as the word "novel" in "Hollywood (Vidal novel)" are ignored.

- **Common generic hypernyms that precede a more specific hypernym are skipped.**

> *Example.*
> Consider again the sentence: *Kanai Anzen is a type of omamori, or Japanese amulet of the Shinto religion.* THD skips the word "type" and returns the word "omamori". The list of these generic hypernyms is specified in the grammar for each language, and includes for example the "name of" expression, but also already relatively specific hypernyms such as species ("species of").

- **The result of THD is lemmatized**. In languages where hypernyms often appear in inflected forms lemmatization ensures that a base form is used as the hypernym.[9]

> *Example.*
> Consider sentence: *Die York University ist eine von drei Universitäten in Toronto.* With the first hypernym being Universitäten, the result of lemmatization is Universität, which is used as the plain text hypernym for this entry.

The set of Wikipedia article–hypernym pairs output by THD is referred to as the *"Plain Text" Hypernyms Dataset*.

4. Hypernym linking

The limitation of THD is that its output is a plain string, which is unusable in the Linked Data environment. As a first attempt to address the problem, the "most frequent sense" disambiguation is used.

This approach is based on a simple, yet according to experimental results [18], effective way of discovering links to DBpedia—the Wikipedia Search API.[10] Since there is an unanimous mapping between Wikipedia articles and DBpedia resources, the linking algorithm first searches for an article describing the hypernym in Wikipedia and then the URL of the first article hit is transformed to a DBpedia URI.

In the TAC English Entity Linking task [18], this approach had a close median performance among the 110 submissions with B^{3+} F1 measure on 2190 queries of 0.54–0.56 (depending on whether live Wikipedia or a Wikipedia mirror was used). The best system achieved B^{3+} F1 result of up to 0.75, the average B^{3+} F1 result was 0.56. Compared to other solutions, using Wikipedia search for disambiguation in the LHD generation process has several advantages. Wikipedia search is readily available for all Wikipedia languages, is fast, and implies no dependency on a third-party component.

4.1. Disambiguation

Wikipedia Search API uses a PageRank-like algorithm for determining the importance of the article in addition to the textual match with the query. Since the hypernyms tend to be general words with dominant most frequent sense, the most frequent sense assumption works well as experimentally demonstrated in Section 7.2.

It should be noted that the following possibility was investigated: using the hyperlinks that are sometimes placed on the hypernym in the source article. However, only a small fraction of articles contains such links, furthermore, the quality of these links seems to be lower than what can be obtained by the search-based mapping. Linked hypernyms are the output of the disambiguation process.

4.2. Data cleansing

The first step, applicable only to non-English DBpedia, is to use the DBpedia's interlanguage links to replace the linked hypernyms with their English counterparts.

The main cleansing step amounts to performing replacements and deletions according to manually specified rules. These rules were identified by manually checking several hundreds of the most frequent types assigned by THD.

Mapping rules are used to replace a particular linked hypernym. Mapping rules were introduced to tackle two types of problems:

- For some types the hypernym discovery makes systematic errors, typically due to POS tagger error or deficiency in the THD grammar.

[9] During LHD dataset generation, the lemma was used instead of the underlying string if it was made available by the tagger for the given language.

[10] http://www.mediawiki.org/wiki/Extension:Lucene-search.

Table 2
Hypernyms and Linked Hypernyms Datasets—statistics and comparison with DBpedia and YAGO2s. The largest dataset for each language is listed in bold. The Wikipedia snapshots used to generate the datasets: December 1st, 2012 (German), October 11th, 2012 (Dutch), September 18th, 2012 (English).

Statistic	Dutch	English	German
Linked Hypernyms Dataset			
Wikipedia articles	1691k	5610k	2942k
–without redirect articles (is_page_redirect = 1 database field)	1505k	3299k	2252k
–without lists, images, etc. (identified from article name)	1422k	2590k	1930k
"Plain text" Hypernyms dataset	889k	1553k	937k
linked hypernyms (before data cleansing)	670k	1393k	836k
Linked Hypernyms Dataset—instances	**664k**	1305k	**825k**
Linked Hypernyms Dataset—classes	1k	4k	3k
Other datasets			
DBpedia 3.8—instances with type (instance_types_{lang}.nt)	11k	2351k	449k
YAGO2s—instances with type (yagoTypes.ttl)		**2886k**	

Example. A mapping rule tackling such issue is "`dbp:Roman` → `dbp:School`". The word "Roman" is an adjective that should never be marked as a hypernym. The reason is that the POS tagger incorrectly marks "Roman" as a noun if it appears in collocation "Roman catholic school" resulting in the THD grammar yielding "Roman" instead of "School". Since "Roman" is not output by THD virtually in any other case, the existence of the mapping rule increases recall without negatively impacting precision.
Based on this mapping rule, the following statement

```
dbp:Father_Hendricks rdf:type dbp:Roman .
```

is replaced by

```
dbp:Father_Hendricks rdf:type dbp:School .
```

- For some hypernyms, the hypernym linking algorithm produces an incorrect disambiguation.

Example. The `dbp:Body` carries the "physical body of an individual" meaning, while it appears almost exclusively in the "group of people" sense. This is corrected by mapping rule: "`dbp:Body` → `dbp:Organisation`".
Based on this mapping rule, the following statement

```
dbp:National_Executive_Committee rdf:type dbp:Body.
```

is replaced by

```
dbp:National_Executive_Committee rdf:type
dbp:Organization .
```

Deletion rules were introduced to remove all entities with a "black-listed" hypernym. Again, there were two reasons to blacklist a hypernym:

- The linked hypernym is too ambiguous with little information value. Example: `dbp:Utility` or `dbp:Family`.
- The linked hypernym cannot be disambiguated to a single concept that would hold for the majority of its instances.

Example.
Consider `dbp:Agent`, which either denotes an organization or a chemical compound. Since none of the senses is strongly dominating, a deletion rule for statements with this concept as a hypernym was introduced.
Based on this mapping rule, the following statements were deleted (among others):

```
dbp:Metoclopramide rdf:type dbp:Agent.
dbp:US_Airline_Pilots_Association rdf:type dbp:Agent.
```

In the current release, these rules have global validity, i.e. it is not possible to specify a context in which they apply.

The resulting *Linked Hypernyms Dataset* is published using the N-Triples notation [19]. The *"Plain text" Hypernyms Dataset* is made available in one article–hypernym tuple per line format. A separate file is downloadable for each language. The number of records in the Linked Hypernyms Dataset is about 10%–20% (depending on the language—ref. to Table 2) smaller than for the "Plain text" Hypernyms Dataset, which is in part caused by the application of the deletion rules.

5. DBpedia and YAGO alignment

The results of hypernym linking, described in the previous section, are DBpedia URIs that are not well connected to the LOD cloud. The linked hypernyms are URIs from the (http://dbpedia.org/resource/) namespace (`dbp:` prefix), which is used in DBpedia to identify entities. Each DBpedia resource can be mapped to a Wikipedia article using the following naming scheme:

`http://dbpedia.org/resource/`*`Name`* corresponds to `http://en.wikipedia.org/wiki/`*`Name`* (similarly for other languages). While there are other knowledge bases that use entities from the `dbp:` namespace as types (cf. Tipalo in Section 2), it is preferred to use as types concepts from the DBpedia Ontology. These concepts reside in the http://dbpedia.org/ontology/ namespace (`dbo:` prefix).

This section describes the alignment of the LHD types from the `dbp:` namespace to the DBpedia ontology (version 3.8 containing 359 classes). This ontology is particularly suitable for two reasons: it facilitates the use of the Linked Hypernyms Dataset for DBpedia enrichment, and the fact that many concepts in the ontology have names of one or a few word length simplifies the alignment process, since the THD generated linked-hypernyms are concepts with a short name consisting mostly of one word. For DBpedia ontology alignment, a conservative string-based approach is adopted, which requires complete match with the class name. Complementary set of mappings was generated using a substring match with a follow-up manual verification.

In the second step alignment with the version 2s of the YAGO ontology [2] was performed. YAGO2s does not only contain complementary facts to DBpedia, but with 450.000 concepts in the taxonomy it provides much wider possibilities for matching with the linked hypernyms than the DBpedia Ontology. Again, a simple string-based ontology alignment algorithm was used. The substantially higher number of classes in YAGO resulted in a higher number of mappings. For this reason, the manual verification of the approximate mappings was not performed. It should be noted that this has no effect on the quality of the dataset, since the YAGO mapping was performed only to identify the RDF type triples which are novel w.r.t. to DBPedia *and* YAGO and to gather the corresponding statistics. Types from the YAGO ontology are not used in LHD.

5.1. Alignment with the DBpedia ontology

The alignment with DBpedia is performed using the "exact match" algorithm in order to ensure the highest reliability. For each RDF type triple in LHD, the algorithm tries to find a DBpedia Ontology class for the object (the hypernym) based on a complete textual match. If such a match is successful, the object of the statement is replaced by the DBpedia Ontology class.

Example.
The output of the disambiguation phase is the following statement:

```
dbp:Karel_Čapek rdf:type dbp:Writer .
```

Since for "Writer" there is a class in DBpedia Ontology, this statement is replaced with:

```
dbp:Karel_Čapek rdf:type dbo:Writer .
```

The new statement is better interconnected in the LOD cloud.

If no concept with a fully matching name[11] is found, an approximate match is attempted in order to improve the interconnectedness.

Approximate matching returns the DBpedia Ontology concept which ends with the linked hypernym as substring. In case of multiple matches, the one with longest match is selected. Arbitrary selection is made in case of a tie. The result of this process is a set of candidate subclass relations between linked hypernyms and the DBpedia ontology concepts. Since there are only 359 classes in the DBpedia 3.8 ontology, there were 600 mapping candidates for English,[12] it was possible to perform manual verification. Based on the result, the type was either marked as confirmed, a mapping/deletion rule was created, or no action taken indicating that the mapping is incorrect. After the manual processing of the results, the algorithm was re-executed excluding the confirmed mappings.

Example.
Some of the mappings reviewed included:
1) '`dbp:Township` → `dbo:Ship`',
2) '`dbp:Warship` → `dbo:Ship`',
3) '`dbp:Planets` → `dbo:Planet`',
4) '`dbp:Bicyclist` → `dbo:Cyclist`'.
Except for the first mapping, all were confirmed.

It should be emphasized that all mappings identified based on approximate matching are serialized as extra RDF type triples, preserving the original statements.

Example.
For the "HMS Prince Albert (1984)" entity mentioned earlier, LHD contains both the original specific type, a DBpedia resource, and a universal mapping of this type to its superclass in the DBpedia Ontology:

```
dbp:HMS_Prince_Albert_(1864) rdf:type dbp:Warship
dbp:Warship rdfs:subClassOf dbo:Ship
```

The results of this phase are:

- replacements in LHD in case of an exact match,
- mapping file for confirmed approximate matches,
- mapping file with unconfirmed approximate matches.

5.2. Alignment with the YAGO ontology

While the primary goal of the DBpedia Ontology alignment is to use the better connected concepts from the DBpedia Ontology namespace instead of DBpedia resources as linked hypernyms, the purpose of YAGO alignment is to detect facts (RDF type triples) in the Linked Hypernyms Dataset that are confirmed by YAGO2s.

Overlap with YAGO2s[13] was checked only for a portion of entity-hypernym tuples with high confidence, which passed the novelty check against DBpedia. These are three partitions commonly denoted in Table 3 as *DBpedia Enrichment Dataset*. Each entity in the dataset was assigned to one of the four categories (listed in the order of priority):

- **YAGO No Type**, entity is not assigned any YAGO2s type,
- **YAGO Exact**, a perfect match between the linked hypernym and YAGO2s type assigned to the entity was found,
- **YAGO Approximate**, a YAGO2s type assigned to the entity containing the linked hypernym as a substring was found,
- **YAGO No Match**, none of the above applies.

To perform the comparison, a transitive closure of YAGO2s ontology types was used. The number of RDF type triples falling into the individual partitions is reported in Table 4.

Example.
Consider statement:

```
dbp:H._R._Cox rdf:type dbp:Bacteriologist .
```

The DBpedia Ontology 3.8 does not contain a class for bacteriologist, which places this statement (after other preconditions discussed in section 6.5 have been tested) to the *DBpedia Enrichment Dataset* partition *Not mapped/New*. YAGO assigns this entity multiple classes,[a] but none of these or their superclasses have "bacteriologist" as a substring. This places the statement into the *YAGO No Match* partition of *Not mapped/New* in Table 4.

[a] `wikicategory_American_microbiologists, wikicategory_Indiana_State_University_alumni`

6. Partitions of the dataset

LHD is divided into several partitions according to the ontology alignment results and redundancy of RDF type triples with respect to DBpedia 3.8 Ontology and the DBpedia 3.8 instance file, which contains statements assigning DBpedia instances to DBpedia Ontology classes. The individual partitions are described in the remainder of this section. Table 3 gives the essential statistics on each partition.

6.1. Mapped/classes

This partition contains statements, where the entity (the subject) is found to be used as a hypernym (object) in another LHD statement. The entity does not have any DBpedia Ontology type assigned in the DBpedia instance file.

Example.

```
dbp:Llama rdfs:subClassOf dbp:Camelid .
```

It should be noted that compared to partition "Notmapped /Spurious Entity" (Section 6.7), there is no contradicting evidence for `dbp:Llama` to be a class. As a result, this partition uses the `rdfs:subClassOf` relation.

[11] The stemmed substring after the last "/" in the URI, and `rdfs:label` are considered as concept name.

[12] It follows from the type of the matching algorithm employed that the space of mapping candidates is restricted to linked hypernyms that have one of the classes from the DBpedia Ontology as a substring (excluding exact match).

[13] The latest release as of submission.

Table 3
LHD subdatasets.

Dataset	Mapped classes	Mapped existing	Notmapped probable overlap	Mapped new—no overlap	Notmapped new	Mapped new—no type	Notmapped spurious entity	Notmapped spurious hypernym
				DBpedia enrichment dataset				
Relation	Subclass	Type	type	type	type	Type	Type	Type
Entries (EN)	4043	217,416	5330	126,032	736,293	198,040	1149	20,850
Entries (DE)	2854	50,539	622	58,765	586,419	125,013	59	3,692
Entries (NL)	1304	15,392	235	16,884	563,485	67,990	0	57
Accuracy (EN)				0.82	0.83	0.94		

Table 4
Partitions of the DBpedia Enrichment Dataset (English) according to overlap with YAGO2s. The accuracy of plain text hypernyms is marked with †, the accuracy of linked hypernyms with ‡.

Partition according to DBpedia alignment result	Subpartitions according to YAGO Ontology alignment result									
	No type		Exact		No match		Approx.		All	
	size	acc	size	acc	size	acc	size	acc	size	acc
Mapped/New—No Overlap	9,699	0.98† 0.91‡	59,365	1.00† 0.99‡	35,775	0.95† 0.90‡	21,193	NA	126,032	0.97† 0.82‡
Not mapped/New	150,333	0.89† 0.81‡	199,916	1.00† 0.86‡	295,217	0.93† 0.77‡	90,827	NA	736,293	0.93† 0.83‡
Mapped/New—No Type	38,258	0.95† 0.87‡	74,503	1.00† 0.95‡	72,745	0.98† 0.94‡	12,534	NA	198,040	0.97† 0.94‡
all	198,290	0.91† 0.83‡	333,784	1.00† 0.90‡	403,737	0.95† 0.90‡	NA	NA	1,060,365	0.94† 0.85‡

Most, but not all, of the statements have type from the dbp namespace.

6.2. *Mapped/existing*

This partition contains statements, where the entity was not found to be used as a hypernym in another LHD statement. The entity does have a DBpedia Ontology type assigned in the DBpedia 3.8 instance file. The type assigned by LHD was successfully mapped to a DBpedia Ontology class. Consequently, it was found out that the same statement already exists in the DBpedia instance file.

Example.

`dbp:Czech_Republic rdf:type dbo:Country .`

Identical statement to the above LHD triple is already contained in the DBpedia instance file.

6.3. *Notmapped/probable overlap*

This partition contains statements, where the entity was not found to be used as hypernym in another LHD statement. The entity does have a DBpedia Ontology type assigned in DBpedia instance file. The type assigned by LHD was *not* mapped to a DBpedia Ontology class, however, it was found out that a similar statement already exists in the DBpedia instance file.

Example.

`dbp:Boston_Cyberarts_Festival rdf:type dbp:Festival .`

The DBpedia 3.8 ontology does not contain a class that would have "festival" as a substring, therefore the mapping failed and the type is represented with a DBpedia resource. However, the instance `dbp:Boston_Cyberarts_Festival` is assigned type `schema.org/Festival` in the DBpedia 3.8 instance file. Since there is a textual match between concept names of the LHD and Schema.org types, this triple is classified as a probable overlap.

All statements have type from the dbp namespace.

6.4. *Mapped/new—no overlap*

This partition contains statements, where the entity was not found to be used as hypernym in another LHD statement. The type assigned by LHD was mapped to a DBpedia Ontology class, however, it was found out that while the DBpedia 3.8 instance file assigns at least one DBpedia Ontology type to this entity, none of the assigned types matches the LHD type.

Example.

`dbp:Karel_Čapek rdf:type dbo:Writer .`

The `dbp:Karel_Čapek` entity has already multiple types in the DBpedia 3.8 instance file, with the most specific type being `dbo:Person`. The type assigned by LHD is new with respect to this list.

It should be noted that this partition contains also statements, whose type can be mapped to the DBpedia Ontology via the approximate mappings (cf. Section 5.1).

Example.

`dbp:HMS_Prince_Albert_(1864) rdf:type dbp:Warship .`

About 89% of the statements in the English dataset have type from the dbo namespace and the rest from the dbp namespace (these are mapped via the approximate mappings).

6.5. *Not mapped/new*

This partition contains statements, where the entity was not found to be used as hypernym in another LHD statement. The type assigned by LHD was not mapped to a DBpedia Ontology class.

Example.

`dbp:H._R._Cox rdf:type dbp:Bacteriologist .`

This partition contains typically statements with a specific type that is not covered by the DBpedia Ontology. All statements have type from the dbp namespace.

6.6. Mapped/new—no type

The entity was not found to be used as hypernym in another LHD statement. The type assigned by LHD is mapped to a DBpedia Ontology class. The entity is not assigned any DBpedia Ontology type in the DBpedia 3.8 instance file. As a consequence, the type assigned by LHD must be new.

Example.

`dbp:Vostok_programme rdf:type dbo:Project .`

The `dbp:Vostok_programme` entity does not have any entry in the DBpedia 3.8 instance file.

About 93% of the statements in the English dataset have type from the dbo namespace and the rest from the dbp namespace (these are mapped via the approximate mappings).

6.7. Notmapped/spurious entity

This partition contains statements, where the entity (the subject) is found to be used as a hypernym (object) in another LHD statement and at the same time the entity has a DBpedia Ontology type assigned in the DBpedia 3.8 instance file.

Example.

`dbp:Coffee rdf:type dbp:Beverage .`

The subject is used as a hypernym (class) because it is used in LHD statements such as:

`dbp:Organic_coffee rdf:type dbp:Coffee .`

At the same time DBpedia contains statements that use `dbp:Coffee` as an instance:

`dbp:Coffee rdf:type dbo:Food .`

This contradicting evidence places the statement into the spurious category.

While using the same concept both as instance and class is possible through the OWL 2 *punning* construct, the purpose of this and the following LHD partitions is to isolate such possibly dubious statements for further validation.

6.8. Notmapped/spurious hypernym

The hypernym is used as an instance in a statement in the DBpedia 3.8 instance file.

Example.

`dbp:Aspartate_transaminase rdf:type dbp:Phosphate .`

The `dbp:Phosphate` concept is already assigned a type in the DBpedia instance file:

`dbp:Phosphate rdf:type dbp:ChemicalCompound .`

The fact that `dbp:Phosphate` is used as an instance in DBpedia renders suspicious the extracted LHD statements, which use it as a class.

7. Evaluation

This section presents experimental results that demonstrate the coverage as well as the quality of the datasets. Evaluation of the hypernym discovery algorithm is covered in Section 7.1 and of the disambiguation algorithm in Section 7.2. The assessment of the final Linked Hypernyms Dataset is reported in three subsections. Section 7.3 introduces the evaluation methodology and presents the overall accuracy. Accuracy of the entity-linked hypernym pairs novel w.r.t. existing knowledge bases is examined in Section 7.4 and the accuracy of the rediscovered (redundant) pairs in Section 7.5.

7.1. Hypernym discovery

The quality of the hypernym discovery was evaluated on three manually tagged corpora (English, German, Dutch) with the GATE framework (http://gate.ac.uk).

Using the random article functionality from the Wikipedia search API, 500 articles for each language were selected. Corpus containing the articles' first sentences was created for each of the languages. The first sentences were extracted automatically using a customized GATE Regex Sentence Splitter plugin with negligible error. Lists, disambiguation articles and redirects were skipped along with empty articles or articles with failed first sentence extraction.

For the English corpus, the first appearance of a hypernym in each of the documents was independently annotated by three annotators with the help of the Google Translate service. The annotators were students with good command of English, elementary German and no knowledge of Dutch. The groundtruth was established by the consensus of two annotators. For German and Dutch, all documents were annotated by two annotators, when there was no consensus, an annotation by the third annotator was provided. To compare with previous work [13], a focused dataset consisting of documents describing people was manually created from the German dataset. It should be noted that the documents used for evaluation were unseen during the grammar development phase.

The GATE Corpus Quality Assurance tool was used to compute precision and recall of the computer generated annotations with human ground-truth. The results are summarized in Table 5. For computing the metrics, partially correct (overlapping) annotations were considered as incorrect. It can be seen that the results are quite consistent, with precision exceeding 0.90 for all languages. The best results were obtained for the German person subset, with precision 0.98 and recall 0.95. This is on par with the 0.97 precision and 0.94 recall reported for lexico-syntactic patterns and the Syntactic–Semantic Tagger respectively, the best performing algorithms in [13]. A statistic significance test was not performed due to differences in annotation methodology: while [13] annotated all hypernyms in the input text, in experiments presented here only the first specific hypernym was annotated.[14] The results are almost identical to those obtained by the Tipalo algorithm [14] for the type selection subtask. This evaluation was performed on 100 English Wikipedia articles with 0.93 precision and 0.90 recall.

7.2. Disambiguation algorithm

Correctly identifying a hypernym is an essential step for linking the source entity to DBpedia. The credibility of the most frequent sense assumption made by the linking algorithm was evaluated on a set of 466 hypernym–document pairs. These were all groundtruth hypernyms in the English dataset introduced in Section 7.1.[15] The hypernyms were issued as queries to the Wikipedia search observing whether the first hit matches the semantics of the hypernym in the context of the original article.

Three raters have evaluated the results of this experiment. The consensus was determined based on a majority vote. The percentage of ratings in each category is presented in Table 6.

[14] Consider sentence: "Rhabditida is an order of free-living, zooparasitic and phytoparasitic microbivorous nematodes (roundworms)". The THD assigned hypernym "order" was considered incorrect, as the annotators agreed on "nematodes". Both "order" and "nematodes" are, however, valid hypernyms for Rhabditida.

[15] For 34 documents the groundtruth was "no hypernym".

Table 5
Hypernym discovery results. In column labels, A refers to the human annotation, and B to computer-generated result.

Language	Docs	Docs with groundtruth	Match	Only A	Only B	Partially correct	Precision	Recall	F1.0
English	500	500	411	55	24	0	0.94	0.88	0.91
German	497	488	409	45	23	2	0.94	0.90	0.92
German-person	225	223	205	10	4	1	0.98	0.95	0.96
Dutch	500	495	428	45	34	3	0.92	0.90	0.91

Table 6
Evaluation of the disambiguation algorithm (consensus rating).

Language	Total docs	Docs with hypernym	Docs with consensus	Precise	Imprecise	Disambiguation page	Incorrect
English	500	466	464	69.4%	7.1%	21.1%	2.4%

Table 7
Inter-rater agreement (English), κ refers to Cohen's Kappa for two raters, and Agreem. to the number of matching ratings divided by the number of all ratings.

Metric	ann1 vs ann2		ann1 vs agr		ann2 vs agr	
	plain	linked	plain	linked	plain	linked
κ	0.702	0.667	**0.930**	**0.925**	0.767	0.743
Agreem.	0.973	0.925	**0.993**	**0.981**	0.980	0.944

Table 8
Overall accuracy.

	ann1		ann2		Agreement	
Dataset	Plain	Linked	Plain	Linked	Plain	Linked
Dutch	**0.93**	**0.88**	NA	NA	NA	NA
English	0.95	0.85	0.96	0.90	**0.95**	**0.86**
German	**0.95**	**0.77**	NA	NA	NA	NA

The results indicate that with only 2.4% incorrect type assignments the hypernym linking algorithm does not make many outright errors. However, 21% of articles is mapped to an ambiguate type (a disambiguation page), selecting a correct specific sense would thus be a valuable direction for future work.

7.3. Overall accuracy

This integrating experiment focused on evaluating the accuracy of entity-linked hypernym tuples in the Linked Hypernyms Dataset. In contrast to the setup of the separate evaluation of the disambiguation algorithm reported in Section 7.2, the input are the RDF type triples that have been subject to the data cleansing and DBpedia alignment. Also, the evaluation guidelines required the rater to assess the correctness of the triples also when the type (linked hypernym) is a disambiguation page. If any of the listed senses covers the entity, the linked hypernym is correct, otherwise it is marked as incorrect.

The sample size of 1000 allowed to report all results with the lower and upper limits of the 95% confidence interval within approximately 2%–3% from the average accuracy on the sample. The 2% span was also used to evaluate the type relation in YAGO2 [2]. For English, all entities were judged by two raters (students), when there was no consensus, judgments of the third rater (expert ontologist) were requested. The groundtruth was established by the consensus of two raters. The degree of agreement among the raters is provided by Table 7.

The results indicate almost perfect match between the judgments provided by rater 1 and the consensus judgments. For German and Dutch, the results are only based on the judgments of the best performing rater 1.

For each entity-linked hypernym pair the task was to assess whether the linked hypernym is a correct type for the entity. For linked hypernym pointing to a DBpedia ontology class, this was determined based on the description of the class, for DBpedia resources, based on the content of the associated Wikipedia page.

As a supplementary task, the rater(s) also assessed the correctness of the plain text hypernym.

The overall accuracy of the Linked Hypernyms Dataset as evaluated on 1000 randomly drawn entities per language is reported in Table 8. A direct comparison with Tipalo has not been attempted, since it uses a different reference type system (DULplus). The accuracy on the English dataset can be, however, compared with the YAGO2 ontology: the accuracy of linked hypernyms (*linked* in Table 8) is at 0.86 lower than the average accuracy of the type relation (0.98) reported for YAGO [2]. It should be noted that the accuracy of the plain text hypernyms (*plain* in Table 8) is in the range of 0.93–0.95 for all three languages. This shows that the error is mainly introduced by the disambiguation algorithm.

The following Sections 7.4 and 7.5 present additional evaluations on 11,350 entities from individual subsets of the English Linked Hypernyms Dataset using the same methodology, but only with one rater. The use of only one rater is justified by the high agreement with the inter-rater consensus in the English overall accuracy evaluation.

It should be noted that in the evaluations, the mappings to ontology classes resulting from approximate matching were not considered. This applies both to the evaluation of the overall accuracy as well as to the evaluation on the individual subsets performed in the following Sections 7.4 and 7.5. Also, the comparison of the results with YAGO2s in this section is only indicative, due to variations in the rating setup.

7.4. Accuracy of the DBpedia enrichment dataset

This experiment focused on evaluating the accuracy of statements that were found to be novel with respect to a) DBpedia, and b) DBpedia and YAGO2s.

As the DBpedia only baseline, all three parts of the *DBpedia Enrichment Dataset* are used: *"Mapped/New—No Overlap"*, *"Not Mapped/New"*, and *"Mapped/New—No Type"*. Each of these was further partitioned to four subsets according to YAGO2s overlap (see Table 4). For measuring the accuracy of entity-linked hypernym pairs novel w.r.t. YAGO2s, the partitions of the *DBpedia Enrichment Dataset* with either no YAGO2s type assigned or with no match against YAGO2s are used. Nine evaluations were performed, each on a random sample of 831–1000 entities from the respective dataset.

The results of the evaluation are provided in Table 3. The best performing dataset is – surprisingly – dataset *Mapped / New—No Type* which contains entities with no type assigned by DBpedia. While type extraction from the semistructured information used to populate the DBpedia type relation presumably failed for these 198,040 entities, THD provides a type with accuracy of 0.94. The weighted average accuracy for the *DBpedia Enrichment* dataset containing 1,060,365 entities is 0.85.

The total number of RDF type triples novel with respect to DBpedia and simultaneously with YAGO2s (*YAGO Enrichment* dataset) amounts to 602,027 (*YAGO No Type* + *YAGO No Match* partitions in Table 4). For the hardest subset, where neither DBpedia nor YAGO2s assign any type,[16] the accuracy is 0.87.

7.5. Accuracy of statements confirmed by YAGO

The subject of evaluation are subsets of the DBpedia Enrichment Datasets containing entities for which the linked hypernym does not match any DBpedia assigned type, but there is an exact match with a YAGO2s type. The number of entities in these subsets is 333,784, the average accuracy is 0.91. Three evaluations were performed, each on a random sample of 878–1000 entities from the respective dataset. The results for all three subsets are reported in bold in Table 4.

Interestingly, the *YAGO Exact Match* partition of *Mapped / New—No Overlap* exhibits accuracy of 0.994. For the entities in this dataset[17] the type is assigned with higher accuracy than is the 0.9768 average accuracy for the type relation reported for the YAGO ontology [2] (chi-square test with $p < 0.05$).

This nearly 2% improvement over YAGO indicates that the free-text modality can be successfully combined with the semistructured information in Wikipedia to obtain nearly 100% correct results. The second, and perhaps more important use for the rediscovered RDF type triples is the identification of *the most common type* as seen by the author(s) of the corresponding Wikipedia entry.

8. Extending coverage—LHD 2.0

Even after the ontology alignment, most RDF type statements in LHD have a DBpedia resource as a type, rather than a class from the DBpedia Ontology.

Increasing the number of entities aligned to the DBpedia Ontology is a subject of ongoing work. Alignment of the types for which the simple string matching solution failed to provide a mapping was attempted with state-of-the-art ontology alignment algorithms in [20]. Experiments were performed with LogMapLt, YAM++ and Falcon, all tools with a success record in the Ontology Alignment Evaluation Initiative.[18]

Best results were eventually obtained with a statistical type inference algorithm proposed specifically for this problem. Using this algorithm, the draft version 2.0 of LHD [20] maps more than 95% of entities in the English dataset to DBpedia Ontology classes. For German and Dutch the number of entities with a type from the dbo namespace is also increased significantly. It should be noted that this increase in coverage comes at a cost of reduced precision. LHD 2.0 draft is thus an extension, rather than a replacement for the version of the dataset presented in this paper.

Example.
The following statements from the "notmapped" partitions (cf. Sections 6.5 and 6.3):

```
dbp:H._R._Cox rdf:type dbp:Bacteriologist .
dbp:Boston_Cyberarts_Festival rdf:type dbp:Festival .
```

are supplemented in LHD 2.0 draft with:

```
dbp:H._R._Cox rdf:type dbo:Scientist .
dbp:Boston_Cyberarts_Festival rdf:type dbo:MusicFestival.
```

9. Extending LHD to other languages

Extending LHD to another language requires the availability of a part-of-speech tagger and a manually devised JAPE grammar adjusted to the tagset of the selected tagger as well as to the language.

The first precondition is fulfilled for most languages with many speakers. POS taggers for French, Italian and Russian, languages currently uncovered by LHD, are all available within the TreeTagger framework. For other languages there are third-party taggers that can be integrated. Next, manually devising a JAPE grammar requires some effort, first on creating a development set of articles with tagged hypernyms, and subsequently on tweaking the grammar to provide the optimum balance between precision and recall.

A viable option, which could lead to a fully automated solution, is generating a labeled set of articles by annotating as hypernyms noun phrases that match any of the types assigned in DBpedia, and subsequently using this set to train a hypernym tagger, e.g. as proposed in [13]. The hypernyms output by the tagger could be used in the same way as hypernyms identified by the hand-crafted JAPE grammars, leaving the rest of the LHD generation framework unaffected.

The LHD Generation framework has been made available under an open source license. The published framework differs in the workflow presented in this article in that it performs hypernym extraction from the article abstracts included in the DBpedia RDF n-triples dump (instead of the Wikipedia dump).

10. Conclusion

This paper introduced the Linked Hypernyms Dataset containing 2.8 million RDF type triples. Since the types were obtained from the free text of Wikipedia articles, the dataset is to a large extent complementary to DBpedia and YAGO ontologies, which are populated particularly based on the semistructured information—infoboxes and article categories.

The Linked Hypernyms Dataset generation framework adapts previously published algorithms and approaches, which were proposed for extracting hypernyms from electronic dictionaries and encyclopedic resources, and applies them on large scale on English, Dutch and German Wikipedias.

Using three annotators and 500 articles per language, the F1 measure for hypernym discovery was found to exceed 0.90 for all languages. The best results were obtained for the German person subset, with precision 0.98 and recall 0.95.

The disambiguation algorithm, which is used to link the hypernyms to DBpedia resources, was evaluated on 466 English article–hypernym pairs. This experiment pointed at the fact that while there was only 2.4% incorrect type assignments, 21% of the linked hypernyms are disambiguation entities (articles). Selecting the correct specific sense would be an interesting area of future work.

The third integrating evaluation assessed the cumulative performance of the entire pipeline generating the Linked Hypernyms Dataset: hypernym discovery, disambiguation, data cleansing and DBpedia ontology alignment. The human evaluation was reported separately for the entire English, German and Dutch datasets. The English dataset was subject to further analysis, with evaluation results reported for its twelve interesting partitions. Compared to existing work on DBpedia enrichment or hypernym learning (e.g. [13,14,16]), an order-of-magnitude more human judgments were elicited to assess the quality of the dataset.

Some of the results are as follows: The accuracy for the 1 million RDF type triples novel with respect to DBpedia is 0.85% ± 2%, out of these the highest accuracy (0.94) is for the subset of 198,040

[16] Note that part of the discrepancy in entity coverage between the Linked Hypernyms Dataset, DBpedia and YAGO2s is due to Wikipedia snapshots used to populate the datasets being from different timepoints.

[17] Out of the total 59,365 entries, entities for evaluation were sampled from the 50,274 entities with type from the dbo namespace (entities with approximate mappings were excluded).

[18] http://oaei.ontologymatching.org/.

entities, which have no DBpedia type. With accuracy 0.87, the Linked Hypernyms Dataset provides a new type for 38,000 entities that had previously no YAGO2s or DBpedia Ontology type.

There are about 770 thousand novel RDF type triples for the German dataset, and 650 thousand for the Dutch dataset. The number of these RDF type triples exceeds the number of entities in the localized DBpedia 3.8 for the respective language. Version of the YAGO2s ontology for localized Wikipedias is not provided.

In addition to enriching DBpedia and YAGO2s with new types, it was demonstrated that the part of the Linked Hypernyms Dataset which overlaps with YAGO2s or DBpedia can be utilized to obtain a set of RDF type triples with nearly 100% accuracy.

There is a body of possible future extensions both on the linked data and linguistic levels. A certain limitation of the Linked Hypernyms Dataset is that a large number of linked hypernyms is not mapped to the DBpedia *ontology*. In the draft 2.0 version of the dataset, a statistical ontology alignment algorithm has been used to achieve a close to 100% coverage with DBpedia Ontology classes [20], however, at the cost of lower precision. Another viable direction of future work is investigation of the supplementary information obtainable from Targeted Hypernym Discovery. For example, according to empirical observation, the first sentence of the article gives several hints regarding temporal validity of the statements. For people, the past tense of the verb in the first sentence indicates that the person is deceased, while the object in the Hearst pattern preceded with limited vocabulary of words like "former" or "retired" hints at the hypernym (presumably vocation) not being temporarily valid.

The datasets are released under the Creative Commons license and are available for download from http://ner.vse.cz/datasets/linkedhypernyms. The raw data (human and computer generated hypernyms) used for the experimental evaluation, the annotation results, ratings and guidelines are also available. The LHD 1.3.8 release described and evaluated in this article targets DBpedia 3.8, version for DBpedia 3.9 containing 4.5 million RDF type triples is also available for download. Updated LHD generation framework for DBpedia 3.9 is available under an open source license. An example of an application which uses LHD to complement DBpedia and YAGO is a web-based entity recognition and classification system http://entityclassifier.eu [21]. The German LHD partition has been incorporated into the German DBpedia by the German DBpedia chapter to improve coverage with RDF Type triples.[19]

Acknowledgments

The author thanks the anonymous reviewers for their insightful comments. This work has been supported by the EU LinkedTV project (FP7-287911). The author would like to thank Milan Dojchinovski for insightful comments, and, in the first place, programming entityclassifier.eu, the first application using LHD.

References

[1] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia—a crystallization point for the web of data, Web Semant. 7 (2009) 154–165.

[2] J. Hoffart, F.M. Suchanek, K. Berberich, G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia, Artificial Intelligence 194 (2013) 28–61.

[3] Y.A. Wilks, B.M. Slator, L. Guthrie, Electric Words: Dictionaries, Computers, and Meanings, ACL-MIT Series in Natural Language Processing, The MIT Press, Cambridge, Mass., 1996.

[4] N. Calzolari, Detecting patterns in a lexical data base, in: Proceedings of the 10th International Conference on Computational Linguistics and 22Nd Annual Meeting on Association for Computational Linguistics, ACL'84, Association for Computational Linguistics, Stroudsburg, PA, USA, 1984, pp. 170–173. http://dx.doi.org/10.3115/980491.980527.

[5] M.A. Hearst, Automatic acquisition of hyponyms from large text corpora, in: Proceedings of the 14th Conference on Computational Linguistics, Vol. 2, COLING'92, ACL, Stroudsburg, PA, USA, 1992, pp. 539–545. http://dx.doi.org/10.3115/992133.992154.

[6] R. Snow, D. Jurafsky, A.Y. Ng, Learning syntactic patterns for automatic hypernym discovery, in: Advances in Neural Information Processing Systems, 17, MIT Press, Cambridge, MA, 2005, pp. 1297–1304.

[7] R. Snow, D. Jurafsky, A.Y. Ng, Semantic taxonomy induction from heterogenous evidence, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, ACL-44, Association for Computational Linguistics, Stroudsburg, PA, USA, 2006, pp. 801–808. http://dx.doi.org/10.3115/1220175.1220276.

[8] P. Cimiano, J. Völker, Text2onto: a framework for ontology learning and data-driven change discovery, in: Proceedings of the 10th International Conference on Natural Language Processing and Information Systems, NLDB'05, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 227–238. http://dx.doi.org/10.1007/11428817_21.

[9] S.P. Ponzetto, M. Strube, Deriving a large scale taxonomy from wikipedia, in: Proceedings of the 22nd National Conference on Artificial Intelligence, in: AAAI'07, vol. 2, AAAI Press, 2007, pp. 1440–1445. URL: http://dl.acm.org/citation.cfm?id=1619797.1619876.

[10] J. Kazama, K. Torisawa, Exploiting Wikipedia as external knowledge for named entity recognition, in: Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL'07, 2007, pp. 698–707.

[11] A. Ritter, S. Soderland, O. Etzioni, What is this, anyway: Automatic hypernym discovery, in: Proceedings of AAAI-09 Spring Symposium On Learning, 2009, pp. 88–93.

[12] T. Kliegr, K. Chandramouli, J. Nemrava, V. Svátek, E. Izquierdo, Combining image captions and visual analysis for image concept classification, in: Proceedings of the 9th International Workshop on Multimedia Data Mining: Held in Conjunction With the ACM SIGKDD 2008, MDM'08, ACM, New York, NY, USA, 2008, pp. 8–17. http://dx.doi.org/10.1145/1509212.1509214.

[13] B. Litz, H. Langer, R. Malaka, Sequential supervised learning for hypernym discovery from Wikipedia, in: A. Fred, J.L.G. Dietz, K. Liu, J. Filipe (Eds.), Knowledge Discovery, Knowlege Engineering and Knowledge Management, in: Communications in Computer and Information Science, vol. 128, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 68–80.

[14] A. Gangemi, A.G. Nuzzolese, V. Presutti, F. Draicchio, A. Musetti, P. Ciancarini, Automatic typing of DBpedia entities, in: P. Cudre-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J. X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, E. Blomqvist (Eds.), The Semantic Web—ISWC 2012, in: Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2012, pp. 65–81. http://dx.doi.org/10.1007/978-3-642-35176-1_5.

[15] A. Gangemi, A comparison of knowledge extraction tools for the semantic web, in: ESWC, 2013, pp. 351–366.

[16] H. Paulheim, Browsing linked open data with auto complete, in: Proceedings of the Semantic Web Challenge co-located with ISWC2012, Univ., Mannheim, Boston, US, 2012.

[17] A.P. Aprosio, C. Giuliano, A. Lavelli, Automatic expansion of DBpedia exploiting Wikipedia cross-language information, in: The Semantic Web: Semantics and Big Data, 10th International Conference, ESWC 2013, Montpellier, France, May 26–30, 2013. Proceedings, 2013, pp. 397–411. http://dx.doi.org/10.1007/978-3-642-38288-8_27.

[18] M. Dojchinovski, T. Kliegr, I. Lašek, O. Zamazal, Wikipedia search as effective entity linking algorithm, in: Proceedings of the Sixth Text Analysis Conference, TAC'13, NIST, 2013.

[19] RDF Core working group, N-triples: W3C RDF Core WG internal working draft, 2001. http://www.w3.org/2001/sw/RDFCore/ntriples/.

[20] T. Kliegr, O. Zamazal, Towards linked hypernyms dataset 2.0: complementing dbpedia with hypernym discovery, in: Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC-2014), Reykjavik, Iceland, May 26–31, 2014, 2014, pp. 3517–3523. URL: http://www.lrec-conf.org/proceedings/lrec2014/summaries/703.html.

[21] M. Dojchinovski, T. Kliegr, Entityclassifier.eu: real-time classification of entities in text with Wikipedia, in: ECML'13, Springer, 2013, pp. 654–658.

[19] http://de.dbpedia.org/node/30.

Appendix B: LHD 2.0: A text mining approach to typing entities in knowledge graphs

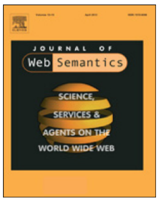
J2 Tomáš Kliegr, Ondřej Zamazal, LHD 2.0: A text mining approach to typing entities in knowledge graphs, Web Semantics: Science, Services and Agents on the World Wide Web, Volume 39, 2016, Pages 47-61, ISSN 1570-8268, https://doi.org/10.1016/j.websem.2016.05.001.

Web Semantics: Science, Services and Agents on the World Wide Web 39 (2016) 47–61

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem



LHD 2.0: A text mining approach to typing entities in knowledge graphs

Tomáš Kliegr[a,b,*,1], Ondřej Zamazal[a,1]

[a] *Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, Prague, nám. W Churchilla 4, 13067, Prague, Czech Republic*

[b] *Multimedia and Vision Research Group, Queen Mary, University of London, 327 Mile End Road, London E1 4NS, United Kingdom*

ARTICLE INFO

Article history:
Received 23 July 2015
Received in revised form
10 May 2016
Accepted 10 May 2016
Available online 2 June 2016

Keywords:
Type inference
Support Vector Machines
Entity classification
DBpedia

ABSTRACT

The type of the entity being described is one of the key pieces of information in linked data knowledge graphs. In this article, we introduce a novel technique for type inference that extracts types from the free text description of the entity combining lexico-syntactic pattern analysis with supervised classification. For lexico-syntactic (Hearst) pattern-based extraction we use our previously published Linked Hypernyms Dataset Framework. Its output is mapped to the DBpedia Ontology with exact string matching complemented with a novel co-occurrence-based algorithm STI. This algorithm maps classes appearing in one knowledge graph to a different set of classes appearing in another knowledge graph provided that the two graphs contain common set of typed instances. The supervised results are obtained from a hierarchy of Support Vector Machines classifiers (hSVM) trained on the bag-of-words representation of short abstracts and categories of Wikipedia articles. The results of both approaches are probabilistically fused. For evaluation we created a gold-standard dataset covering over 2000 DBpedia entities using a commercial crowdsourcing service. The hierarchical precision of our hSVM and STI approaches is comparable to SDType, the current state-of-the-art type inference algorithm, while the set of applicable instances is largely complementary to SDType as our algorithms do not require semantic properties in the knowledge graph to type an instance. The paper also provides a comprehensive evaluation of type assignment in DBpedia in terms of hierarchical precision, recall and exact match with the gold standard. Dataset generated by a version of the presented approach is included in DBpedia 2015.

1. Introduction

One of the most important pieces of information in linked data knowledge graphs is the *type* of the entities described. The next generation linked open data enabled applications, such as entity classification systems, require complete, accurate and specific type information. However, many entities in the most commonly used semantic knowledge graphs miss a type. For example, DBpedia 3.9 is estimated to have at least 2.7 million missing types with the percentage of entities without any type being estimated at 20% [1]. Type inference has thus received increased attention in the recent years, with the approaches proposed taking either of the two principal paths: statistical processing of information that is already present in the knowledge graph, or extraction of additional types from the free text. In this article we introduce a novel technique for type inference which combines lexico-syntactic analysis of the free text and machine learning. This combined approach can complete types for about 70% of Wikipedia articles without a type in DBpedia.

Our previously published Linked Hypernyms Dataset (LHD) framework [2] extracts types from the first sentence of Wikipedia articles using lexico-syntactic patterns. In this work we extend it with Statistical Type Inference (STI) which helps to map LHD results to the DBpedia Ontology used by the native DBpedia solution. STI algorithm is a generic co-occurrence-based algorithm for mapping classes appearing in one knowledge graph to a different set of classes appearing in another knowledge graph provided that the two knowledge graphs contain common set of instances. In our setup, our target knowledge graph is DBpedia, and the source knowledge graph is LHD.

[*] Corresponding author at: Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, Prague, nám. W Churchilla 4, 13067, Prague, Czech Republic.

E-mail addresses: tomas.kliegr@vse.cz (T. Kliegr), ondrej.zamazal@vse.cz (O. Zamazal).

[1] Both authors contributed equally.

http://dx.doi.org/10.1016/j.websem.2016.05.001
1570-8268/© 2016 Elsevier B.V. All rights reserved.

There are many articles for which lexico-syntactic patterns fail to extract any type. To address this, we employ Support Vector Machines (SVMs) trained on the bag-of-words representation of short abstracts and categories of Wikipedia articles. This supervised machine learning approach gives us a second set of entity type assignments.

In order to exploit the complementary character of the co-occurrence based STI algorithm and the supervised SVM models, we implement an ontology-aware fusion approach based on the multiplicative scoring rule proposed for hierarchical SVM classification. The hSVM algorithm can also be used separately as a language independent way to assign types since it uses abstract or categories as input feature set and it does not require language-specific preprocessing.

We validate our work on DBpedia 2014 [3], one of the most widely used Wikipedia-based knowledge graphs, the algorithmic approach is applicable also to the YAGO knowledge base [4], as well as to other semantic resources which contain instances (entities) that are (a) classified according to a taxonomy, and (b) described with a free text definition.

The evaluation of our algorithms is performed on DBpedia using a gold standard dataset comprising more than 2000 entities annotated with types from the DBpedia ontology using a crowdsourcing service.

The dataset generated with an earlier version of our approach is part of the DBpedia 2015-04 release as *Inferred Types LHD* dataset.

Parts of the work presented in this article have been published within the conference paper "Towards Linked Hypernyms Dataset 2.0: complementing DBpedia with hypernym discovery and statistical type inference (Kliegr and Zamazal, 2014)" [5]. This article extends the conference paper by introducing the hierarchical SVM approach and by performing extensive evaluation on the contributed gold standard dataset allowing the community to track progress in accuracy and coverage of entity typing and extraction tools. Also, the review of related work was substantially expanded.

The article is organized as follows. Section 2 gives an overview of related work, focusing on approaches for inference of entity types in DBpedia. Section 3 gives an overview of our approach. Section 4 describes how our LHD framework extracts types from the first sentence of Wikipedia articles and disambiguates them to DBpedia concepts. Section 5 presents the proposed algorithm for statistical type inference. Section 6 introduces the hierarchical support vector machines classifier. Section 7 describes the fusion algorithm. Section 8 presents the evaluation on the crowdsourced content and comparison with the state-of-the-art SDType algorithm and the DBpedia infobox-based extraction framework. The conclusions provide a summary of the results and an outlook for future work.

2. Related work

Completing missing types based on statistical processing of the information already present in the knowledge graph is in current research approached from several directions: (a) RDFS reasoning, (b) obtaining types through the analysis of the unstructured content with patterns, (c) machine learning models trained on labeled data, (d) unsupervised models that perform inference from statistical distributions of types, instances and the relations between them.

The four approaches listed above are covered in Sections 2.1–2.4. Section 2.5 covers the comparison of our STI/hSVM with SDType, which is a state-of-the-art unsupervised algorithm actually used for type inference in DBpedia 3.9 and DBpedia 2014. Section 2.6 motivates our choice of hSVM as a suitable machine learning classifier. Since we perceive the crowdsourced gold standard as an important element of our contribution, Section 2.7 reviews methods and resources for evaluation of algorithms that assign types to DBpedia entities. Table 1 gives an overview of selected related algorithms in terms of the methods and input features used and provides a comparison with our solution described in this article. A recent broader overview of approaches for knowledge graph refinement is present in [6].

2.1. RDFS reasoning

The standard approach to the inference of new types in semantic web knowledge graphs is RDFS reasoning. There are two general requirements enabling RDFS reasoning. First, these graphs need to have *domain* and *range* for properties specified and, second, they need to contain the corresponding *RDF facts* employing the defined properties. However, since according to common ontology design best practices (e.g. in Noy et al. [11]), domain and range should be defined in a rather general way, the inferred types tend not to be very specific. Also, *type propagation* goes upward along the taxonomy as a result of interaction of the subsumption knowledge from the ontology with the RDF facts from a dataset. Hence, RDFS reasoning usually cannot infer a specific type (i.e. type low in the hierarchy).

Furthermore, it is well known that RDFS reasoning approach will not correctly work for problems where the knowledge graph contains false statements (which is the case for DBpedia), since the errors are amplified in the reasoning process. Additional discussion on unsuitability of reasoners for type inference in DBpedia has been presented by Paulheim and Bizer in [8].

2.2. Pattern-based analysis of unstructured content

Major semantic knowledge graphs DBpedia and YAGO are populated from the *semistructured data* in Wikipedia—infoboxes and article categories using extraction framework that primarily relies on hand-crafted patterns. Approaches that extract types from the *free text* of Wikipedia articles can be used to assign types to articles for which the semistructured data are either not available, or the extraction for some reason failed.

The analysis of the unstructured (free text) content also often involves hand-crafted patterns. Tipalo, presented by Gangemi et al. in [7], covers the complete process of generating types for DBpedia entities from the free text of Wikipedia articles using a set of heuristics based on graph patterns. The algorithm starts with identifying the first sentence in the abstract which contains the definition of the entity. In case a coreference is detected, a concatenation of two sentences from the article abstract is returned. The resulting natural language fragment is deep parsed for entity definitions.

Our STI component uses as input types that were extracted from the free text with lexico-syntactic patterns with the Linked Hypernyms Dataset extraction framework presented in [12]. This framework proceeds similarly with Tipalo in that it extracts the hypernym directly from the POS-tagged first sentence and then links it to a DBpedia entity.

The accuracy of LHD matches the results for Tipalo algorithm – as reported by its authors in [7] – for the type selection subtask (0.93 precision and 0.90 recall). A detailed comparison between LHD and Tipalo is presented in [2] as well as a more extensive literature review on pattern-based extraction.

A conceptual disadvantage of pattern-based approaches is that they require relatively complex natural language processing pipeline, which is costly to adapt for a particular language. In contrast, the hSVM approach that we introduce in this article has essentially no language-specific dependencies, apart from basic tokenization, which makes its portability to another language comparatively straightforward.

Table 1
Overview of related algorithms and components of our solution (simplified).

Algorithm	Method	Input features
Related algorithms		
Tipalo [7]	Linguistic parsing	First two sentences of Wikipedia articles
SDtype [8]	Co-occurrence analysis	Ingoing properties in DBpedia
TRank [9]	Supervised machine learning (best–decision tree)	Schema and instance relations in DBpedia and YAGO
"Autocomplete" [10]	Co-occurrence analysis	Existing type assignments in DBpedia
Components of our algorithmic solution		
LHD [2]	Linguistic parsing	First sentence in Wikipedia articles
STI	Co-occurrence analysis	Type assignments in DBpedia and LHD
hSVM	Supervised ml. (Support Vector Machines)	Wikipedia article abstract and categories

2.3. Supervised methods

One of the first supervised approaches was, according to Paulheim and Bizer [1], an iterative algorithm proposed in a relational data context by Neville and Jensen in [13]. The training instances are described by attributes derived from relations of the instance (object) to other instances (objects). Additionally, the high confidence inferred statements are inserted into the data and used in the subsequent inference process, which allows to define attributes that are dependent on the result of classification in earlier iterations.

In the experiments presented in the original paper the inferred property was the type (companies were classified by industry). The relations considered included *subsidiary*, *owner* and *percentage owned for given owner*. Example attributes included *the number of subsidiaries* and *whether the company is linked to more than one chemical company through its insider owners*. Interestingly, for a given instance the value of the latter attribute can change as the algorithm progresses through the iterations.

To the best of our knowledge, the first supervised type inference algorithm applied directly in the semantic web context to assign type was described by Sleeman and Finin in [14]. This approach uses information gain as a feature selection algorithm and Support Vector Machines (SVM) for classification. The reported F-measure is between 24.9%–92.9%.

In addition to other differences to our approach such as a different input feature set, the two algorithms presented above take the flattened approach to classification, as they do not consider the taxonomical structure of target labels: each target label is a separate class. In contrast, our hSVM algorithm takes the hierarchical approach to classification, which has been shown by Liu et al. in [15], to have a superior performance when large taxonomies are involved.

Another type of supervised approach is exemplified by the TRank system [9], which ranks possible entity types given an entity and context. The TRank authors evaluated several type-hierarchy and graph-based approaches that exploit both schema and instance relations. This work is not directly comparable to ours, because the aim of TRank is to select *type for given entity mention in a longer context* (sentence, paragraph, three paragraphs), while we aim to assign types for already *disambiguated articles* describing the entity. What is particularly relevant to our work is the evaluation methodology, as the collection of TRank algorithms was similarly to our work evaluated with crowdsourcing.

2.4. Unsupervised methods

Recently, several unsupervised machine learning algorithms for type inference emerged. Paulheim [10] describes the use of association rule mining to discover missing types for a specific entity. To improve scalability, a lazy association rule algorithm is used to learn only rules that are relevant for the types associated with the specific entity. The confidence value associated by the apriori algorithm with a rule is used as type confidence. If multiple rules predict the same type, their confidence scores are aggregated.

This algorithm bears some resemblance to the STI algorithm that we proposed in [5] (also covered in Section 5), since both algorithms exploit the occurrence of types. The association rule approach is more advanced in that if the entity has multiple types, all of them can potentially contribute to the type prediction.

The STI algorithm generates a universally applicable mapping from one type to a set of types, each associated with a confidence score. The final output of the algorithm is one type which is a compromise between specificity and reliability. The advantage of STI is thus speed, since the algorithm tries to infer the mapping for the relatively small number of types (such as `dbpedia:Playwright`), rather than individually processing all entities. Since the algorithm can also be applied to instances without any type previously assigned, STI can be expected to cover wider range of untyped entities than the association rule learning approach.

SDType, covered in detail in the next subsection, is a state-of-the-art algorithm for type inference proposed by Paulheim and Bizer [8], which as its authors assert provides superior results in terms of F-measure compared to all the earlier approaches. The results of the SDType algorithm are also included in the official release of English DBpedia as the *Heuristics* dataset.

2.5. SDType algorithm

The SDType algorithm assigns types based on ingoing *properties* of the object. The properties are readily available in DBpedia as they have been extracted from the article infoboxes.

For each relation p (e.g. `dbo:location`)[2] the algorithm computes the conditional probability that a specific entity x is of certain type if x appears as a *subject* of the relation p. Likewise, a dual conditional probability is computed for x as the object of the same relation. Additionally, each relation p is assigned a weight, which reflects the discriminative power of the property.

SDType authors consider as untypeable e.g. lists or disambiguation articles. To limit the number of false statements that would be generated if these entities are reassigned with types, the initial step of SDType is to determine whether the entity is typeable using a machine learning classifier. The authors report that 5.5% of entities was found as not typeable. Our LHD generation process excludes entities listed in the DBpedia *disambiguations* dataset, which also corresponds to roughly 5.6% of entities for English DBpedia 2014.

Using the probability distributions associated with properties attached to an entity, the SDType algorithm outputs a confidence score for each entity-type pair. A predefined cutoff threshold balances the number of inferred types and their quality.

[2] `dbo` refers to the DBpedia ontology namespace http://dbpedia.org/ontology/.

SDType assigns multiple types per entity. A higher confidence threshold assigns more types at lower precision. The self-reported precision at a confidence threshold producing on average 3.1 types is 0.99 (0.95 confidence at 4.8 types). Inspection of SDType results shows that while multiple types are assigned to a given entity, these are, in our observation, typically composed of a specific type and its supertypes. STI/hSVM assigns only the most specific type (cf. Examples 1 and 2).

Example 1.
`dbpedia:Triple_Stamp_Records`
is assigned types: `dbo: RecordLabel`, `dbo:Company`, `dbo:Organisation` and `owl:Thing`
by SDType.[a] The STI/hSVM algorithm assigns a single type `dbo:RecordLabel`.

[a] DBpedia 3.9 `instance_types_heuristic_en.nt` file

Example 2.
`dbpedia:Terry_Sejnowski`
is assigned types: `dbo:Person`,
`dbo:Agent` and `owl:Thing`
by SDType. The STI/hSVM algorithm assigns a single type `dbo:Scientist`.

It should be noted that SDType has the advantage that it can generate types also for entities which are derived from Wikipedia red links. This is impossible with both STI and hSVM algorithms, which require that the article contains a short abstract. However, if an article is not referenced from infobox of another article then it cannot be processed by SDType. For STI/hSVM this is not an obstacle.

It can thus be concluded that both SDType and STI/hSVM approaches are largely complementary both what concerns the algorithmic techniques used and the set of applicable untyped entities. Section 8.5 presents a comparison of SDType and STI/hSVM in terms of accuracy on a crowdsourced gold standard dataset.

2.6. Text categorization with SVM

In order to enhance type assignment provided by the STI algorithm, we introduce a supervised model trained on the bag-of-words representation of article content. In this way, we effectively cast the problem of assigning a type to an entity as a text categorization task. The entity-type assignments already present in DBpedia serve as the training data.

From the range of applicable machine learning algorithms, we opted for Support Vector Machines (SVMs) [16]. SVMs have been found to be more accurate than other standard machine-learning algorithms such as Naive Bayes, neural networks and the Rocchio classifier on the text categorization task as reported in [17]. Experimental results presented within our evaluation (in Section 8.7) confirm the superior performance of linear SVMs over other common classification algorithms in the flat text categorization task on our data. The SVM classifier is particularly suitable as it is scalable and has been previously successfully adapted to handle tasks involving large web taxonomies [17]. We adapt the *hierarchical SVM* approach (hSVM), where a separate classifier is built for all non-terminal leaves in the class hierarchy.

The complexity of flat SVMs is proportional to the number of target classes as reported in [15]. With SVMs in a hierarchical setup, there are several options. The *sequential Boolean rule* [17] or Pachinko-machine search [15] has typically a significant performance benefit for the testing phase, since for a given test instance an SVM model for class "c" is used only if its parent category classifies the test instance to class "c". Another approach is the *multiplicative scoring rule* [17], which applies all SVM models and then combines their resulting models by multiplying the probabilities obtained by classifiers on individual levels.

The computationally efficient sequential Boolean rule was found to perform equally well as the multiplicative scoring rule and better than a flat SVM as reported in [17]. The way of merging the outputs of classification models on the individual layers is a major design choice for hierarchical classification algorithms. Since computational complexity is not a major design constraint for our use case, we opted for multiplicative scoring rule as it provides structurally more convenient output for fusion with our other approach, STI.

2.7. Evaluation of type assignment

An important part of our contribution is the evaluation of the accuracy of the inferred types and the comparison with the average accuracy in the original knowledge graph. Two fundamental approaches to checking the accuracy of the inferred types were given by Gangemi et al. in [7]: *gold standard* and *type checking*.

In the gold standard approach, one needs to create a dataset assigning each entity identifier (DBpedia URI) with one or more type URIs. Typically, several annotators participate on the design of the dataset. The advantage of this approach is that the resulting dataset is reusable as long as the system which is evaluated is able to assign types to the same set of entities. The disadvantage is that this evaluation scheme is not straightforward to apply. Requiring exact match between the assigned type and the gold standard implies that if the assigned type is more general than the gold standard (e.g. footballer vs. midfielder) then the assignment is considered as incorrect.

In the type checking approach, human users evaluate the accuracy of the types. In the Tipalo evaluation a three-value scale was available: *yes*, *maybe*, *no*. Similar evaluation scheme was also employed for YAGO [4] and LHD [2].

The type checking evaluation scheme is not reusable and potentially difficult to reproduce. The evaluation, unless performed in an environment controlled by a third-party, may be difficult to repeat. It is common that the human evaluators are students or postdocs from the same department as are the authors of the algorithm that the evaluation is intended to support. The human evaluators may thus be under implicit pressure to judge more types as relevant than they would do under other circumstance. A second problem with this scheme is that it does not express how far the type assigned by the system is from the most specific type available in the reference ontology. For example, if the system assigns type "Person" to Diego Maradona it is counted as correct to the same degree as if the assignment is "Footballer".

In this article, we present a freely available gold standard dataset that can be used for evaluation of knowledge graphs that use types mappable to the DBpedia 2014 ontology. This gold standard dataset consists of over 2000 entities with a type. The annotation process was performed using a third-party operated crowd-sourcing tool with a built-in interface for assignment of categories from a taxonomy. There was no direct contact between the authors and the annotators (three or four per entity-type assignment). The detection of under-performing annotators was handled automatically by the crowdsourcing tool. The design of the gold standard dataset was thus completely decoupled from the evaluation of the algorithm. To compare with, the gold standard used in the Tipalo tool was created for 100 entities and using annotation tool designed by the authors, the annotators were four senior researchers and six PhD students in the area of knowledge engineering.

Another broader evaluation setup that aimed at assessing the quality of data in DBpedia using crowdsourcing is presented by Zaveri et al. in [18]. This paper describes a methodology and a software tool for detecting errors in DBpedia. The authors identified 17 data quality problem types. The annotators evaluated in total 521 resources. While this research pioneers the use of crowdsourcing for evaluating DBpedia triples, it does not specifically report on the *rdf:type* relation, which is the focus of this article.

A very recent survey that scopes evaluation of type assignment is presented in [6].

3. Overview of our approach

Our algorithmic solution to type inference consists of several components. The *Linked Hypernyms Dataset* [2] is used to extract types with lexico-syntactic patterns from the first sentence of Wikipedia articles. Part of the types are mapped to DBpedia ontology using reliable exact string matching. The remaining types are mapped using our co-occurrence based *Statistical Type Inference* algorithm. STI is a novel approach for mapping classes appearing in one knowledge graph to a different set of classes appearing in another knowledge graph provided that the two knowledge graphs contain common set of instances.

A parallel path to obtaining types for an entity is a *supervised machine learning approach* with Support Vector Machines (SVMs). Entities with already assigned types in DBpedia are used as a training set and the text of the abstract and the list of article categories are used as input features.

In order to fuse the outputs of all three models (STI, SVMs on abstract, SVMs on categories), we perform early fusion by aggregating (averaging) the individual probability distributions using the *linear opinion pool* [19, Chapter 9]. After that we combine the (already aggregated) distributions for individual classes in the class hierarchy. For this, we use either the *Multiplicative Scoring Rule* (MSR) designed for combining results of SVM models in a hierarchical setting, or a variant of the algorithm called *Additive Scoring Rule* (ASR).

Our approach consists of the following succession of steps:

1. Extracting types from free text with lexico-syntactic patterns using the LHD framework, resulting types are DBpedia resource.
2. Mapping types to DBpedia ontology with exact string matching (LHD Core).
3. Mapping remaining types with Statistical Type Inference (STI), the result for each input type (in the DBpedia resource namespace) is a probability distribution over DBpedia Ontology classes.
4. Training SVM models for a subset of classes in the DBpedia ontology.
5. Applying SVM models to obtain prediction for given entity, the output for a given entity is a probability distribution over a subset of DBpedia Ontology classes.
6. The probability distributions output by the SVM models and STI are aggregated using linear opinion pool.
7. The aggregated probability distribution is processed with respect to the DBpedia ontology in order to make reliable choice of a specific type.
8. The results of LHD Core (step 2) and SVM and STI models are combined to create the final dataset.

It should be emphasized that most of the steps above correspond to individual components, which can also be used independently. Steps 1–2 are performed by the Linked Hypernyms Dataset Framework described in Section 4. Step 3, the STI algorithm, is covered by Section 5. Steps 4–5 training and applying SVM models are covered in Section 6. Finally, steps 6–7 model fusion and final type selection are described in Section 7.

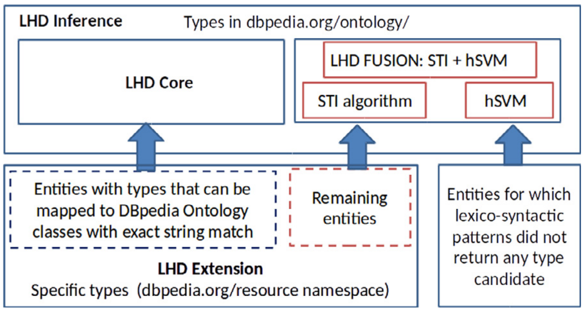

LHD Inference
Types in dbpedia.org/ontology/
LHD Core
LHD FUSION: STI + hSVM
STI algorithm
hSVM
Entities with types that can be mapped to DBpedia Ontology classes with exact string match
Remaining entities
LHD Extension
Specific types (dbpedia.org/resource namespace)
Entities for which lexico-syntactic patterns did not return any type candidate

Fig. 1. Partitions of the Linked Hypernyms Dataset.

Table 2
LHD Statistics. The dbo column indicates the portion of entities in LHD with type from the DBpedia ontology namespace, the rest is in the dbpedia namespace. The size is in thousands for the 3.9 dataset release and the accuracy was computed on the 3.8 release as reported by Kliegr in [2].

Language	Linked (total)	Linked dbo	Acc linked	Acc plain
German	893 k	199 k	0.773	0.948
English	3013 k	1136 k	0.857	0.951
Dutch	834 k	305 k	0.884	0.933

4. Linked Hypernyms Dataset

The Linked Hypernyms Dataset (LHD), introduced by Kliegr in [2], associates DBpedia entities (corresponding to Wikipedia articles) with a type which is obtained by parsing the first sentences of the respective Wikipedia article. The type is initially a plain text string, which is further disambiguated to a DBpedia entity creating a “linked hypernym”. Fig. 1 shows that the dataset is partitioned into several subsets.

The *Extension* dataset contains types in the dbpedia.org/resource namespace. This provides the highest precision types, but also the least semantic interoperability.

The types of about 50% of entities (for English, less for other languages) can be mapped to a DBpedia ontology type using a simple string matching algorithm, constituting the *Core* dataset. An overview of LHD Core in terms of size and accuracy is given in Table 2.

The entities with types extracted by the LHD framework but not mapped to the DBpedia ontology are used as input for the STI algorithm introduced in this paper. The remaining entities, for which the lexico-syntactic pattern extraction did not succeed, can be processed only with the hSVM approach, also introduced in this paper.

The *Inference* (Inferred types) dataset is published as a merge of all our approaches.

The remainder of this section briefly describes the individual steps of the LHD extraction framework: hypernym discovery, linking and the string matching approach leading to the *Core* dataset.

4.1. Hypernym discovery

The Wikipedia Manual of style [20] asserts that the page title should be the subject of the first sentence, and that it should tell the nonspecialist reader what, or who, the subject is. If the first sentence complies with these and other stated requirements, its structure can take only a limited number of forms, allowing a small number of hand-crafted patterns to cover most variations.[3]

Also, according to the Wikipedia Manual of Style "emphasis given to material should reflect its relative importance to the subject". Our decision to give preference to the first hypernym is based on the assumption that editors typically implement this clause by ordering hypernyms (e.g. occupations of a person) in the first sentence according to importance, starting with the most important one.

Our extraction framework exploits this regularity in the first sentence of Wikipedia articles. The framework is implemented on top of GATE.[4] The core of the system is a JAPE transducer (a GATE component) which applies lexico-syntactic patterns encoded as grammar in the JAPE language on the first sentence of Wikipedia articles.

The extraction grammars require that the input text is tokenized and assigned part-of-speech (POS) tags. For English, the framework relies on the ANNIE POS Tagger, available in GATE, for German and Dutch on TreeTagger.[5] Extraction grammars were hand-crafted using a development set of 600 manually annotated articles per language. The process of designing the grammars is described in detail in [2].

Example 3.
An example input for this phase is the first sentence of Wikipedia article on Václav Havel: *Havel was a Czech playwright, essayist, poet, dissident and politician.* The output is the word "playwright", the first hypernym in the sentence. The current version of the grammar outputs the head noun as the hypernym, not the complete noun chunk. Favoring head noun improves reliability as argued in [2].

The output of the hypernym discovery phase is provided as a separate dataset providing plain text, not disambiguated hypernyms. The accuracy for this dataset (denoted as "plain") is reported in Table 2.

4.2. Linking hypernyms to DBpedia instances

Once the hypernym is extracted from the article, it is disambiguated to a DBpedia identifier. The disambiguation algorithm relies on the Wikipedia Search API to resolve the string to a Wikipedia article.

Example 4.
Picking up on the Václav Havel example, the word "playwright" is used as a query, which returns the Wikipedia article http://en.wikipedia.org/wiki/Playwright. This is then translated to the DBpedia URI http://dbpedia.org/resource/Playwright.

Even if this disambiguation approach is simple, it is effective as confirmed both by our evaluation (Table 2) and by the recent results of the NIST TAC 2013 *English Entity Linking Evaluation* task, where it performed at median F1 measure (overall) [22].

4.3. Alignment with the DBpedia Ontology

While formally the output of the linking phase is already a Linked Open Data (LOD) identifier, the fact that the type is in the http://dbpedia.org/resource/ namespace (further referenced by prefix `dbpedia`) is not ideal. Concepts from this namespace are typically entities, while this term is used as a type within LHD (cf. Example 5).

Example 5.
Entity Václav Havel has type http://dbpedia.org/resource/Playwright in LHD Extension. This entity is not present in LHD Core, because there is no Playwright class in the used DBpedia Ontology version. STI assigns this entity with additional type http://dbpedia.org/ontology/Writer.

DBpedia already contains a predefined set of types within the DBpedia ontology namespace http://dbpedia.org/ontology/ (further abbreviated as `dbo`) such as `dbo:Person` or `dbo:Work`. The focus of the alignment phase is to map the original type, which is in the `dbpedia` namespace, to the `dbo` namespace.

The mappings are generated using a string matching algorithm, which requires total match in concept name (`dbpedia:Person` → `dbo:Person`). For these *exact match* mappings, only the `dbo:` type is output by the generation process.

This simple approach provides a mapping to the DBpedia ontology for a large number of entities across all three supported languages. However, in relative terms, this is less than 50% for each language as shown in Table 2, the types for almost all the remaining entities are mapped with the STI algorithm covered in the next section.

A more detailed description of the LHD framework as well as additional size and evaluation metrics are presented in [2].

5. Statistical type inference (STI)

The STI algorithm is a generic co-occurrence-based algorithm for mapping classes appearing in one knowledge graph to a different set of classes appearing in another knowledge graph provided that the two knowledge graphs contain common set of instances.

The algorithm thus works with two knowledge graphs, a primary knowledge graph $\mathcal{KG}$ associated with an ontology $\mathcal{O}_{\mathcal{KG}}$, and a knowledge graph $\mathcal{KG}_{map}$ that holds entity-type assignments that we desire to map to classes in $\mathcal{O}_{\mathcal{KG}}$. Both knowledge graphs hold entity-type assignments.

STI is based on a simple co-occurrence principle. First, for a specific input type $type_{map} \in \mathcal{KG}_{map}$ it finds the distribution of types that are assigned in $\mathcal{KG}$ to the same entities as $type_{map}$ is in $\mathcal{KG}_{map}$. The problem addressed is that the most frequently co-occurring types are very generic and thus it is necessary to identify out of the pool of the co-occurring types (classes from $\mathcal{O}_{\mathcal{KG}}$) those providing the best compromise between specificity and correctness.

The approach comprises two successive algorithms. The *Candidate generation* algorithm generates a set of candidate $\mathcal{O}_{\mathcal{KG}}$ types for $type_{map}$. The *Candidate pruning and selection* algorithm then performs removal of types for which a more specific one exists while maintaining reasonable trade off with correctness. From the types surviving the pruning, the type with the highest number of

[3] In [21] we studied whether article popularity could have an effect on the adherence to the Wikipedia manual of style, and in turn to the extractability of hypernyms from the first sentence. There was some evidence as to that may be the case, but due to the small size of the sample the results were inconclusive.

[4] http://gate.ac.uk.

[5] http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/.

supporting entities is selected. A detailed description of the two algorithms follows.

Candidate generation (Algorithm 1) first identifies the set E that contains entities which have as a type in $\mathcal{KG}_{map}$ the type $type_{map}$ that we desire to map to ontology $\mathcal{O}_{\mathcal{KG}}$. Algorithm output is the list of distinct $\mathcal{O}_{\mathcal{KG}}$ types which the entities in E have along with the number of occurrences of each type stored as *supp*. Example 6 illustrates this algorithm.

Example 6.
For the entity Václav Havel, the set E contains 1842 entities with `dbpedia:Playwright` as a type in LHD Extension ($\mathcal{KG}_{map}$) for DBpedia ($\mathcal{KG}$). Skipping entities without any type in DBpedia or with a type not in the DBpedia Ontology namespace, the list of the types associated with these 1842 entities (each type is followed by entity count): Comedian:1, MemberOfParliament:1, Royalty:1, BritishRoyalty:1, MilitaryPerson:1, Presenter:1, Politician:2, OfficeHolder:7, MusicalArtist:5, Writer:266, Artist:277, Agent:521, Person:521.

The output of the Candidate generation algorithm can already be used for probabilistic type prediction for a given entity. This process is exemplified in Algorithm 3 (contained in Section 7), which outputs the conditional probabilities for the specified parent class in the target ontology.

The selection process (Algorithm 2) is two stage. In the pruning step, the algorithm iterates through the candidates removing those which are, as indicated by the numbers of supporting entities, only a supertype of a more specific type on the list of Candidates C. Higher number of supporting entities implies reliability, however, the specific types tend not to have the highest values.

Candidate *type* is removed if there is its subtype *type′* in the list of Candidates C, which has more than *TRADEOFF* * *type.supp* supporting entities. Finally, the type with the highest support is selected from the pruned set of types. The process is illustrated in Example 7.

The effect of the setting of the *TRADEOFF* constant on the specificity and accuracy of the resulting types is investigated in Section 8.8.

Algorithm 1 Candidate Generation

```
Require: $type_{map}$ a class which we desire to map, $\mathcal{O}_{\mathcal{KG}}$ a target ontology containing
    types to which the mapping should be performed, $\mathcal{KG}$ knowledge graph containing
    instances of classes from $\mathcal{O}_{\mathcal{KG}}$, $\mathcal{KG}_{map}$ knowledge graph containing instances of class
    $type_{map}$.
Ensure: $C$ – set of candidate mappings $\{\langle type \rangle\}$, where $type$ is class from $\mathcal{O}_{\mathcal{KG}}$ associated
    with probability
$C := \emptyset$
$E$ := set of instances of $type_{map}$ in $\mathcal{KG}_{map}$
for $entity \in E$ do
    $types$ := set of classes $entity$ has in $\mathcal{KG}$
    for $type \in types$ do
        if $type$ is not a $\mathcal{O}_{\mathcal{KG}}$ class then
            continue
        end if
        if $type \notin C$ then
            add $type$ to $C$
            C[type].supp := 1
        else
            // holds the number of entities assigned with $type$ in $\mathcal{KG}$ and simultane-
            ously with $type_{map}$ in $\mathcal{KG}_{map}$
            C[type].supp += 1
        end if
    end for
end for
return $C$
```

Example 7.
Candidate pruning removes Royalty, Agent, Artist and Person and Politician from the list of candidates. Royalty is removed in favor of its subclass BritishRoyalty, which has the same number of supporting entities (one). The following three types Agent, Person and Artist are removed in favor of their subclass Writer. While Writer has less supporting entities than Artist or Person or Agent, the drop in support is within tolerance of the *TRADEOFF* constant set to 0.2. Similarly, Politician is removed in favor of its subclass MemberOfParliament.
The result of pruning is: Comedian, MemberOfParliament, BritishRoyalty, MilitaryPerson, Presenter, MusicalArtist, OfficeHolder, Writer. Finally, the algorithm selects $type_{opt}$ = Writer as the type with the highest number of supporting instances in the pruned set.

The standalone output of the STI algorithm for given type is one mapping, such as `dbpedia:Playwright` → `dbo:Writer`.

Algorithm 2 Candidate Pruning and Selection

```
Require: $C = \{\langle type \rangle\}$ set of Candidates from Alg. 1, each associated with support, $T$ –
    TRADEOFF threshold
Ensure: $type^{opt}$ – class from $\mathcal{O}_{\mathcal{KG}}$
$totalSupp := \sum_{type} C[type].supp$
$discardMade$ := true
while $discardMade$ do
    $discardMade$ := false
    for $type \in C$ do
        if $\exists type' \in C$: $type'$ subclass of $type$,
        $type \neq type'$, $type'.supp > T * type.supp$ then
            remove $type$ from $C$
            $discardMade$ := true
            break
        end if
    end for
end while
return $type^{opt}$: type with the highest $supp$ from $C$
```

6. Support Vector Machines Classifiers

Since the set of target classes forms a hierarchy and we would like to experiment with fusing outputs of multiple models, we needed an algorithm that can output probability distributions, which can be easily aggregated in a hierarchical setup. SVMs meet this requirement, additionally this approach has a previous strong record in the hierarchical text categorization domain.

Our setup involves a knowledge graph $\mathcal{KG}$ containing entities, each associated with zero or more types. The types form an ontology (taxonomy) $\mathcal{O}_{\mathcal{KG}}$. The purpose of the classifier is to assign the most specific correct type from the ontology to those entities in the knowledge graph $\mathcal{KG}$ that have a missing type. In order to train the classifier, existing entity-type assignments in $\mathcal{KG}$ are used as the training data. The entities are represented using a bag-of-words model created from the textual properties associated with the entities in $\mathcal{KG}$. If an entity does not have the required textual property it is exempt from the processing.

As the classification algorithm, we use SVM with *linear kernel*, the choice of which is justified in Section 8.7. We also let the SVM implementation output probability distribution for all target classes, which is required by the fusion process.

Further, we describe our setup in a greater detail using DBpedia as the knowledge graph $\mathcal{KG}$. In DBpedia there are multiple textual properties associated with most entities. To build the classifier, we selected two of them: short abstracts and article categories.

We should ideally have an SVM classifier for each non-leaf class in the DBpedia ontology. However, since multiple classes in the DBpedia ontology have only a few instances, better results are obtained if a dedicated *classification ontology* $\mathcal{O}_{cl}$ is derived from the DBpedia ontology.

For each non-leaf type in the classification ontology, we create two classifiers: *abstract* classifier, which uses the text of the short abstract, and the *cat* classifier, which uses article categories (treated as text).

Once the classifiers have been trained, the classification models are applied to assign types to entities using the standard *Multiplicative Scoring Rule* approach or our *Additive Scoring Rule* approach. The latter has the advantage that it outputs more specific types.

This section is organized as follows. The bag-of-words feature set used by our classifier is described in Section 6.1. Section 6.2 covers the classification ontology. The final type selection from the prediction of the individual SVM models is performed after the STI results have been merged in. This is described in Section 7.

6.1. Feature set

The dataset consists of instances that correspond to entities (articles) in Wikipedia. Each entity is represented with the bag-of-words vector space model, which is created from the *short abstract* and article *categories* as retrieved from DBpedia.

Short abstracts represent entity in a more concise way than full abstracts (e.g. *John Forrest* entity is described by 208 words and 1317 characters in the case of its full abstract and by 72 words and 447 characters in the case of short abstract). In our experience, short abstracts provide comparable results to full abstracts with lower computational demands.

Categories naturally reflect a type of a given entity to a certain extent. Interestingly, they are not necessarily shorter than short abstract. It should be emphasized that we treat the article category data as text.

During the pre-processing step, short abstracts and categories are lowercased and tokenized into separate words. Further, stop words along with numbers are removed and term frequencies are computed for each pre-processed token per given entity. In the case of categories we further applied noun stemming.

6.2. Classification ontology $\mathcal{O}_{cl}$

Since a supervised model is applied, it is necessary to restrict the classification to types in the knowledge graph for which sufficient amount of training data (i.e. instances) is available.

In order to achieve this the DBpedia ontology is reduced to DBpedia types having at least 100 instances while preserving asserted hierarchical relationships. Second, DBpedia types having only one to four direct subclasses are removed. This implies that these removed DBpedia types are replaced by their DBpedia subtypes. All DBpedia types are subsumed by the most general class Thing in the classification ontology. The thresholds of 100 and 4 respectively were chosen based on small-scale experimentation of the data, additional performance improvement can be gained when these are result of proper parameter tuning.

It should be noted that we obtained slightly improved results when the automatically built classification ontology is further manually edited. We explored this possibility in one of our development prototype. Our conclusion is that the small improvement in accuracy does not offset the costs associated with this manual intervention into the classification process each time the DBpedia ontology is changed. From these experiments we include in the following at least several figures. While the particular numbers are slightly different from the automatic version, these figures can be used to illustrate the role of the classification ontology in our workflow.

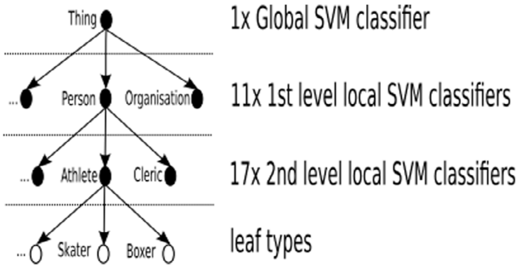


Fig. 2. Structure of hierarchical SVM classifier (29 classifiers and 276 classes in total).

Since the maximum depth of the manually edited ontology was set to three, we have three layers of SVM classifiers (see Fig. 2). There is one global SVM classifier, 11 first level local SVM classifiers and 17 s level local SVM classifiers:

- The *Global* SVM classifier covers top level types from the DBpedia Ontology (i.e. subtypes of the most general class *Thing*).
- *First level local SVM classifiers* enable classification into subtypes of (some) top level types.
- *Second level local SVM classifiers* enable classification into subtypes of (some) types assigned by the *first level local SVM classifiers*.

Table 3 contains details about the global SVM classifier and the first level local SVM classifiers, Table 4 covers second level SVM classifiers: *types* refers to the number of types the classifier distinguishes, *entities* refers to the number of entities on which the classifier was trained, *attributes* refers to the number of attributes (in the bag-of-words setting) the classifier works with and finally *accuracy* states how accurate the SVM classifier was in a ten-fold cross-validation setting.

7. Hierarchical combination of classifiers

The final step in our solution for type inference is merging the results of STI and SVM models and selecting the type that poses a compromise between specificity and reliability from the assigned ones.

The results of STI and the two SVM models (abstract and categories) are merged using *linear opinion pool*: the probability distributions output by the individual models are simply averaged. This is performed by Algorithm 4. The merging process also takes into account the situation that a prediction from a particular classifier may be missing for given class.

The SVM classifiers provide the function *parent.prob_classify(e)* that for given entity *e* outputs the conditional probabilities for a particular parent concept (an individual SVM classifier). Algorithm 3 presents how a structurally compatible output can be generated from the STI output. This short algorithm addresses two principal points:

- Making a prediction for a specific entity as STI provides mapping for classes not entities.
- Use of different target ontology as STI Candidate Generation algorithm uses the full target ontology $\mathcal{O}_{\mathcal{KG}}$, while the SVM are trained on its subset $\mathcal{O}_{cl}$. This is achieved by simply skipping the classes on STI Candidate generation output, which are not included in $\mathcal{O}_{cl}$.

Example 8 illustrates the classification with Algorithm 3.

Table 3
Global SVM classifier and first level local SVM classifiers. Each classifier has two variants (*abstract* and *cat*). The first number corresponds to a classifier based on short abstract (*abstract*) and the second one to a classifier based on categories (*cat*).

Classifier	Types	Entities	Attributes	Accuracy
Global	29/28	2900/2745	20419/4562	86%/89%
Work	13/13	1300/1295	11153/2654	86%/91%
Species	5/5	500/496	3402/623	92%/90%
Place	12/12	1200/1187	9136/2253	83%/89%
Transportation	7/7	700/700	5639/1137	95%/96%
Event	6/6	600/599	5472/1078	90%/93%
Device	3/3	300/298	3627/449	98%/98%
Organisation	12/12	1200/1194	9392/1925	91%/92%
Person	30/30	3000/3000	18980/6122	81%/81%
AnatomicalSt.	8/8	800/799	3878/191	93%/96%
CelestialBody	4/4	400/400	1969/318	97%/87%
SportsSeason	4/4	400/400	2530/590	91%/98%

Table 4
Second level local SVM classifiers. The first number corresponds to a classifier based on short abstract (*abstract*) and the second to a classifier based on categories (*cat*). *Arch.* means *ArchitecturalStructure*, *Educat.* means *EducationalInstitution* and *Popul.* means *PopulatedPlace*.

Classifier	Types	Instances	Attributes	Accuracy
WrittenWork	6/6	600/599	6287/1289	91%/95%
MusicalWork	4/4	400/400	3810/1134	82%/93%
Animal	9/9	900/896	6099/1006	87%/77%
Plant	7/7	700/692	4318/612	93%/92%
Arch.	22/22	2176/2174	13946/2991	89%/91%
SportsEvent	8/8	800/798	4253/858	96%/96%
Athlete	34/34	2550/2549	12674/4031	98%/96%
Broadcaster	3/3	300/300	2736/605	87%/83%
Company	4/4	400/400	3881/518	98%/99%
Educat.	3/3	300/300	2806/778	96%/96%
SportsLeague	5/5	500/491	2940/419	98%/98%
SportsTeam	6/6	600/595	4094/759	99%/97%
Artist	7/7	700/700	6690/1577	90%/86%
Cleric	4/4	400/400	3562/1224	95%/95%
Politician	7/7	700/700	5081/2089	76%/80%
NaturalPlace	8/8	775/773	5809/1252	90%/93%
Popul.	6/6	600/587	4972/1234	85%/86%

Example 8. Consider the use of STI-prob on the classification entity e = Václav Havel with respect to the *Artist* parent class. Method Artist.prob_classify(e) first looks up i in $\mathcal{KG}_{map}$ obtaining $type_{map}$ = `dbpedia:Playwright`. Next, it executes *candidate generation*($type_{map}$) obtaining the set of candidate classes from $\mathcal{O}_{\mathcal{KG}}$ along with support values (ref. to Example 6 featured in Section 5). Finally, these support values are converted to the following probabilities for subclasses of Artist in $\mathcal{O}_{cl}$: Comedian 0.4%, MusicalArtist 0.4%, Writer 99.2% (only classes with non-zero probability are listed).

The result of Algorithm 4 is a set of conditional probabilities assigned to classes in the classification ontology. Next we apply *multiplicative scoring rule* approach (Algorithm 5) proposed in [17] for hierarchical classification of web content with SVMs. This algorithm takes on the input computed set of conditional probabilities from Algorithm 4 and propagates their values downward the taxonomy, removing classes with joint probability lower than a preset threshold. While this approach is very simple, we feature our implementation in Algorithm 5 and 6 for reference purposes.

One modification to Algorithm 5 we experimented with was averaging the probabilities rather than computing the joint probability by multiplying them. This modification aims at more reliable selection of the final type, while maintaining reasonable specificity of the selected type. With the MSR approach, the types associated with highest probability are the ones on the most general level of the ontology. Assignment of these types would not be very useful. With averaging as the aggregation operator the maximum can be on any level. We call this modification the *additive scoring rule* (ASR). We tried adapting the pruning for ASR since the confidence associated with subtype can be higher than of its supertype in the ASR approach, however, we found the current version in Algorithm 6 to work better.

We introduce two strategies for selecting one type per entity from the multiple types that can survive the pruning step in the following subsection.

Algorithm 3 Classify instance with STI-prob *parent.prob_classify(e)*

```
Require: e entity to classify, parent in function name is a concept ∈ $\mathcal{O}_{cl}$ with respect to
    which the classification should be performed, the source knowledge base $\mathcal{KG}_{map}$
Ensure: prob prob. distribution over children of parent ∈ $\mathcal{O}_{cl}$
$type_{map}$ := type of e in $\mathcal{KG}_{map}$
C := candidate generation($type_{map}$) // see Alg.1
for type in children of parent in $\mathcal{O}_{cl}$ do
    $prob[type] = \frac{C[type].supp}{\sum_{s \in siblings(type, \mathcal{O}_{cl})} C[s].supp}$
end for
return prob
```

Algorithm 4 Linear opinion pool for hierarchy

```
Require: e – entity to be classified, $\mathcal{O}_{cl}$ Classification Ontology, cl – grid of |M| x |N|
    probabilistic classifiers, where N is the set of non-terminal types in $\mathcal{O}_{cl}$ and M the
    set of modalities, classifier for some combination of $m \in M$ and $n \in N$ may not exist,
    weight $w_m$ for each modality, $\sum_{m \in M} w_m = 1$
Ensure: prob – array of conditional probabilities associating every class $c \in \mathcal{O}_{cl}$ except
    root (Thing) with a conditional probability of c given its parent p in $\mathcal{O}_{cl}$
//there is at least one classifier for each non-terminal class
prob[∗] := 0
for non-terminal class $p \in \mathcal{O}_{cl}$ do
    // we have up to 3 modalities: SVM categories, abstract and STI. If a classifier in any
    modality is missing, the weight vector needs to be adjusted by a factor of ws
    ws := 0
    for m ∈ M do
        if classifier cl[m, p] exists then
            ws = ws + $w_m$
        end if
    end for
    for m ∈ M, c ∈ target classes of cl[m, p] do
        prob[c] := prob[c] + $\frac{w_m}{ws}$ ∗ cl[m, p].prob_classify(e)[c]
    end for
end for
return prob
```

Algorithm 5 Multiplicative Scoring Rule – Computing joint probability

```
Require: prob – conditional probability for $c \in \mathcal{O}_{cl} \setminus \{Thing\}$ given its parent p in $\mathcal{O}_{cl}$
Ensure: jprob – joint probability for $c \in \mathcal{O}_{cl} \setminus \{Thing\}$
jprob := prob[class]
// proceeds breadth-first from root to leaf
for type ∈ non-leaf classes from $\mathcal{O}_{cl} \setminus \{Thing\}$ do
    for subtype ∈ children($\mathcal{O}_{cl}$, type) do
        jprob[subtype] := jprob[subtype] × jprob[type]
    end for
end for
return jprob
```

Algorithm 6 Multiplicative Scoring Rule – Pruning

```
Require: jprob – joint probability for $c \in \mathcal{O}_{cl} \setminus \{Thing\}$, threshold T
Ensure: jprob – joint probability with some types removed
// proceeds breadth-first from root to leaf
for type ∈ classes from $\mathcal{O}_{cl} \setminus \{Thing\}$ do
    if jprob[type] ≤ T then
        for subtype ∈ descendants($\mathcal{O}_{cl}$, type) do
            remove subtype from jprob and from $\mathcal{O}_{cl}$
        end for
        remove type from jprob
    end if
end for
return jprob
```

7.1. Final type selection

By default MSR approach returns set of types. In order to provide a classification result, the algorithm selects a final type from *Candidates* according to probabilities associated with each type.

We use two approaches to determine the final type from the output of Algorithm 5 (MSR or ASR):

- α strategy selects the type with maximum joint probability from non-top[6] *leaf types*. This approach is used in conjunction with the default MSR version of the algorithm.
- β strategy selects the type with maximum joint probability from *all types*. This approach is used in conjunction with the ASR version of the algorithm.

8. Evaluation

Due to the unavailability of a suitable evaluation resource, we decided to build a *gold standard* dataset that associates a DBpedia entity (a Wikipedia article) with a manually curated list of types from the DBpedia Ontology. Such dataset allows not only to report on performance of our approach, but also to provide a comparison with other algorithms in an objective way.

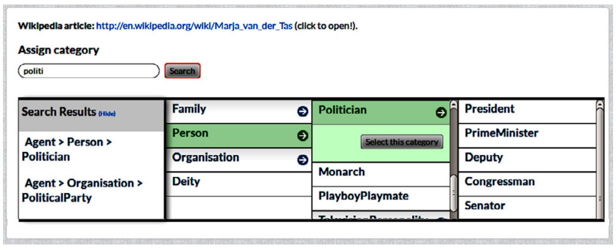
The annotation setup for the three gold standard datasets GS1, GS2 and GS3 is described in Section 8.1. Evaluation metrics are described in Section 8.2. Section 8.3 describes the setup of our algorithms and Section 8.4 presents their results. Section 8.5 provides a comparison with the SDType algorithm. Section 8.6 evaluates the quality of types in DBpedia and assesses the suitability of or approach for completing types for entities without any type in DBpedia.

Section 8.7 justifies the choice of linear SVMs as the base learner comparing performance with other common classification algorithms. Section 8.8 evaluates the effect of varying the *TRADEOFF* parameter of the STI algorithm.

8.1. Building the gold standard

Since the task of assigning a final type to the entity described in English Wikipedia article does not necessarily need an expert we rely on collecting judgments from paid volunteer contributors via a *crowdsourcing* service.

We decided to perform crowdsourcing as opposed to expert annotation based on experimental evidence presented in a seminal article of Snow et al. [23] that evaluates the quality of crowdsourced annotations on five different natural language processing tasks. For all five task types the paper reports high agreement between Amazon Mechanical Turk non-expert annotations and expert labelers.



Wikipedia article: http://en.wikipedia.org/wiki/Marja_van_der_Tas (click to open!).

Fig. 3. Interface of the CrowdFlower taxonomy annotation tool. The annotators can navigate through the taxonomy either by clicking on a concept, which shows its subtypes, or by fulltext search, which shows all concepts with substring match in the concept name along with the full path.

8.1.1. Task setup

For the crowd sourcing service we opted for *CrowdFlower*[7] as Amazon Mechanical Turk is not available for Europe. The annotation instructions asked the CrowdFlower workers to assign *the most specific category* (categories) from the presented *taxonomy of categories* for each Wikipedia article describing certain entity from the given list. The taxonomy used corresponded to the DBpedia 2014 ontology, which contains almost 700 DBpedia types.[8] The annotators were aided in the task of locating the right class among the 700 candidates by the taxonomy annotation tool offered by the CrowdFlower platform, which enables the annotators to quickly browse through the taxonomy using fulltext queries. Fig. 3 shows a screenshot of the tool.

It should be noted that it was up to the annotators to choose which part of Wikipedia articles they will read and identify types from, however, many of them might have opted only for reading the start of the article. This could have slightly favored our SVM algorithm trained on short abstracts, and the evaluation of the LHD Core, which is based on the lexico-syntactic analysis of the article's first sentence.

The CrowdFlower platform has a wide range of setting for controlling the quality of the work done by its workers. Our setup was as follows:

- Only workers residing in the following countries were eligible: Australia, Canada, Denmark, Germany, Ireland, Netherlands, Sweden, United Kingdom and United States. The workers were Level 1 Contributors, which are described by the CrowdFlower service as accounting for 60% of monthly judgments and maintaining a high level of accuracy across a basket of jobs.
- Amount of 0.02 USD was paid for each annotated entity to a worker.
- The workers were given a quiz before starting a task with minimum of four test questions (entities to annotate). Only workers with accuracy of 30% or higher could continue in the task.
- To maintain high accuracy, additional test questions were asked as the workers were completing their job.
- A speed trap was put in place that eliminated workers who took less than 10 s to complete a task.

Concerning the appropriateness of the remuneration, [24] gives half-a-penny per question as the rule of thumb for payment on crowd sourcing services, which our remuneration exceeded. To further ensure that the pay is appropriate, we checked the satisfaction scores reported in the final questionnaire by the

[6] That is leaf types with parent Thing. We obtained better results when these were excluded.

[7] http://www.crowdflower.com/.

[8] Since CrowdFlower only allows one super-category for each category in a taxonomy, we did one correction: *Library* is originally subsumed by both *EducationalInstitution* and *Building* in DBpedia Ontology, for the taxonomy we only kept subsumption to *EducationalInstitution*.

workers. On a 1–5 Likert scale (1 worst, 5 is best), the workers rated their remuneration on average between 3.1 to 4.0. None of the jobs had pay rating in the red band.[9]

Each entity was typically annotated by three to four workers. The CrowdFlower platform ensured that the annotations from workers who failed the test questions were replaced by untainted annotations.

Our setup can be somewhat compared the crowdsourcing evaluation performed in [9]. There the number of workers annotating each entity was similar to ours (three). Also, similarly to our setup, the workers were supposed to select only one best type. One major difference is that in [9] the workers were presented preselected types (with the option to enter a new type), while in our system they had to select the type from a larger fixed list of types. Another difference is that in [9] no majority type was selected for given entity. Instead, all types were used with a relevance score corresponding to the number of workers selecting the respective type.

8.1.2. Interannotator agreement

For measuring interannotator agreement we have opted for Krippendorff's alpha [25] (as implemented in [26]), since this measure supports multiple annotators and is applicable to incomplete data. The values of Krippendorff's alpha as reported in Table 5 are in the 0.4–0.6 range which is considered as moderate agreement for kappa-like coefficients ([27] cited according to [28]). While some sources would consider already value below 0.8 as unacceptable for any serious purpose [25, Chapter 11, page 242], it should be noted that our annotation task with hundreds of distinct concepts to choose from was exceptionally difficult. Also, when computing the α we used binary distance function (i.e. the similarity of two distinct yet semantically close annotations was not considered). Annotations assigning more than one concept were ignored for the purpose of computing the α value.

8.1.3. Gold standard datasets

The gold standard for given entity consists of all types that were assigned by at least two annotators to the entity. As a consequence, not all entities included in the annotation task are contained in the gold standard (cf. Table 5). The process of establishing the gold standard is illustrated by Example 9.

Example 9.
Wikipedia article describing *August Nybergh* entity was annotated in the following way by four annotators:

- {Agent > Person > Politician > Senator}, {Agent > Person > Politician > MemberOfParliament}
- {PersonFunction > PoliticalFunction}
- {Agent > Person > Politician > Senator}, {Agent > Person > Politician}
- {Agent > Person > Politician}

The first and the third annotator assigned two different most specific types. The final most specific type, having frequency at least two, is the *Senator* type. The Politician type was not added to the gold standard as it is a superclass of Senator.

Any redundant superclasses were removed as also illustrated by the example. The annotators could assign more than one most specific type to the entity. Multiple final types were assigned for less than 1% of entities in our initial annotation task, thus we ignored multiple types in our evaluation, selecting one type randomly in such cases for the gold standard. Besides categories corresponding to types in the DBpedia Ontology, annotators could select 'not found' category if they could not find the article or 'disambiguation page' category in case the article was a disambiguation page in their opinion. Entities with these categories are omitted from the gold standard. In order to foster reusability of the dataset as the evaluation ontology we used the most up-to-date released version of the DBpedia Ontology (2014) at the time.

The gold standard resulting from the annotation process is composed of three datasets depending on the subset of DBpedia/Wikipedia from which the entities to be annotated were drawn. Table 5 shows an overview of the three gold standard datasets, totaling 2214 entities with groundtruth.

8.2. Evaluation metrics

We use four evaluation measures: exact precision, hierarchical precision, hierarchical recall and hierarchical F-measure. The first measure corresponds to precision which does not take into account the type hierarchy:

$$P_{exact} = \frac{\sum_i |P_i \cap T_i|}{\sum_i |P_i|}, \tag{1}$$

where P_i is the set of the most specific types predicted for test example i, T_i is the set of the true most specific type of test example i.[10]

The other three measures consider the type hierarchy. Hierarchical precision (hP), hierarchical recall (hR) and hierarchical F-measure (hF) are defined according to [29] as follows:

$$hP = \frac{\sum_i |\hat{P}_i \cap \hat{T}_i|}{\sum_i |\hat{P}_i|}, \tag{2}$$

$$hR = \frac{\sum_i |\hat{P}_i \cap \hat{T}_i|}{\sum_i |\hat{T}_i|}, \tag{3}$$

$$hF = \frac{2 * hP * hR}{hP + hR}, \tag{4}$$

where $\hat{P}_i$ is the set of the most specific type(s) predicted for test example i and all its (their) ancestor types and $\hat{T}_i$ is the set of the true most specific type(s) of test example i and all its (their) ancestor types.

8.3. Evaluated setups

Our evaluation involves the following setups of our algorithms:

- *LHD Core*: lexico-syntactic patterns, extracted types were successfully mapped to DBpedia Ontology with exact string matching (LHD Core, approach published in [2]).
- *STI$_{prune}$*: lexico-syntactic patterns, type mapping was performed by the standalone STI with pruning (exact string matching failed).
- *hSVM$_{cat}$*: hierarchy of SVM models trained on article categories with the final types selected with Multiplicative Scoring Rule (MSR).
- *hSVM$_{abstract}$*: hierarchy of SVM models trained on article abstracts with the final types selected with MSR.

[9] The crowdflower platform assigns three color codes to the final scores (red, orange and green) to help interpreting the questionnaire results.

[10] We measure P_{exact} only for algorithms that assign at most one type (P_i and T_i always contain at most one element).

Table 5
Overview of evaluation dataset. Column *entities* denotes the number of entities in the annotation task (*all*), number of entities where annotators agreed on ‘not found’ category (*nf*), number of entities where annotators agreed on ‘disambiguation page’ category (*dp*), number of entities where annotators did not agree based on majority vote (*nma*), number of entities with ground truth (*gt*) and the number of the “hard” entities—those with groundtruth for which there is no type in DBpedia (gt_h). Interannotator agreement is reported in terms of Krippendorff’s alpha. Column *workers* reports the number of unique annotators. LHD Fusion 3.9 denotes the set of entities in DBpedia 3.9 for which a hypernym was extracted but not mapped with exact string matching to DBpedia Ontology, cf. Fig. 1.

Dataset	Entities						Kr. α	Workers	Sample source
	all	nf	dp	nma	*gt*	gt_h			
GS 1	1219	140	5	53	1021	373	0.529	64	LHD Fusion 3.9
GS 2	176	2	2	12	160	NA	0.514	16	Intersection of SDType 3.9 and LHD Fusion 3.9
GS 3	1165	22	47	63	1033	331	0.503	48	Randomly drawn articles from Wikipedia

- $hSVM_{text}$: $hSVM_{cat}$ and $hSVM_{abstract}$ merged with linear opinion pool using equal weights.
- $hSVM_{text}STI$: all three models ($hSVM_{cat}$, $hSVM_{abstract}$, STI without pruning) were merged with linear opinion pool, the final types were selected with MSR.
- $hSVM^{add}_{text}STI$: all three model results were merged with linear opinion pool, the final types were selected with ASR.
- $Core + STI_{prune}$: merge of results of LHD Core and STI_{prune}.
- $STI_{prune} + hSVM_{text}$: merge of results of STI_{prune} and $hSVM_{text}$ where results of STI are prioritized (if an entity has types assigned both in STI and $hSVM_{text}$, only results from STI are used).
- $Core + hSVM_{text}$ *STI*: merge of results of LHD Core and $hSVM_{text}$ *STI* where results of LHD Core are prioritized.
- $Core + STI_{prune} + hSVM_{text}$: merge of results of LHD Core, STI_{prune} and $hSVM_{text}$ where results of LHD Core and STI are prioritized.

The results of LHD Core and STI were generated by the LHD framework [12] and are available as part of the DBpedia 2014 release. The tradeoff threshold constant of STI was set to 0.6, which is a value that maximizes F-measure on GS 1 (refer to Section 8.8). Note that this threshold is used only in the standalone STI runs. Based on parameter tuning, the STI weight for linear opinion pool was set to 0.33.

All SVM models were also generated on DBpedia 2014. Threshold for MSR or ASR algorithms for combining SVM models was selected according to the maximum hF-measure based on evaluation on a different dataset. That is, for GS1 dataset we used the best hF-measure computed on GS3 and vice versa. The optimization step was 0.01.

For reference purposes, our evaluation also involves the following:

- *SDType*: SDType results for DBpedia 3.9 obtained from the DBpedia website.[11]
- *DBpedia 2014*. Entity type assignments in the DBpedia ontology namespace that are part of the English DBpedia 2014 release.

The evaluations are performed in addition to GS1, GS2, and GS3 also on GS3 subset GS3h that contains the “hard” entities—those with no type assigned in DBpedia 2014.

8.4. STI, hSVM and their combinations

We evaluated separately the STI and hSVM classifier and their combination using Multiplicative Scoring Rule (MSR) and its ASR variant. The results are presented in Table 6.

Table 6
Evaluation on gold standard GS1 (1021 entities) and GS2 (160 entities).

Classifier	P_{exact}	*hP*	*hR*	*hF*
STI_{prune}	.446	.780	.589	.671
$hSVM_{abstract}$	NA	.622	.550	.584
$hSVM_{cat}$	NA	.587	.644	.614
$hSVM_{text}$	NA	.713	.668	.690
$hSVM_{abstract}\alpha$	.261	.622	.597	.609
$hSVM_{cat}\alpha$	.267	.715	.611	.659
$hSVM_{text}\alpha$	.310	.719	.675	.696
$hSVM_{text}STI\alpha$	.347	.735	.730	.732
$STI + hSVM_{text}\ \alpha$	.400	.763	.734	.748
$hSVM^{add}_{text}\beta$	.365	.719	.706	.712
$hSVM^{add}_{text}STI\beta$	.294	.817	.652	.726
DBpedia (*2014*)	.548	.890	.665	.761
GS2				
SDType (3.9)	.338	.809	.641	.715

With respect to our individual approaches, STI outperforms all runs of the hSVM classifier including its combination with STI in the P_{exact} measure, while hSVM has better results with regard to the hierarchical measures. The good STI result might be to certain extent influenced by existing type assignment in DBpedia, since the STI classifier exploits the co-occurrence information with types already in DBpedia. Also GS1 dataset was used to tune the TRADEOFF threshold affecting the results of STI_{prune}. An unbiased evaluation on GS3h shows that indeed the hierarchical precision of STI drops below hierarchical SVM on this dataset.

With respect to the hSVM classifier, the improvement in all metrics for $hSVM_{text}$, which uses both abstract and categories as input features, suggests that these sets of features are not redundant. What we have not evaluated is if a hSVM model built upon a merge of both feature sets would not provide even better results than building two models and merging them. Individually, the classifiers built upon the *categories* feature set perform slightly better than the ones built upon *abstracts*, but this difference is not statistically significant as the 95% Wilson confidence intervals for binomial probabilities for exact match overlap.[12]

The comparison between the baseline MSR approach $hSVM_{text}$ $STI\alpha$ and our additive variant $hSVM^{add}_{text}STI\beta$ shows that the additive version provides an improvement in hierarchical precision, but this is offset by even higher drop in the remaining metrics.

Selecting one final type with either α or β strategies is better in terms of all metrics than the vanilla MSR approach $hSVM_{text}$, which uses all types with joint probability exceeding the threshold.[13] Since selecting one type per entity is preferred (DBpedia infobox-based framework and STI also assign one type) we therefore select $hSVM_{text} + STI\alpha$ as the final approach. This corresponds to merge of the results of STI and hSVM algorithms rather than their fusion with linear opinion pool.

[11] The reason why we use 3.9 and not 2014 results is that the GS2 dataset designed for comparison of SDType results with our approach was generated on version 3.9. Since SDtype result for version 2014 does not contain many of these entities, the evaluation sample would be too small.

[12] Paper [30] suggests that when interval overlap is used for significance testing, 95% confidence interval will give very conservative results.

[13] A noteworthy comparison is that the α and β strategies, which select one final type, have higher recall than vanilla MSR, which selects all types above the threshold. The reason is that the threshold weights were trained separately for all three approaches.

Table 7
Evaluation on gold standard GS3 (1033 entities) and GS3h (331 entities), 50 entities from GS3 and GS3h are not present in DBpedia 2014.

Classifier	GS3 (randomly drawn articles)					GS3h (untyped instances)				
	entities	P_{exact}	hP	hR	hF	entities	P_{exact}	hP	hR	hF
DBpedia	715	.537	$\underline{.902}$	.611	.729					
SDType						19	.105	.644	.033	.063
Core	402	$\underline{.654}$	.864	.371	.519	26	$\underline{.654}$	$\underline{.713}$	.065	.119
STI_{prune}	379	.449	.754	.274	.403	106	.255	.461	.162	.240
$hSVM_{text}\alpha$	750	.307	.747	.597	.663	131	.130	.635	.293	.400
$hSVM_{text}STI\alpha$	765	.327	.757	.621	.682					
$Core + STI_{prune}$	781	.554	.814	.645	.720					
$Core + hSVM_{text}\ STI\alpha$	864	.439	.786	.720	.752	169	.169	.534	.289	.375
$Core + STI_{prune} + hSVM_{text}\alpha$	$\underline{896}$	.465	.800	$\underline{.724}$	$\underline{.760}$	$\underline{197}$	.205	.565	$\underline{.379}$	$\underline{.454}$

Overall, the hSVM approach can be used to assign type to entities unmatched by the lexico-syntactic patterns, but it does not improve – at least with the current version of the linear opinion pool fusion approach – the existing type assignments generated by the STI algorithm.

8.5. SDType

This section compares our approach to the state-of-the-art algorithm SDType described in Section 2.5.

We evaluated SDType on gold standard dataset *GS*2, which covers untyped instances in DBpedia 3.9 that were assigned a type with SDType. The evaluation statistics are provided in the bottom of Table 6. Results on GS1 show that on this sample SDType is very reliable in selecting types with hierarchical precision very close to that of DBpedia. Hierarchical recall and *F*-measure have little meaning on GS1 for SDType since a criterion for selecting GS1 entities was the presence of a type assigned with SDType.

Our second evaluation was performed on GS3h containing randomly drawn articles from English Wikipedia that are untyped in DBpedia 2014. The hierarchical *F*-measure and the number of covered entities show that SDType assigned a type only to a very small number of instances compared to all other approaches. When SDType did assign the type, the hierarchical precision was on par with hSVM. Inspection of P_{exact} on GS2 and GS3h evaluation shows that the specificity of types assigned by SDType is relatively low.

Overall, SDType completes a high number of untyped instances, but these are often instances without any Wikipedia page that were possibly created in DBpedia from Wikipedia "red links". In contrast, our algorithms require at least the abstract of categories to be present. Overall, this shows that SDType and our approach are highly complementary.

8.6. DBpedia

The entities in the gold standard GS3 were randomly selected from all the Wikipedia articles. The evaluation using GS3 thus provides the most objective evaluation of all approaches for type assignment.

For DBpedia type assignment to given entity we consider only the most specific DBpedia Ontology types, which is in-line with how our gold standard is constructed. First, we obtained all DBpedia Ontology types for given entity and next we selected the most specific types.[14]

Overall, DBpedia has the best hierarchical precision. However, the results, presented in Table 7, perhaps surprisingly show that the lexico-syntactic patterns (LHD Core) provide exact types with higher precision than DBpedia (22% relative improvement in P_{exact}). We hypothesize that this is caused by some infoboxes being mapped in the DBpedia extraction framework to higher-level types than is the most specific available type in the DBpedia ontology. This interpretation is supported by DBpedia having marginally higher hierarchical precision than LHD Core. Another possible reason contributing to LHD Core having higher exact precision than DBpedia is that it was easiest for annotators to base their type assignment on the first sentence of the article from which the LHD patterns extract the type.

The results on GS3 show that all our approaches combined achieve higher hierarchical *F*-measure and assign types to more entities than the DBpedia infobox-based DBpedia extraction framework. The GS3 dataset contains 331 entities untyped in DBpedia (out of which 50 do not exist in DBpedia 2014 at all).[15] Out of these entities composing the GS3h dataset, our combined approach is able to assign types to 197 entities (which is 70% of untyped instances existing in DBpedia).

There are two main reasons why our most universal hSVM approach was unable to type the remaining 30% of untyped instances: part of these instances did not have any abstract and categories in DBpedia and for some instances the type assignment was computed, but was not considered reliable enough given the precomputed threshold in Algorithm 6.

8.7. Comparison with other classifiers

In order to further ground (beyond the related work discussed in Section 2.6) the selection of SVMs with linear kernel as our base model, we performed a benchmark on all 58 datasets, which were used to train the individual SVM classifiers. Ten percent of each dataset was used for testing, the rest for training (stratified selection). The feature set was pruned by removing features with less than 0.1 standard deviation in each dataset.

No parameter tuning for any of the classifiers was performed, the default values from the RapidMiner 5 implementation[16] of the respective classifier was used:

- **Ripper** [31]: information gain criterion used, sample ratio $=$ 0.9, pureness $=$ 0.9, minimal prune benefit $=$ 0.25.
- **SVM linear kernel**: $C = 0.0$, $\epsilon = 0.001$, shrinking applied.
- **SVM RBF kernel**: $C = 0.0$, $\epsilon = 0.001$, $\gamma = 0.0$, shrinking applied.
- **SVM polynomial kernel**: degree 3, $\epsilon = 0.001$, $C = 0.0$, $\gamma = 0.0$, shrinking applied.
- **Logistic regression**: dot kernel used, convergence $\epsilon = 0.001$, $C = 1.0$, value scaling applied.

[14] Out of 1021 DBpedia entities there was not any case with more than one specific type from DBpedia ontology namespace.

[15] Based on the *titles file*.

[16] http://rapidminer.sourceforge.net.

Table 8
Comparison of linear SVMs with other common classifiers.

Metric	Naive B.	SVM (linear)	SVM (RBF)	SVM (poly)	Ripper	Log Reg
Macro avg accuracy	0.76	**0.90**	0.88	0.85	0.80	0.86
Run time	less 1 min	5 min	6 min	12 min	4 h	5 min

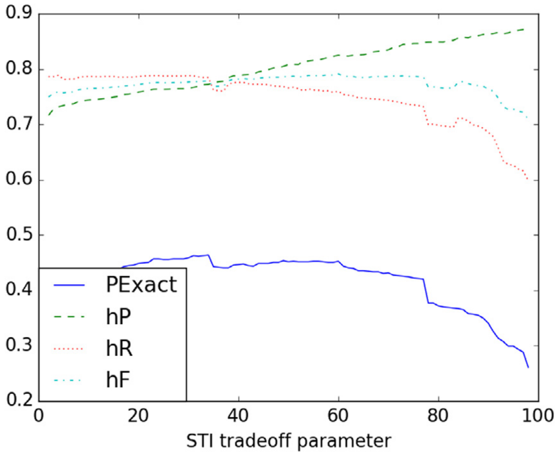


Fig. 4. Effect of tradeoff threshold.

The results depicted in Table 8 show that SVMs with linear kernels provide the best accuracy and at the same time have one of the smallest run times (aggregate for training and testing phase) on a core i5 2.6 GHz laptop with 16 GB of available memory running Open JDK 1.7. Our results are consistent with linear kernel being chosen for hierarchical classification of web content in Dumais and Chen [17] and by Liu et al. [15].

8.8. STI: tuning the tradeoff parameter

We performed parameter tuning of the STI algorithm's *tradeoff* constant on GS1 dataset and DBpedia 3.9. We executed the algorithm with *tradeoff* set to values ranging from 0.02 to 0.99 with step 0.01.

Fig. 4 shows that increasing value of this parameter improves hierarchical precision, which follows from more high level types surviving pruning. For the same reason, hierarchical recall drops as less types survive pruning. As a result, the hierarchical F-measure remains stable until around 0.8, with maximum having at tradeoff $=$ 0.6. Focusing on exact match, the best interval for the tradeoff parameter value lies between 0.2 and 0.6.

Based on this examination, we suggest to set the value of the tradeoff parameter to 0.6.

9. Conclusion and future work

This article introduced a novel technique for inferring entity types in semantic knowledge graphs. The free text describing the entities is analyzed using algorithms from the two major directions of computational linguistics: lexico-syntactic analysis and statistical natural language processing.

The types extracted with lexico-syntactic patterns are processed with an unsupervised Statistical Type Inference (STI) algorithm, which analyzes their co-occurrence with types already assigned in the knowledge graph. Further, we adapted the hierarchical Support Vector Machines (hSVMs) classifier, which we found particularly suitable due to the fact that our problem consists of a high number of taxonomically ordered classes.

During the course of the research, we were unable to find any resource that could be used for the evaluation of our algorithms providing an unbiased comparison with the accuracy of the DBpedia extraction framework. In response to this, we designed a new dataset using the commercial CrowdFlower crowdsourcing platform, which consists of more than 2.000 Wikipedia articles (DBpedia entities) that are assigned a type from the DBpedia 2014 Ontology. This dataset was made freely available along with the annotation guidelines under a Creative Commons license.

We evaluated the STI and hSVM algorithms and their fusion on the crowdsourced content and provide a comparison with DBpedia and its heuristics dataset, generated by the state-of-the-art SDType algorithm.

According to this evaluation we concluded that (1) the quality of types assigned with lexico-syntactic patterns from first sentence of Wikipedia articles is comparable to the quality of types inferred from information boxes by the DBpedia extraction framework, (2) the text categorization approach (hierarchical SVM) applied to the type inference problem has the highest recall of all but also the lowest precision (3) our approach has precision comparable to the state-of-the-art SDType algorithm while generating types for a largely different set of instances.

Notably, the hSVM approach requires as input only a free-text representation of the Wikipedia articles. Even the labeled data required to train the classifier for a particular language (i.e. DBpedia ontology types for at least some instances for each target class) can be obtained from Wikipedia's interlanguage links. The hSVM approach thus could serve as a starting point for populating type assignments in Wikipedia-based knowledge graphs for "smaller" languages or those with less development resources available.

As a future work, accuracy improvements could be gained by utilizing more sophisticated feature representation of the textual modality. A more involved enhancements would be replacement of the linear opinion pool with some meta machine learning approach such as *stacking*.

Resources for this article including the Inference dataset and the gold standard datasets are located at http://ner.vse.cz/datasets/linkedhypernyms/.

Acknowledgments

We wish to thank the three anonymous reviewers for their very helpful comments. Tomáš Kliegr thanks André Melo and Heiko Paulheim for insightful comments that led to our adoption of the standard set of hierarchical evaluation metrics and the German DAAD grant agency for making this interaction possible. The development of the STI and hSVM algorithms was supported by the European Union's 7th Framework Programme via the LinkedTV project (FP7-287911). Ondřej Zamazal has been additionally supported by the CSF grant no. 14-14076P, "COSOL—Categorization of Ontologies in Support of Ontology Life Cycle". This research is also partially supported by long term institutional support of research activities by Faculty of Informatics and Statistics, University of Economics, Prague.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.websem.2016.05.001.

References

[1] H. Paulheim, C. Bizer, Improving the quality of linked data using statistical distributions, Int. J. Semant. Web Inf. Syst. (IJSWIS) 10 (2014) 63–86.

[2] T. Kliegr, Linked hypernyms: Enriching DBpedia with targeted hypernym discovery, Web Semant. 31 (2015) 59–69.

[3] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia-a crystallization point for the web of data, Web Semant.: Sci. Serv. Agents World Wide Web 7 (2009) 154–165.

[4] J. Hoffart, F.M. Suchanek, K. Berberich, G. Weikum, YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia, Artificial Intelligence 194 (2013) 28–61.

[5] T. Kliegr, O. Zamazal, Towards linked hypernyms dataset 2.0: complementing DBpedia with hypernym discovery in: Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC-2014, Reykjavik, Iceland, May 26–31, 2014, pp. 3517–3523.

[6] H. Paulheim, Knowledge graph refinement: A survey of approaches and evaluation methods, Semantic Web (2016) 1–20. Preprint.

[7] A. Gangemi, A.G. Nuzzolese, V. Presutti, F. Draicchio, A. Musetti, P. Ciancarini, Automatic typing of DBpedia entities, in: P. Cudre-Mauroux, J. Heflin, E. Sirin, T. Tudorache, J. Euzenat, M. Hauswirth, J.X. Parreira, J. Hendler, G. Schreiber, A. Bernstein, E. Blomqvist (Eds.), The Semantic Web—ISWC 2012, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2012, pp. 65–81.

[8] H. Paulheim, C. Bizer, Type inference on noisy RDF data, in: The Semantic Web—ISWC 2013, Springer, 2013, pp. 510–525.

[9] A. Tonon, M. Catasta, G. Demartini, P. Cudré-Mauroux, K. Aberer, *TRank*: Ranking entity types using the web of data, in: The Semantic Web—ISWC 2013: 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21–25, Proceedings, Part I, Springer, Berlin, Heidelberg, 2013, pp. 640–656.

[10] H. Paulheim, Browsing Linked Open Data with auto complete, in: Proceedings of the Semantic Web Challenge co-located with ISWC2012, Springer, Boston, US, 2012.

[11] N.F. Noy, D.L. McGuinness, et al., Ontology Development 101: A Guide to Creating Your First Ontology, Technical Report, 2001.

[12] T. Kliegr, V. Zeman, M. Dojchinovski, Linked hypernyms dataset—generation framework and use cases, in: The 3rd Workshop on Linked Data in Linguistics: Multilingual Knowledge Resources and Natural Language Processing, co-located with LREC 2014, LDL-2014.

[13] J. Neville, D. Jensen, Iterative classification in relational data, in: Proc. AAAI-2000 Workshop on Learning Statistical Models from Relational Data, pp. 13–20.

[14] J. Sleeman, T. Finin, Type prediction for efficient coreference resolution in heterogeneous semantic graphs, in: Semantic Computing, ICSC, 2013 IEEE Seventh International Conference on, IEEE, 2013, pp. 78–85.

[15] T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, W.-Y. Ma, Support vector machines classification with a very large-scale taxonomy, SIGKDD Explor. Newsl. 7 (2005) 36–43.

[16] C. Cortes, V. Vapnik, Support-vector networks, Mach. Learn. 20 (1995) 273–297.

[17] S. Dumais, H. Chen, Hierarchical classification of web content, in: Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'00, ACM, New York, NY, USA, 2000, pp. 256–263.

[18] A. Zaveri, D. Kontokostas, M.A. Sherif, L. Bühmann, M. Morsey, S. Auer, J. Lehmann, User-driven quality evaluation of DBpedia, in: Proceedings of the 9th International Conference on Semantic Systems, I-SEMANTICS'13, ACM, New York, NY, USA, 2013, pp. 97–104.

[19] R.A. Howard, The foundations of decision analysis, in: W. Edwards, Ralph F. Miles Jr., D. von Winterfeldt (Eds.), Advances in Decision Analysis, Cambridge University Press, 2007, pp. 32–56. Cambridge Books Online.

[20] Wikipedia, Wikipedia:manual of style/lead section, 2006. (Online; accessed 24.03.16).

[21] T. Kliegr, K. Chandramouli, J. Nemrava, V. Svátek, E. Izquierdo, Wikipedia as the premiere source for targeted hypernym discovery, in: Proceedings of the Wiki's, Blogs and Bookmarking tools—Mining the Web 2.0 Workshop at ECML'08.

[22] M. Dojchinovski, T. Kliegr, I. Lašek, O. Zamazal, Wikipedia search as effective entity linking algorithm, in: Text Analysis Conference, TAC, 2013 Proceedings, NIST, 2013.

[23] R. Snow, B. O'Connor, D. Jurafsky, A.Y. Ng, Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP'08, Association for Computational Linguistics, Stroudsburg, PA, USA, 2008, pp. 254–263.

[24] T. Schnoebelen, V. Kuperman, Using Amazon mechanical turk for linguistic research, Psihologija 43 (2010) 441–464.

[25] K. Krippendorff, Content Analysis: An Introduction to Its Methodology, second ed., Sage, 2004.

[26] M. Gamer, J. Lemon, I.F.P. Singhf, irr: Various Coefficients of Interrater Reliability and Agreement, 2012. R package version 0.84.

[27] G.G.K.J. Richard Landis, The measurement of observer agreement for categorical data, Biometrics 33 (1977) 159–174.

[28] R. Artstein, M. Poesio, Inter-coder agreement for computational linguistics, Comput. Linguist. 34 (2008) 555–596.

[29] C.N. Silla Jr., A.A. Freitas, A survey of hierarchical classification across different application domains, Data Min. Knowl. Discov. 22 (2011) 31–72.

[30] M.E. Payton, M.H. Greenstone, N. Schenker, Overlapping confidence intervals or standard error intervals: what do they mean in terms of statistical significance? J. Insect Sci. 3 (2003) 34.

[31] W.W. Cohen, Fast effective rule induction, in: Proceedings of the Twelfth International Conference on Machine Learning, pp. 115–123.

Appendix C: Antonyms are similar: paradigmatic association approach to rating similarity in SimLex-999 and WordSim- 353

J3 Tomáš Kliegr, Ondřej Zamazal, Antonyms are similar: Towards paradigmatic association approach to rating similarity in SimLex-999 and WordSim-353, Data & Knowledge Engineering, 2018, In press , ISSN 0169-023X, https://doi.org/10.1016/j.datak.2018.03.004.

Data & Knowledge Engineering 115 (2018) 174–193

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

Antonyms are similar: Towards paradigmatic association approach to rating similarity in SimLex-999 and WordSim-353

Tomáš Kliegr [*], Ondřej Zamazal

Department of Information and Knowledge Engineering, University of Economics, Winston Churchill sq. 4, Prague 3, 130 67, Czech Republic

ARTICLE INFO

Keywords:
Word similarity
Word relatedness
WordSim353
SimLex-999

ABSTRACT

SimLex-999 is a widely used lexical resource for tracking progress in word similarity computation. It anchors similarity in synonymy, while other researchers such as Agirre et al. (2009) adopt broader similarity definition, involving also hyponymy and antonymy relations. Paradigmatic association covers synonymy, antonymy and co-hyponymy relations (Lapesa et al., 2014) largely overlapping with this broader similarity definition. Two words are paradigmatically associated if they can replace one another without affecting the grammaticality or acceptability of the sentence. Paradigmatic association can be elicited by asking for word interchangeability, which we hypothesize might be more natural than instructing raters with a list of relations to consider. To validate the proposed approach, we reannotated WordSim353 and SimLex-999 using two new guidelines: one explicitly qualifying antonymy as a similarity relation, the second one eliciting word interchangeability. As additional datasets we present a crowdsourced version of WordSim353 and a Czech version of SimLex-999. The paper also includes detailed analysis of lexical content of SimLex-999 and benchmark of thesaurus-based and distributional algorithms on multiple word similarity and relatedness datasets.

1. Introduction

Current state-of-the-art algorithms for representing meaning of words in text are based on the distributional hypothesis, which postulates that if two words co-occur in the same context, they tend to have similar meanings [1]. However, later research showed that co-occurrence in text applies both to *related* words (“coffee” and “cup”) as well as to words that have truly *similar* meanings (“cup” and “mug”). Being able to distinguish similarity from relatedness is important to a number of applications which aim to intelligently understand text. One emerging application area is the *entity classification* problem, which aims at assigning entities (words or noun chunks) to one category from a given list. An example task is to decide whether “cup” is a “drink” or “container” in given context. This may present a difficult choice for distributional algorithms, since word “cup” more often co-occurs with “drink”.

In order to select the best algorithms for the word similarity task, it is necessary to have suitable benchmarking resources, which distinguish similarity from other word relations. The WordSim353 dataset was contributed for measuring semantic similarity in Finkelstein et al. [2] and has been since used as the gold standard for tracking progress in the field of word similarity and relatedness computation. However, the current consensus is that the dataset is – in spite of its name – designed to measure *semantic relatedness*. The recently proposed SimLex-999 dataset [3] filled this gap in the computational linguistics research: it is both larger than WordSim353 and it explicitly quantifies similarity.

[*] Corresponding author.

E-mail addresses: tomas.kliegr@vse.cz (T. Kliegr), ondrej.zamazal@vse.cz (O. Zamazal).

https://doi.org/10.1016/j.datak.2018.03.004
Received 14 March 2017; Received in revised form 23 November 2017; Accepted 18 March 2018
Available online 11 April 2018

In this article, we perform a critical analysis as of the lexical content as well as the similarity definition reflected in the SimLex-999 guidelines. While we found that this dataset is well-designed, its several design choices may not fit all use cases. Most importantly, its guidelines anchor similarity definition solely in synonymy, which implies antonyms being annotated as dissimilar. According to a wide-spread notion in computational linguistics [4–7] as well as in cognitive science [8], antonyms are similar in all but one aspect, in which they are maximally opposed [9,10].

As an alternative approach, we propose to adopt the paradigmatic association (similarity) concept used in psychology [11] when eliciting similarity judgments. This should support high similarity scores not only for synonyms, but also for antonyms. Regarding the lexical composition of our dataset, we argue not to leave WordSim353 in favour of the larger SimLex-999. WordSim353 has long been used to track progress in the field and it has lexical composition complementary to SimLex-999. It contains relatively less direct synonyms, antonyms and informal words than SimLex-999. Direct synonyms and antonyms can make similarity computation easier for WordNet-based similarity methods, while informal words may complicate application of distributional approaches when the training corpus containing informal words is unavailable. Also, the reannotation of WordSim353 with explicit similarity guidelines has been previously called for [4].

Recent research has shown that judgment language has influence on the human scores and that vector space models can benefit from multilingual annotations [12]. There is a growing list of language mutations of similarity and relatedness datasets: WordSim353 was translated to Czech, SimLex-999 to Russian, Italian and German. We complement these efforts by providing Czech versions of the existing WordSim353 and SimLex-999 datasets as well as of datasets newly proposed in this paper.

Finally, this article provides a benchmark of a range of word similarity and relatedness measures on multiple datasets. While these algorithms can be decoupled from the knowledge base they are used with, word similarity measures typically require a structured lexical resource with explicitly stated hyponym-hypernym relations between entries. WordNet is the most commonly used choice for similarity measures, while relatedness measures are typically distributional algorithms that have been crafted for Wikipedia. The purpose of this evaluation is to show which algorithms are good at which datasets. The analysis of the reasons can contribute to future algorithm and dataset design.

This article is organized as follows. Section 2 confronts the notions of word similarity, relatedness and association. WordSim353 and SimLex-999 datasets are described in Section 3. The English datasets proposed in this paper are described in Section 4. The Czech mutations are covered in Section 5. Section 6 describes the benchmark. Section 7 presents a digest of the scientific discussion on the use of crowdsourcing for gold standard design and an overview of related evaluation resources. This section also presents the comparison between the proposed guidelines and SimLex-999, including also the results of the lexical analysis. Conclusions provide a summary of the contributed datasets, present the main findings resulting from the benchmark and discuss the limitations of our work.

2. Similarity, relatedness or association?

In the field of distributional word models, the notion of similarity captures a wide range of semantic relations, such as synonymy, antonymy, hypernymy or even relatedness [6,13]. In this section, we first present a brief review of similarity definitions across computational linguistics, cognitive science and “traditional” linguistics, and then describe our choice, pointing at commonalities and differences with the definition used in the SimLex-999 dataset.

2.1. Brief multidisciplinary review of similarity definitions

The purpose of this section is to motivate the definition of similarity and relatedness used to construct our datasets, and to point at similarities and differences between our definitions and those used in psychology and cognitive science.

Computational linguistics: similarity vs relatedness. It is difficult to find two papers that define the terms semantic relatedness, similarity and association in the exactly same way. Nevertheless, the gist of the difference between semantic similarity and relatedness can be illustrated on the following example given by Resnik [14]:

> “Cars and gasoline would seem to be more closely related than, say, cars and bicycles, but the latter pair are certainly more similar.”

Formally, Agirre et al. [4] define *semantic similarity* through hyponym-hypernym, synonymy and antonymy relations, and relatedness through meronym-holonym and other types of relations.

Semantic relatedness is a commonly used term in many seminal papers in the area of artificial intelligence [4,15–18]. All this research uses the WordSim353 dataset, and except of Agirre et al. [4] all of them implicitly or explicitly accept that it measures relatedness. Agirre et al. [4] argue that similarity and relatedness were annotated without distinction in WordSim353. The relatedness definition used in WordSim353 (for guidelines cf. Appendix A) is, in our opinion, best matched by the definition provided by Budanitsky and Hirst [19]; who view relatedness as a more general concept than similarity:

> “similar entities are semantically related by virtue of their similarity (*bank-trust company*), but dissimilar entities may also be semantically related by lexical relationships such as meronymy (*car-wheel*) and antonymy (*hot-cold*), or just by any kind of functional relationship or frequent association (*pencil-paper, penguin-Antarctica, rain-flood*).”

Cognitive science: attributional, relational and literal similarity. According to Gentner and Markman [20] the general consensus in cognitive science is on similarity definition provided by Tversky [8] [1]:

> “pair’s similarity increases with its commonalities and decreases with its differences.”

A more recent research suggests that there are the following factors contributing to humans perceiving two objects as similar: *attributional similarity* and *relational similarity* [21]. According to Turney et al. [7]; the definition of semantic relatedness provided by Budanitsky and Hirst [19] within the scope of computational linguistics corresponds to attributional similarity in cognitive science [22]. The notions of relational similarity and attribute similarity can be used to explain *analogy*, which occurs when the two objects have high degree of relational similarity and very little attributional similarity [21]. *Literal similarity* in cognitive science requires that both relational predicates and object attributes are shared [21].

According to the behavioral experiments performed by Lund et al. [23]; similar words – as judged by human subjects – tend to appear in similar contexts:

> “Semantically similar word pairs are interchangeable within a sentence; the resulting sentences may be pragmatically improbable, but they are not nonsensical (…) Associated-only pairs tend to produce awkward sentences when interchanged, sentences that often cannot be taken literally.”

Other results by the same authors indicate that semantic similarity between words – as manifested by *word interchangeability in a sentence* – is instrumental for human subjects to consider two words as associated in terms of results of priming experiments.

Lund et al. [23] also found that being “associated” (that is co-occurring in a large text corpus) was not found to be sufficient to produce a priming in an experiment involving 64 human subjects. To this end, Hutchison [24] gives statistics showing that word pairs with the strongest association in word association norms[2] are linked with antonymy and synonymy relations. This is illustrated also by the following example, which is based on association norms used to create the SimLex-999 dataset.

Example.

The top seven word pairs from the Free association database of South Florida [25] according to cue-to-target strength: close synonyms (*trout-fish*, *shove-push*, *weep-cry*), antonymy (*left-right*, *in-out*), functional relationship (*moo-cow*) or frequent association (*cheddar-cheese*).

Linguistics: paradigmatic vs syntagmatic association. In the linguistic context, Saussure [26] (cited according to Rapp [11] defines two types of free word associations: syntagmatic associations and paradigmatic associations. There is a *syntagmatic* association between two words

> “if they co-occur in spoken or written language more frequently than expected from chance and if they have different grammatical roles in the sentence in which they occur.”

The association is *paradigmatic*

> “if the two words can substitute one another in a sentence without affecting the grammaticality or acceptability of the sentence.”

This distinction is embraced and supported by empirical results of Rapp [11]; Lapesa et al. [6] on free association datasets.

2.2. *Similarity as paradigmatic association*

The SimLex-999 dataset annotation guidelines effectively narrow definition of similarity to synonymy; cf. Section 7.3 for more detailed discussion of the guidelines. While Budanitsky and Hirst [19] also include antonymy under relatedness, more authors seem to include antonymy under similarity [4,6]. The general justification is that antonyms are similar in all respects but one, in which they are maximally opposed [9,10]. The psychological literature on priming [5,7] also includes antonymy under the similarity relation. As noted earlier, Agirre et al. [4] define similarity through synonymy, antonymy and hyponym-hypernym relations. Since assessing all these relations might be difficult for humans when they are asked to assign a similarity score, we propose to anchor definition of similarity in paradigmatic association. This according to Lapesa et al. [6] covers synonyms, antonyms and co-hyponym relations.

3. Base datasets

In this section, we briefly describe the WordSim353 dataset, which provides a basis for our reannotation effort. Our review includes also its WSSim and WSRel subsets. We also cover the SimLex-999 dataset, which provides the state-of-the-art word similarity

[1] This definition is used in the Pirro&Secco WordNet similarity metric, which is included in our evaluation.

[2] These are databases created by psychologists that associate an ordered word pair with association strength. The norms contain aggregated data from word priming experiments executed by psychologists, in which multiple subjects are given a cue word (such as *car*) and asked to respond with the first word that comes to their mind (such as *petrol*).

benchmark.

3.1. WordSim353

The most widely used benchmark dataset in the word similarity and relatedness areas is the freely available[3] WordSim353 dataset proposed in Finkelstein et al. [2]. This dataset consists of two sets of English word pairs containing 153 and 200 word pairs along with similarity judgments assigned by 13 and 16 human subjects respectively. The judgments range from 0 (totally unrelated words) to 10 (very much related or identical words). It was not disclosed how the word pairs for the dataset were selected.

Out of the 437 unique words in the dataset, 7 words cannot be directly mapped to a WordNet noun synset, for these words the mapping was created manually by selecting a replacement word. These seven words are featured in 9 word pairs, which is the same number as reported in Agirre et al. [4]. These mappings are with one exception straightforward: media → medium, children → child, live → living, Maradona → footballer, eat → eating, earning → earnings, defeating → defeat. We feature the dataset with the replacements made as WordSim353-WNAlign in our evaluations. Some of the changes in this dataset (e.g media → medium) shift the meaning substantially, and as a result these word pairs are not directly comparable to the ones from the original dataset. For this reason, our evaluation also reports on results for the original WordSim353 dataset.

There is a growing consensus that despite its name the WordSim353 dataset is not suitable for measuring word similarity. For example. Gabrilovich and Markovitch [15] [4] explicitly state that WordSim353 was designed to measure semantic relatedness, since its annotation guidelines specifically direct the raters "to assess the *degree of relatedness* of the words".

3.2. WordSim353#WSSim and WordSim353#WSRel

Agirre and Soroa [27] further partitioned WordSim353 into two gold standard datasets. The *similarity dataset* (WSSim) contains pairs of words considered as similar (synonyms, antonyms, identical, hyponym-hyperonym) and unrelated pairs (pairs with no clear relationship and with similarity equal or below a certain threshold). The *relatedness* dataset (WSRel) contains pairs considered as meronym-holonym, and those pairs in WordSim353 which have the "similarity" rating above certain threshold, but are not included in the WSSim subset. Additionally, the WSRel subset contains also the unrelated pairs. The number of pairs in the similarity dataset is 203 and the number of pairs in the relatedness dataset is 252.[5]

The WSSim dataset was criticized in Hill et al. [3]; as the fundamental limitation they give the fact that this dataset was annotated according to the same guidelines as WordSim353, for which the guidelines ask to annotate association rather than similarity. In Section 4 we introduce a new version of the WSSim dataset reannotated according to the similarity guidelines.

3.3. SimLex-999 and SimLex-666

The SimLex-999 dataset was proposed by Hill et al. [3] to provide a lexical resource for evaluation of word similarity computation methods. The dataset was created by sampling 999 pairs of words from the University of South Florida Free Association Database [25]. The word pairs in SimLex-999 were rated for synonymy using the Amazon Mechanical Turk crowdsourcing platform. The dataset can be decomposed to three subsets depending on Part of Speech (POS) tags of the participating words: 666 noun pairs, 111 adjective pairs and 222 verb pairs.

The annotation instructions focused on explaining the distinction between similarity and relatedness, featuring several examples. The raters were 500 residents in the USA with previous 95% approval rate for work on the service. Quality was further assured by removing a) annotations for raters who failed a checkpoint question and b) raters who had small agreement with other responses. Each pair in the final dataset is rated at least by 36 raters.

All words in SimLex-666 are directly mappable to WordNet, which can be probably attributed to the fact that the WordNet Wu&Palmer measure was possibly used to select pairs for the dataset.[6] As a consequence, the results of the WordNet measures on this dataset might be somewhat biased towards higher correlations.

4. WIN353 and WordSim353-crowd

As previously discussed, the main deficiency of WordSim353 are the vague guidelines. The primary goal of the reannotation effort is to generate a new version of the WordSim353 dataset using guidelines based on paradigmatic association. We also use this opportunity to reannotate the dataset according to the original guidelines in a crowdsourced environment.

[3] http://www.cs.technion.ac.il/~gabr/resources/data/wordsim353/.

[4] The first author of [15] is among the coauthors of the original publication introducing the dataset.

[5] The original WordSim353 dataset contains twice the pair of words 'money, cash', which was included only once in the relatedness dataset.

[6] Hill et al. [3] are not entirely clear about the role of Wu&Palmer measure in the dataset design process.

Table 1
Annotations by source country (aggregate counts for all three tasks).

dataset	DEU	FRA	SWE	FIN	IRL	CHE	ISR	USA	GBR	CAN	n/a
WS353-nat								1634	837	353	
WS353-notnat	1658	468	344	353							
ES353-nat								4143	1715	682	348
ES353-notnat	3458	696	696	1044							
WIN353-nat								1629	695	386	114
WIN353-notnat	1390	581	239	250	125	125	114				

4.1. Annotation guidelines

An important factor in our reannotation effort was the design of new annotation guidelines. We experimented with three versions, all of which are reprinted in the Appendix:

Original WordSim353: guidelines exactly correspond to the *original WordSim353* guidelines. These ask the raters to assign 0 if *words are totally unrelated* and 10 if *words are VERY closely related.* The guidelines have explicit instructions for antonyms – these are to be considered as *similar* as they are *belonging to the same domain or representing features of the same concept.*

Explicit Similarity: guidelines feature explicit definition of similarity: *the more similar the words are, the more in common the concepts behind the words have.* These guidelines explain the difference between similarity and relatedness based on the work presented in Agirre et al. [4] providing several examples. The guidelines also explicitly list relations that are included in similarity: synonymy, antonymy, hyperonymy, giving examples for each category.

Word INterchangeability (WIN): The proposed approach to eliciting paradigmatic association adopts the word interchangeability approach, which is well-understandable for the rater simple. This was hypothesized decrease the cognitive effort required and improve the quality of the resulting similarity scores.

All three versions of the guidelines are in the Appendix.

4.2. Annotation setup

Similarity to other recent annotation efforts [3,12] we relied on collecting judgments from paid "workers" (raters) via crowdsourcing. The crowdsourcing task was performed in two runs using the http://www.crowdflower.com/ also used by Leviant and Reichart [12]. The purpose of the first run was to calibrate the guidelines, determine the impact of the number of raters, and to study the performance of raters at different levels. While the result of the test run was not used to generate our final dataset, we used some of the answers with the highest agreement as test questions for the final run.

CrowdFlower offers three levels of workers, out of which we considered only the highest two levels, as measured by accuracy on previous tasks. We did not experience significant difference between the inter-rater agreement of L3 workers and L2 workers in the first test run.

After the initial testing of guidelines and crowdsourcing service, we established the final six annotations tasks (run 2) summarized in Table 2. There are two rating tasks for each revision of the guidelines depending on the country of the rater: Northern America and United Kingdom, and Western Europe incl. Ireland and Israel. The first group is largely English speaking, therefore we abbreviate it as *nat* (for Native). The level of English in countries in the second group is at least in the "Moderate proficiency" group according to the EF English Proficiency Index [28]. English is the official language in Ireland and it is regularly used in Israel. We abbreviate the second group as notnat. The distribution of rating across countries is shown in Table 1.

For both the explicit similarity and the WIN guidelines we also employed *test questions*. These were raised at the beginning of the annotation process. The raters failing the test questions were not eligible to continue in the task and were replaced by other raters.

4.3. Inter-rater agreement

Table 2 presents inter-rater agreement as the average of pairwise Spearman ρ correlations between the ratings of all raters, adopting the same methodology as described by Hill et al. [3].

We first computed ρ for each pair of raters with at least one shared rated word pair, and then averaged the results. In other

Table 2
Overview of rating tasks. Judges refers to minimum judges per unit and Units to units per task. ρ is Spearman correlation coefficient, σ is the average standard deviation of the assigned score, $\sigma(\rho)$ the average standard deviation of ρ.

name	judges	units	rater countries	ρ	σ	$\sigma(\rho)$
WS353-nat	8	115	USA, CAN, UK	.642	2.30	.168
WS353-notnat	8	115	DEU, FRA, SWE, FIN	.516	2.60	.216
ES353-nat	8	88	USA, CAN, UK	.467	2.36	.201
ES353-notnat	8	88	DEU, FRA, SWE, FIN	.265	2.48	.231
WIN353-nat	8	115	USA, CAN, UK	.451	2.52	.244
WIN353-notnat	8	115	DEU, FRA, SWE, FIN, IRL, CHE, ISR	.328	1.87	.292

Table 3
Scores for selected word pairs in the original WordSim353 dataset (WS-353) and our annotation: WordSim353-crowd, WIN353 and ES353.

word1	word2	WS353	WS353-crowd	WIN353	ES353
Arafat	terror	7.65	3.06	1.62	3.13
Arafat	peace	6.73	2.12	0.93	2.58
Arafat	Jackson	2.50	1	2	2.05
precedent	antecedent	6.04	4	4.79	4.6
cup	coffee	6.58	5.06	3.77	4.36
cup	object	3.69	4.25	3.33	3.82
jaguar	cat	7.42	6.44	5.14	4.98
king	cabbage	0.23	0.64	0.34	1.26

words, the ρ reported in is an unweighted average of Spearman correlations computed between mutual ratings of each pairwise pair of raters. In the table, the original WordSim353 guidelines are referred to as *WS353*, the explicit similarity guidelines are abbreviated as *ES*, and the Word INterchangeability guidelines as *WIN*.

The analysis of results shows that raters from the native speakers group (nat) have higher inter-rater agreement, and that the WIN guidelines have better inter-rater agreement than the explicit similarity guidelines, although both are below the agreement on the original guidelines.

The agreement between the raters in the notnat group is lower than the agreement in the nat group. The variability in the notnat group can be accounted to two reasons. Lower overall command of English might have resulted in some raters not identifying all senses of the given word. Since the words in WordSim353 are nearly all among the 3000 elementary English words [29], we hypothesize that most of the variability is associated with the intercultural differences. This is partially supported by our observation that the interculturally diverse non-nat group has the lowest inter-rater agreement.

4.4. Filtering and merging

The ratings output by Crowdflower have already passed two quality checks. First, the raters had to have a good record on previous jobs they took in the Crowdflower platform to qualify for the task. Second, raters had to correctly answer a number of test questions. Third, we performed additional filtering of the ratings. This was done in a semi-manual manner, by clustering the ratings and then removing three raters which were far from any of the clusters. A limitation of this approach was that raters with small number of ratings could not be reliably clustered and thus were not subject to the final filtering.

The filtering was performed separately for individual guidelines and nat/notnat version of each dataset.

4.5. Final datasets

From the six datasets presented in Table 2 we created thee final ones: WordSim353-crowd, WIN353 and ES353. The overview of the final datasets is given in Table 4.

The *WordSim353-crowd* dataset is created from WordSim353 rated according to the original guidelines. Since we considered the inter-rater agreement in the raw re-rated dataset as satisfactory, we did not perform the filtering step. The scores in the resulting dataset are computed as *macroaverage* of scores obtained on the nat and notnat datasets.

The *WIN353 and ES353* datasets are created from the filtered datasets rated according to the WIN guidelines. The scores in the resulting dataset are computed as *microaverage* of relatedness scores obtained on the nat and notnat datasets after the removal of the outlying raters. The reason for the use of micro average is to reflect the fact that more raters were removed in the notnat group.

Table 3 gives a comparison of average human judgment scores for selected word pairs. For both final datasets, we merge raters from nat and notnat groups. The resulting dataset, created by merging ratings provided by participants from 10 different countries across three continents, can thus better reflect intercultural differences in the perception of relatedness and similarity.

4.6. Disambiguating words to Wikipedia

Some word similarity and relatedness measures require that the input words are disambiguated to specific Wikipedia articles. We foresee that this disambiguation may play even more important role in an emerging class of algorithms that rely on semantic web knowledge bases such as DBpedia [30] to perform the computation. The disambiguation was carried out again using the CrowdFlower

Table 4
Datasets after filtering and merging. ρ is Spearman correlation coefficient, σ is the average standard deviation of the assigned score, $\sigma(\rho)$ the average standard deviation of ρ.

name	judges	tasks	ρ	σ	$\sigma(\rho)$
WordSim353-crowd	16	WS353-nat + WS353-notnat	.525	2.61	.270
WIN353	16	WIN353-nat + WIN353-notnat	.429	2.31	.311
ES353	39	ES353-nat + ES353-notnat	.352	2.54	.226

platform. For each word pair, the workers were given two tasks: to asses word similarity and to disambiguate each of the words to a specific Wikipedia article (not a disambiguation page). The crowdsourcing was setup in a similar manner as for the preceding tasks. The minimum number of workers was set to three, but additionally the tool was setup to dynamically increase the number of judgments to ensure agreement is reached.

After the judgments were gathered they were further manually cleaned and the URIs were normalized. The cases when one word appearing in multiple pairs was mapped to a different URI were inspected. It turned out that after cleaning and normalization, there is only one word ("jaguar") associated with two different URIs.[7] Making an arbitrary choice for jaguar, we could simplify the dataset to mapping of unique words in WordSim353 to Wikipedia articles without noticeable impact on accuracy.

4.7. Limitations of the new datasets

The two new datasets presented in this article – WIN353 and WordSim353-crowd address some of the most important shortcomings of the original WordSim-353 dataset. However, we are aware that the result is affected by the fact that we decided to re-rate the original dataset, rather than propose a new one. We acknowledge the following main limitations:

Unrepresentative word pair selection: The WordSim353 dataset was criticized for the pairs not being selected in a methodically sound and politically correct way by Jarmasz and Szpakowicz [31]; this argument is repeated by Budanitsky and Hirst [19]; Strube and Ponzetto [18]. The gist of the argument focuses on the presence of pairs such as "Arafat, terror", which additionally have high scores in the original dataset.[8] Some other pairs are included in Table 3.

To ameliorate the political bias in WordSim353, we sourced the ratings multinationally. After re-rating according to the original guidelines, the "politically incorrect" pairs are assigned much smaller score as could be expected. Nevertheless, these words remain in the dataset.

Dataset size: Some of the more recent attempts to create benchmarking datasets use many more pairs than 353. For example, SimLex-999 contains 999 word pairs and the BLESS dataset contains even over 250.000 entries.

To help address the dataset size problem, in the following section we present SimLex-999 word pairs rated according to the word interchangeability guidelines. The WINLex-999 dataset is currently only available for Czech.

5. Czech Datasets

Recent research has shown the utility of translating word similarity datasets and obtaining judgments from native speakers in the respective countries [12]. In this paper we present three Czech datasets:

- WIN353cs: WordSim353 word pairs, as translated in Cinková [32]; rated according to Word INterchangeability guidelines
- SimLex999cs: SimLex-999 word pairs translated to Czech and rated according to the original SimLex guidelines. Both word pairs and instructions are in new translation.
- WINLex999cs: SimLex-999 word pairs translated to Czech and rated according to the Word INterchangeability guidelines.

In the following, we describe the process leading to these three new lexical resources.

5.1. Translation of SimLex-999

Within this work we provide a translation of SimLex-999 to the Czech language. We applied similar methodology used for the translation of WordSim-353 as described in Cinková [32]: the translation was done by four paid translators and one adjudicator. The translators were students of the University of Economics in Prague with good command of English and proficiency in Czech.

Table 5 presents inter-translator agreement between all pairs of translators (T1 - T4) and also between all four translators and the adjudicator (A). The inter-translator agreements are relatively high and consistent; the lowest agreement between a pair of translators is 68% and the highest agreement is 72%. All four translators agreed in roughly half of the word pairs (54%). All translators agreed on both words in a pair in 328 cases (33%). In 605 word pairs (61%) translators agreed based on majority vote (at least three), in this case the outcome of the majority vote was used. The adjudicator had to resolve 394 word pairs.

5.2. Guidelines

The design of the Czech dataset required not only translated word pairs, but also guidelines: the original SimLex-999 guidelines and the WIN guidelines. Both were translated by the authors of this article.

[7] This word is featured in four pairs: *jaguar*; *car*, *stock*; *jaguar*, *jaguar*; *cat*, *tiger*; *jaguar*.

[8] The authors of the original WordSim353 paper [2] are affiliated with an Israeli company, and the rating was probably performed by students in Israel.

Table 5
Inter-translator Agreement (Czech version of SimLex-999). Bold denotes the pair of translators with the highest agreement. Italics denotes the pair of translators with the lowest agreement.

	T1	T2	T3	T4	A
T1	×	0.69	**0.721**	0.715	0.82
T2	0.69	×	0.706	*0.678*	0.79
T3	**0.721**	0.706	×	0.711	0.818
T4	0.715	*0.678*	0.711	×	0.82
A	0.82	0.79	0.818	0.82	×

5.3. Setup

Similarly as for the English datasets presented above we used the CrowdFlower platform to collect the ratings. Unfortunately, unlike for English and other languages with many speakers, the CrowdFlower platform does not have many Czech workers – the task would take excessively long time. For this reason, we recruited students of the University of Economics, Prague enrolled in a graduate or undergraduate program in Czech. They accessed the task via the CrowdFlower platform as in the English annotation task described previously. Students were remunerated for their work. The raters were divided into two groups. One group worked on one or both of the WIN datasets, the second group rated the translated SimLex-999 dataset according to the original, but also translated, SimLex-999 guidelines. For Simlex-999, the ratings were elicited on discrete scale 1 to 7 and then transformed to the interval 0 to 10 as for the original SimLex-999 dataset.

5.4. Final datasets

The overview of the final datasets is provided in Table 6. Not all raters provided judgments for all word pairs in each dataset. As a result, the average number of ratings per word pair varies slightly. The reason why the number of unique pairs is lower than the number of total pairs for the two datasets derived from SimLex follows from the fact that several distinct word pairs in English were translated to the same pairs of words in Czech. For example, “give, lend” and “give, borrow” were both translated as “dát, půjčit”. For WIN353cs, one pair of words (money, cash) is included twice, since it also appears twice in the original WordSim353 dataset.

Similarly as Hill et al. [3] we computed inter-rater agreement as the average of pairwise Spearman ρ correlations between the ratings of all respondents. Table 6 also reports the standard deviation of the assigned score (σ) and the average standard deviation of ρ. The statistics for WIN353cs can be directly compared to results of WIN353-nat in Table 2 and to WIN353 in Table 4. It follows that with $\rho = 0.55$ the WIN353cs dataset has better inter-rater agreement than WIN353-nat ($\rho = 0.45$), which we obtained for the English original. The inter-rater agreement as measured by Spearman ρ in the original SimLex-999 dataset is $\rho = 0.67$ [3]. For the re-annotated version in Czech with original guidelines, we obtained $\rho = 0.61$ and with the WIN guidelines we obtained $\rho = 0.58$.

6. Evaluation

This section presents the evaluation of selected WordNet measures and Wikipedia-trained distributional algorithms on commonly used similarity and relatedness datasets.

6.1. Choice of datasets

For relatedness, the benchmark includes WordSim353 [15], MTurk-771 [33], MEN-3000 [34], and our reannotated version of WordSim353 called WordSim353-crowd. As similarity datasets we include RG-65 [35], MC-30 [36], the noun subset of SimLex-999 [3], and Since the dataset obtained with the explicit similarity guidelines (ES353) had lower inter-rater agreement than we obtained for the word interchangeability guidelines we decided to use WIN353 from the newly proposed datasets.

Specific setup for several datasets:

- *SimLex:* In the primary evaluation we include only the noun subset of SimLex-999. We call this dataset *SimLex-666.* The motivation for excluding the verb and adjective pairs is to provide a level field for algorithms that require the word to be disambiguated to a Wikipedia article (BOW and WLM measures) as Wikipedia does not in general contain articles describing adjectives and verbs. Additionally, the knowledge base of the JWSL library, which provides multiple WordNet similarity measure implementations

Table 6
Overview of Czech datasets.

dataset	pairs	unique pairs	avg annotations	ρ	$\sigma(\rho)$	σ
WIN353cs	353	352	10	0.55	0.1	2.15
WINLex999cs	999	993	8.1	0.58	0.11	2.39
SimLex999cs	999	993	28.4	0.61	0.08	2.41

used in our evaluation, is restricted only to nouns, and precomputed information content files are not available for adjectives.

- *MEN-3000:* The WordNet-based algorithms failed to map at least one word to a WordNet synset in 377 word pairs. All these word pairs contained one word with other than noun part of speech tag.

Due to a limited availability of suitable lexical resources, the benchmark of the three newly proposed Czech Datasets is left for future work.

6.2. Implementations and setup

Our evaluation covers mainstream WordNet-based similarity algorithms and distributional algorithms trained on Wikipedia. Overview of the algorithmic setup is given in Table 7.

For WordNet, the following algorithms are included: Resnik [14], Jiang&Conrath [39], Lin Measure [40], Pirro&Seco Measure [41].

We evaluate the following Wikipedia-trained distributional algorithms: Wikipedia Link Measure (WLM) [16], Explicit Semantic Analysis (ESA) [42], and a Neural Network Language Model (NNLM): Skip-gram with negative sampling [43]. As a baseline we use a simple Vector Space Model (VSM), where the bag-of-words (BOW) representation is created from the complete text of the article describing the word (whole document context).

Additional explanation is required for the WordNet methods. Two Java libraries were selected: JWordnetSim and JWSL, which represent two fundamental approaches to computing information content values required by all the evaluated WordNet measures. It is often the case that a word matches multiple synsets. Both JWordnetSim and JWSL libraries offer two ways to deal with this situation. It is either possible to select the first sense for given word, which corresponds to the Most Frequent Sense (MFS) option. The second option is to let the library return the similarity maximizing combination of senses. We call this Synset Similarity Maximization (SSM).

As the evaluation metric, we employ the Spearman correlation coefficient. This metric is used in more recent papers [3,4]. Some older research uses the Pearson product-moment correlation, this applies for example to the work of Strube and Ponzetto [18]. The values of Pearson product-moment correlation are used interchangeably with Spearman rank correlation by Gabrilovich and Markovitch [15].

6.3. Which algorithms measure relatedness and which similarity?

All WordNet-based algorithms included in our evaluation are considered as similarity measures by their authors. WLM and ESA are considered as relatedness measures by their authors.

For the generic distributional algorithms (BOW and NNLM) the situation is more complex. Sahlgren [44] showed that small context window captures well the similarity relation. This result was confirmed by Peirsman et al. [45]. Agirre and Soroa [27] performed a study with multiple context sizes and vector space algorithms on a similarity dataset of Rubenstein and Goodenough [35] and on a relatedness dataset (WordSim353). Their results also indicate that smaller window sizes better model similarity, but

Table 7
Evaluation setup: algorithms, implementations, parameters.

implementation	description
JWordnetSim (Lin, JCn)	Java implementation of WordNet measures. As Information content files we used the ones available within the WordNet::Similarity project (http://wn-similarity.sourceforge.net/). The JWordnetSim library was used in conjunction with WordNet 2.0. Website: http://nlp.shef.ac.uk/result/software.html, used version 1.0.0
JWSL (Pirro&Seco, Resnik, Lin, JCn)	Supports additional similarity measures compared to JWordnetSim. It also uses the intrinsic information content, which is computed directly from WordNet. The library is not freely available, but it is provided by the authors upon written request. Website: https://simlibrary.wordpress.com/
WikipediaMiner	This toolkit is an official implementation of the WLM measure [37]. It was used with Wikipedia snapshot from November 2013. Website: http://wikipedia-miner.cms.waikato.ac.nz/, used version 1.2.0
ESAlib	ESA implementation with Wikipedia snapshot from 2005. Website: http://ticcky.github.io/esalib/
Word2vec	Skip gram with negative sampling trained on English Wikipedia using the word2vec software (https://code.google.com/p/word2vec/). The pretrained wordvectors were retrieved via the word2vec homepage for the Google News corpus and from https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/ (Wikipedia corpus). Both resources use 300 dimensions.
BOW (whole document context)	Own implementation using English Wikipedia dump from 2011. Disambiguation: Wikipedia Search or manual disambiguation. Preprocessing: stop-words were removed, terms are sorted according to term frequency and N most frequently occurring terms are kept as suggested by Feldman and Sanger [38] (we used N = 10000). Term weighting: TF-IDF, where TF refers to the term frequency of a word in the article, and IDF to inverse document frequency computed from the entire Wikipedia. Similarity function: cosine similarity.

Table 8
Results of WordNet measures (JWordnetSim). Bold values denote the highest result for each dataset.

dataset	MFS		SSM	
	JCn	Lin	JCn	Lin
Similarity datasets				
SimLex-666	.47	.46	.58	**.58**
WIN353	.33	.42	.47	**.49**
WIN353#WSSim	.50	.57	.61	**.62**
MC-30	.55	.54	**.80**	.73
RG-65	.43	.51	**.78**	.76
Relatedness datasets				
WordSim353	.23	**.32**	.30	.32
* WordSim353_WNAlign	.24	**.33**	.31	.33
WordSim353#WSSim	.49	.58	.61	**.61**
WordSim353-crowd	.30	.38	.38	**.40**
WordSim353#WSRel	.02	.01	-.01	-.01
WordSim353-crowd#WSRel	.09	.10	.10	**.10**
MEN-3000	.26	.25	**.37**	.36
MTurk-771	.29	.30	**.50**	.50

Table 9
Results of WordNet measures (JWSL). Bold values denote the highest result for each dataset.

dataset	Most frequent sense				Synset similarity maximization			
	Resnik	JCn	Lin	P&S	Resnik	JCn	Lin	P&S
Similarity datasets								
SimLex-666	.46	.46	.48	.47	.52	**.59**	.59	.58
WIN353	.42	.42	.42	.42	.49	.50	.50	**.51**
WIN353#WSSim	.56	.58	.57	.58	.62	.65	.64	**.65**
MC-30	.53	.68	.57	.67	.68	**.80**	.74	.79
RG-65	.53	.54	.53	.55	.74	**.80**	.78	.80
Relatedness datasets								
WordSim353	.31	.32	.31	.32	**.33**	.31	.33	.33
WordSim353#WSSim	.56	.58	.57	.58	.60	.62	.62	**.63**
WordSim353-crowd	.38	.38	.38	.38	.40	.40	.41	**.41**
WordSim353#WSRel	.00	.00	.00	.00	**.01**	.00	.01	.00
WordSim353-crowd#WSRel	.10	.09	.09	.10	**.13**	.11	.12	.12
MEN-3000	.24	.25	.25	.25	.32	.34	.34	**.35**
MTurk-771	.28	.28	.28	.28	.39	**.49**	.48	.46

there is no clear tendency in obtaining better results for relatedness when the window size is increased (up to 7 words were tried).

6.4. Thesaurus-based similarity measures

Results of WordNet measures on the evaluation datasets are presented in Table 8 (JWordnetSim library) and Table 9 (JWSL library).

The performance of the individual measures is quite similar for JWSL (Table 9) and JWordnetSim, with the exception of the JWordnetSim implementation of Jiang & Conrath measure (MFS) underperforming by about 8% compared to the JWSL implementation.

It should be noted that for the JWordnetSim library we performed parameter tuning in terms of selecting the most suitable source of information content values. The results are depicted in Table 10. For subsequent experiments we used infocontent file generated from the British National Corpus with Resnik counting and smoothing.[9] Finally, it can be seen from Table 8 that the results for the "Aligned" version, where the seven words not in WordNet were manually replaced by words in WordNet, are about 1% higher. The alignment step has thus only small effect and we could omit it for the other experiments.

Comparing WordSim353 with WordSim353-crowd, the results indicate that WordNet methods are in higher agreement with the crowdsourced annotation (0.08 improvement from 0.30 to 0.38). Regarding WordSim353 and WIN353, the figures show that the WIN guidelines increase ρ by 0.17 points (from 0.30 to 0.47). This approximately 50% relative increase in the results of WordNet similarity algorithms could be considered as a success metric for accomplishing our goal for designing a similarity dataset based

[9] We decided to use BNC corpus instead of the slightly better performing Semcor-raw corpus due to more consistent results when smoothing or Resnik counting was not used.

Table 10
The impact of information content values on the performance of JWordnetSim JCn measure (SSM) on the WordSim353 collection (WNAlign).

smoothing resnik c.	no no	yes no	no yes	yes yes
bnc	.288	.299	.300	.321
brown	.249	.257	.278	.291
semcorraw	.134	,.139	.291	.325
shaks	.121	.128	.257	.277
treebank	.271	.280	.299	.310

on WordSim353 lexical content. In line with our expectations, the overall best results of WordNet measures were achieved on the similarity datasets.

A surprising finding is the performance on the WordSim353#WSRel subset, on which all the measures yield effectively zero correlation coefficient. This shows that WordNet measures are not confused by the association relationship between words, and reliably determine that two words are not similar even if they are strongly related. This result is confirmed also on the reannotated dataset (WordSim353-crowd#WSRel). There the correlation is somewhat higher, with ρ around 0.1, but still very low.

Overall, from the similarity datasets the lowest correlations were obtained on our WIN353 dataset. The highest correlations were obtained for the oldest MC-30 and RG-65 datasets.

6.5. *Distributional measures*

This section presents the evaluation of the distributional measures trained on Wikipedia. Some of the measures involved were directly designed for Wikipedia (ESA, WLM). Also, the NNLM skip-gram algorithm is often trained on Wikipedia. Wikipedia is also suitable for the whole-document context BOW. To complement the results in terms of how the choice of Wikipedia for training impacts the final results, we also provide numbers for NNLM trained on the Google News corpus. The results for the distributional measures are summarized in Table 11.

The best performance on relatedness datasets is obtained by the ESA algorithm, which outperforms all measures on WordSim353 including its WSRel subset. This holds for both the original and reannotated datasets, with the exception of the (complete) WordSim353-crowd, where the skip-gram NNLM with window size 5 obtains the best result. For similarity datasets, the best won-tie-loss record has the skip-gram NNLM with window parameter set to two.

The observation that smaller window size models better similarity is in accordance with earlier experimental results [4,44,45]. Also in accordance with the expectations, the distributional measures obtain lower correlations on similarity datasets than on the relatedness datasets. Overall, the lowest ρ for the distributional algorithms was recorded for SimLex-666. This may be interpreted as SimLex meeting its design objective of being a hard dataset. In Section 7.3, where we analyze the SimLex dataset, we argue that one of the specific causes is the lexical composition of the SimLex dataset, which is adversarial to Wikipedia-trained measures (cf. Section 7.3).

Our attempt to provide manual disambiguations of words in the WordSim353 dataset to Wikipedia articles has a clear conclusion. These disambiguations do not improve the results. For the BOW model, we obtained increase in ρ lower than 1 point. For WLM the results are clearly impaired, which is caused by the inability of WLM to correctly process all the manual disambiguations. This finding

Table 11
Results for distributional measures. For the WLM measure, we list in the parentheses number of words that were not disambiguated or recognized. For NNLM the window size is listed in the parentheses. W succeeded by number indicates the year of the Wikipedia snapshot used as corpus. † personal communication. NEWS – trained on Google News corpus (100 billion words). Highest result on Wikipedia is listed in bold.

algorithm corpus	WLM W13	ESA W05	BOW W11	NNLM (2) W13†	NNLM (5) W13†	NNLM (5) NEWS
Similarity datasets						
SimLex-666	.36 (26)	.32	.38	**.44**	.38	.45
WIN353	.58 (9)	.51	.48	**.61**	.61	.62
WIN353#WSSim	.66 (7)	.65	.56	.67	**.68**	.69
MC-30	**.81** (1)	.76	.68	.73	.73	.79
RG-65	**.83** (7)	.79	.74	.72	.77	.75
Relatedness datasets						
WordSim353	.68 (9)	**.74**	.66	.66	.69	.70
* ManualDisamb	.50 (0)	NA	**.66**	NA	NA	NA
WordSim353#WSSim	.76 (7)	**.77**	.70	.74	.76	.78
WordSim353-crowd	.62 (9)	.65	.61	.65	**.67**	.69
WordSim353#WSRel	.59 (7)	**.74**	.56	.56	.61	.62
WordSim353-crowd#WSRel	.55 (7)	**.66**	.53	.61	.63	.64
MEN-3000	.68 (270)	**.74**	.57	.69	.72	.77
MTurk-771	.51 (35)	.61	.49	**.64**	**.64**	.67

Table 12
Overall best Wikipedia and WordNet-based results (based on Tables 8, 9 and 11).

dataset	Wordnet		Distributional measure	Wikip. ρ_{ds}	$\rho_{ds} - \rho_{wn}$
	measure	ρ_{wn}			
Similarity datasets					
SimLex-666	JCn	0.59	NNLM (2)	0.44	−0.15
WIN353	P&S	0.51	NNLM (2)	0.61	0.11
WIN353#WSSim	P&S	0.65	NNLM (5)	0.68	0.03
MC-30	JCn	0.80	WLM	0.81	0.01
RG-65	JCn	0.80	WLM	0.83	0.03
Relatedness datasets					
WordSim353	Resnik	0.33	ESA	0.74	0.41
WordSim353#WSSim	P&S	0.63	ESA	0.77	0.14
WordSim353-crowd	P&S	0.41	NNLM (5)	0.67	0.26
WordSim353#WSRel	JCn	0.02	ESA	0.74	0.72
WordSim353-crowd#WSRel	Resnik	0.13	ESA	0.66	0.53
MEN-3000	JCn	0.37	ESA	0.74	0.36
MTurk-771	JCn	0.50	NNLM (5)	0.64	0.14

supports the observation of Milne and Witten [16] that the WLM disambiguation algorithm is as good as a human disambiguation. Additionally, the result shows that this holds also for the simple Wikipedia search disambiguation used in our whole-document context BOW model.

The last column in Table 11 shows that the results obtained with skip-gram NNLM trained on Wikipedia are 0.01–0.04 points below those obtained on the much larger Google News corpus with the same window size.

6.6. *Best algorithms for relatedness and similarity*

A summary of our empirical results structured according to the problem type (similarity vs relatedness) and algorithm type (thesaurus-based or distributional) is provided in Table 12. The last column indicates to what extent does the best distributional algorithm outperform the best WordNet measure.

Similarity: The performance of thesaurus-based and distributional algorithms is quite leveled. The WordNet measures perform better on the SimLex-666 dataset, while on WIN353 the NNLM with window size 2 outperforms by 0.11 points the best WordNet measure. An unexpected result is that the WLM measure is the best algorithm overall, though by a small margin, on MC-30 and RG-65 datasets.

Relatedness: On all datasets the results are dominated by distributional algorithms. The ESA algorithm scores the best on five datasets, and NNLM with window size 2 on the remaining two. It is interesting to observe that a relatively dated ESA algorithm using a 2005 release of Wikipedia outperforms a state-of-the-art NNLM algorithm trained on a 2013 (newer and thus larger) Wikipedia snapshot.

7. Related work

This section compares our dataset design with previous work. Special attention is paid to the recently proposed SimLex-999 dataset.

7.1. *Datasets*

The initial research in word similarity algorithms relied on the datasets proposed by Rubenstein and Goodenough [35]. The guidelines asked the raters to give synonymy judgments, thus it can be considered as a similarity dataset. The RG-65 dataset was subsequently augmented by Miller and Charles [36]. Both datasets were used to illustrate the effectiveness of the WordNet similarity algorithms by their respective authors. The MC-30 dataset was used for example by Jiang and Conrath [39]; Lin [40]. Benchmarks of WordNet-based measures performed on these datasets were carried out by Budanitsky and Hirst [19]; Strube and Ponzetto [18]; Pirró and Euzenat [17]. Another benchmark including also other than WordNet measures, but restricted to MC-30 dataset, was presented in Agirre et al. [4].

Another dataset used for measuring similarity is the TOEFL (Test Of English as a Foreign Language) dataset [46], which contains 80 multiple-choice synonym questions with 4 choices per question. This dataset measures only synonymy and thus it complies to all similarity definitions. However, since it requires a binary classification of word pairs as synonymous or not, it does not discern well pairs of medium or low similarity [3]. Also this dataset might be too easy for contemporary algorithms: Rapp [47] achieved 92.5% correct on the 80 TOEFL questions, using a four-word context window (+-2 words, centered on the target word, after removing stop words). Bullinaria and Levy [48] even report obtaining 100% correct results.

Gentner and Markman [20] created a word similarity dataset for an experiment in cognitive science. This dataset contains 40 word pairs, with similarity ratings on a 9-point scale. To the best of the authors' knowledge this dataset has not been used for research within computer science.

Once the larger WordSim353 dataset has been introduced by Finkelstein et al. [2]; it was used to supplement and eventually replace the RG-65 and MC-30 datasets in evaluations of word similarity and relatedness measures. The evaluation in papers introducing the WikiRelate! and WLM algorithms was performed on WordSim353 in addition to RG-65 and MC-30 [16,18]. The paper introducing the ESA algorithm [15], for some time the state-of-the-art in the word relatedness computation, features only evaluation on WordSim353.

Radinsky et al. [49] introduce a new crowdsourced (AMT) dataset MTurk-287 in order to provide additional benchmark for their TSA algorithm. The word pairs for the MTurk dataset were generated automatically based on co-occurrence in a large text corpus. This dataset was annotated according to the WordSim353 guidelines. A larger version of this relatedness dataset containing 771 word pairs was introduced by Halawi et al. [33].

A consensus has been reached in the scientific community on WordSim353 dataset not evaluating similarity [3,4,15]. However, there is some disagreement as to whether it measures relatedness. Gabrilovich and Markovitch [15] argue that it does, while Agirre et al. [4] suggest that two versions of the dataset need to be created with precise instructions for similarity and relatedness annotation, proposing that as an intermediary solution subsets of WordSim353 (with the original ratings) can be used, creating the WSSim and WSRel datasets. Hill et al. [3] assert that WordSim353 measures word *association*. Since the same guidelines were used for WSSim dataset, according to the same authors association is measured also by its WSSim subset [4].

In order to better serve applications in computer vision, the *MEN* collection[10] was recently proposed [34]. This dataset contains only words that occur in image labels in ESP-GAME and MIRFLICKER-1M collections. The dataset contains 3000 word pairs, with ratings crowdsourced using the Amazon Mechanical Turk via the CrowdFlower interface. The raters were presented two pairs of words and asked to judge which pair is more related. To the best of our knowledge, the MEN collection is the largest resource for measuring semantic *relatedness*.

There are several other datasets designed for specific purposes, for example sentence similarity dataset containing ratings for 50 pairs of short documents [50].[11]

WordSim353, MC-30, RG-65 and SimLex-999 are *human judgment* datasets since these associate word pair with a score based on multiple human judgments. *Semantic relation* datasets represent another type of evaluation resource, which is sometimes used for evaluation of word similarity and relatedness algorithms. These datasets associate word pairs with a relation, such as meronymy (alligator, eye) or attribute (alligator, aggressive). The most notable semantic relation datasets are SN [51] and BLESS [52]. Since these datasets are not readily associated with a similarity rating, they cannot be used for evaluation in the same manner as the human judgment datasets. However, their advantage is that they provide additional insight into the performance across individual word relations.

The largest semantic relation resource is the BLESS dataset (Baroni-Lenci Evaluation of Semantic Similarity), which contains 265.554 entries. One entry is a tuple containing: target concept (one POS-tagged word), broader semantic class of the target concept, relation between target word, and relatum (second POS-tagged word). Neubauer et al. [53] annotated all term pairs in BLESS dataset with a similarity rating. The authors do not recommend to use the resulting data for similarity benchmark due to volatility – the dataset has typically only one judgment per pair of terms in BLESS. Nevertheless, the result of their experiment provides sufficient amount of data to show that human participants exhibit clear preference towards hypernyms, with co-hypernyms being the least preferred group of word relations. This indicates that there are significant differences in preferences based on word relation between humans and algorithms.

We are not aware of any other dataset which measures word similarity based on the word interchangeability definition. However, the WIN353 and WINLex999cs datasets are not the only efforts to harness word interchangeability for word similarity computation. Biemann [54] published the Turk Bootstrap Word Sense Inventory (TWSI) dataset, which is a crowdsourced sense inventory for lexical substitution for one thousand highly frequent English common nouns. The TWSI dataset is used, for example, as a component in a system for computing similarity between texts [55].

Focusing on word similarity datasets with available similarity judgments, we consider the SimLex-999 dataset the largest and at the same time most well-founded dataset. Despite the dataset being proposed recently, it is already widely used in benchmarks according to our literature review.

7.2. Crowdsourcing and multilinguality

Our datasets were reannotated using crowdsourcing following the seminal paper of Snow et al. [56]; which asserts that Amazon Mechanical Turk (AMT) workers can replace experts producing essentially the same result if higher number of raters per unit is employed. By including raters from multiple different countries and of different levels we aim to address the objections recently raised by Sen et al. [57] regarding the use of crowdsourcing for creating gold-standard datasets for natural processing research. The authors dispute the conclusions of Snow et al. [56]; hypothesizing that the results might be substantially influenced by the community the raters come from.

To support this hypothesis Sen et al. [57] performed an experiment with creating a gold standard for measuring concept relatedness. The raters came from several distinct communities (Amazon Mechanical Turk workers, scholars, scholars-experts). The findings showed large differences in Pearson correlation coefficient based on the community creating the gold standard. For example, ESA

[10] http://clic.cimec.unitn.it/~elia.bruni/MEN.

[11] Manaal Faruqui maintains a list of such datasets at http://www.cs.cmu.edu/~mfaruqui/suite.html.

Table 13
WIN353 and ES353 vs SimLex-999 comparison: † as reported by Hill et al. [3]].

metric	WIN353	ES353	SimLex-999
rater countries	10	7	1 (USA)
inter-rater agreement	0.43	0.35	0.67†
raters per word pair (min.)	16	33	36
average similarity rating	3.18	3.77	4.07
synonymy as similarity	yes	yes	yes
antonymy as similarity	no	yes	no

obtained $\rho = 0.7$ on the AMT gold standard, $\rho = 0.6$ on the scholar gold standard, and only $\rho = 0.45$ on the expert scholar gold standard.

For completeness, the impact of language ability on word similarity ratings was studied by Pirró [41]. This paper found a high level of agreement after excluding outliers. However, the validity of this research is limited by small dataset size of only 65 word pairs.

Several multilingual datasets for benchmarking word similarity and word relatedness have emerged. One of the first approaches is the work of Zesch and Gurevych [58]; who describe automatic corpus-based system for creating test datasets for evaluating similarity and relatedness measures. The resulting dataset is called ZG-328 and its language is German. Despite this early effort, there are not many non-English similarity and relatedness datasets.

To our knowledge, the most comprehensive work to date to address multilinguality was performed by Leviant and Reichart [12]; who created Italian, German and Russian translations of SimLex-999. The selection of languages covers three branches of the Indo-European language family: Germanic, Romance and Slavic. The range of resources for the Slavic language family has strengthened recently: Cinková [32] translated the WordSim-353 dataset to Czech and Panchenko et al. [59] translated multiple datasets from the similarity and relatedness domain to Russian, including the HJ dataset, which is a union of RG-65, MC-30 and WordSim353.

7.3. SimLex-999 vs WIN353 and ES353

Both SimLex-999 dataset and our WIN353 aim to measure word similarity. There is a number of differences among the datasets, which are summarized in Tables 13 and 14. The most important differences and their impact on algorithmic performance is discussed in the remainder of this section.

Definition of similarity. SimLex-999 guidelines aim to distinguish word pairs in *semantic similarity* relation (synonymy) from those in *associative relation* (remaining types of relations). This is reflected in the annotation guidelines of SimLex-999:

> “If you are ever unsure, think back to the examples of synonymous pairs (glasses / spectacles), and consider how close the words are (or are not) to being synonymous.”

This instruction implies that antonymy should result in dissimilarity.

The justification for the exclusion of antonymy from the similarity relations in the SimLex-999 paper is not completely consistent, since antonyms match the intuitive definition of similarity the authors give in their paper:“… can be understood as similar … because of their common function”. Antonyms typically have a common function: consider SimLex pairs “south, north” and “top, bottom”. These words are antonyms, but yet they can both serve for giving directions. As Table 15 shows antonyms are indeed assigned low similarity in SimLex-999.

While the definition of similarity used in WIN353 is based on word interchangeability guidelines, as can be seen from Table 16 antonymy was not considered as similarity. We attribute it to the following clause in the instructions:

> “By interchangeability of two words, we understand the degree with which one word can be replaced by the other word in a randomly chosen sentence without a change in the meaning.”

Future version of the instructions should thus more closely follow the definition of paradigmatic association cited in Section 2.1.

The EX-353 guidelines explicitly qualified synonymy, antonymy and hypo-hypernymy as similarity relations. These guidelines

Table 14
Lexical content: WordSim-353 (WIN353/ES353) vs SimLex-999 comparison: † this metric was computed only on the adjective subset of 111 word pairs,* number of pairs with both words having a sense with noun POS tag in WordNet 2.1.

metric	WordSim-353	SimLex-999
pairs with informal word	2 (1%)	42 (38%) †
synonym pairs	48 (14%)	280 (28%)
antonym pairs	5 (1%)	57 (6%)
sentiment pairs	33 (9%)	145 (15%)
noun pairs	344*	666
adjective pairs	–	222
verb pairs	–	111

Table 15
Antonyms in SimLex-999: groundtruth vs WordNet. The first number gives the position of the pair in a list sorted according to rating in a descending order. The number in parenthesis gives the actual SimLex groundtruth score, which is in interval 0 to 10 (most similar). Control pairs (not antonyms) are given in the bottom of the table. Antonym pairs were randomly chosen. For NNLM the window size is listed in the parentheses.

word1	word2	groundtruth	JCn (SSM)	JCn (MFS)	ESA	WLM	NNLM (5)
south	north	532 (2.2)	178	313	36	59	6
north	west	402 (3.63)	298	465	69	50	210
bottom	top	614 (0.7)	109	220	129	53	153
bottom	side	491 (2.63)	93	222	449	658	172
mouth	tooth	177 (6.3)	327	316	307	92	338
breakfast	bacon	337 (4.37)	594	348	167	135	399
flower	endurance	656 (0.4)	631	613	581	628	660

Table 16
Averagescore for antonyms in the original WordSim-353 (WS353) and in the datasets we derived from it.

Word 1	Word 2	WS353	WS353-crowd	WIN353	WIN353cs	ES353
student	professor	6.81	4.94	1.21	3.56	3.93
smart	stupid	5.81	3.25	3.21	3.6	4.15
life	death	7.88	4.38	1.6	3	4.34
profit	loss	7.63	3.88	2.07	3.2	3.97
man	woman	8.3	4.5	2.57	4.3	5.3
average		7.29	4.19	2.13	3.53	4.34

resulted in the highest average scores for antonyms in datasets rated in our experiments (cf. Table 16).

Diversity of annotators and inter-rater agreement. There are about 375 million of speakers with English as a first language, 375 million of speakers of English as a second language and 750 million speakers of English as a foreign language according to Crystal [60]; cited according to Anchimbe [61]. The WIN353 dataset composition thus better matches this distribution than the original WordSim353 dataset (annotated solely by non-native English speakers) or the SimLex-999 dataset (annotated solely by native English speakers). However, the higher diversity of WIN353 and ES353 raters impacts the inter-rater agreement, which is lower than the one for SimLex-999 or WordSim353. The fact that the low agreement for WIN353 can largely be attributed to diversity of raters is supported by higher agreement for WIN353cs, which was annotated by more homogeneous group of raters. Agreement rate is sometimes considered as a ceiling against which to compare natural language processing algorithms [62].

Informal words. To evaluate the presence of informal words, we used the list of 255 adjectives that are at least twice as frequent in soap operas than in the 450-million word Corpus of Contemporary American English.[12]

Since the informal word list is available only for adjectives, we could not evaluate entire SimLex-999, but only its adjective subset. Out of the 111 word pairs in the SimLex-999 adjective subset, 42 contain an informal word. Example of informal words include “happy”, “insane” of “funny”. For WordSim353 and datasets derived from it, there are only two word pairs (out of 47 containing an adjective) with at least one of the words on this list: (*smart-student*, *smart-stupid*).

Presence of informal words has an adverse impact on the correlation of distributional algorithms trained on Wikipedia. We assume that this is because these words are generally underrepresented in Wikipedia due to its encyclopedic character. Also, according to our preliminary observation, these words tend to frequently occur in other contexts than their formal synonyms. For example, the Wikipedia disambiguation page for word “happy” lists 10 films and television episodes, five albums, several dozens of songs, but also books, people and places with this word in name. The impact on the correlation coefficient is illustrated in Table 17.

Sentiment words. In order to assess the sentiment of words present in SimLex-999 and WordSim353 datasets, we used the SentiStrength library [63].[13] This library assigns positive and negative sentiment scores to short texts. We processed individually both words in each pair. If any of the words was assigned either stronger positive sentiment than 1 (“not positive”) or stronger negative sentiment than −1 (not negative), we tagged the pair as containing a sentiment word.

The statistics depicted in Table 13 show that SimLex-999 has with 15% sentiment pairs 66% more words expressing sentiment than WordSim353.

Synonyms and antonyms. In order to identify word pairs in synonymy or antonymy relationship, we used the Roget’s 21st century thesaurus, 3rd edition.[14] In comparison to WordNet, Roget’s thesaurus contains more synonyms and antonyms (500.000). Moreover, its antonymy listing are not restricted only to adjectives as in WordNet. It should be noted that thesaurus.com distinguishes three degrees of antonymy and synonymy, for our statistics we used all degrees.

Table 13 shows that SimLex-999 contains twice as many pairs in direct synonymy relation than WIN353. Many synonym pairs make SimLex-999 a less challenging resource for algorithms that use thesaurus. This is in-line with the results we obtained for

[12] http://corpus.byu.edu/coca/100k_data.asp?query=7.

[13] Available from http://sentistrength.wlv.ac.uk/.

[14] Available from thesaurus.com.

Table 17
Effect of antonyms, informal words and sentiment on SimLex-111 (SimLex-999 adjective subset) and SimLex-999 using word2vec (Skip-gram, trained on Wikipedia, window size 2), WLM and ESA methods. WLM obtained correlation very close to zero due to many disambiguation failures (most adjectives do not have a Wikipedia article).

dataset	SimLex-111			SimLex-999		
	NNLM (2)	WLM	ESA	NNLM (2)	WLM	ESA
All	0.54	−0.07	0.21	0.41	0.23	0.23
- antonyms	0.65	−0.03	0.30	0.46	0.22	0.23
- informal adjectives	0.66	−0.06	0.33	0.45	0.24	0.23
- sentiment	0.73	−0.07	0.36	0.46	0.23	0.24

Table 18
Summary of earlier results reported for WordSim353. Subscript $_P$ stands for Pearson correlation coefficient and $_S$ for Spearman respectively, † denotes that the paper explicitly mentions the result only on pairs with words contained in WordNet, ? denotes that the type of correlation coefficient is not known.

measure	source	correlation
WordNet measures		
Resnik	Pirró and Euzenat [17]	0.40_P†
Resnik	Strube and Ponzetto [18]	0.34_P†
P&S	Pirró and Euzenat [17]	0.41_P†
Lin	Pirró and Euzenat [17]	0.40_P†
JCn	Pirró and Euzenat [17]	0.40_P†
Pers. PageRank	Agirre and Soroa [27]	0.58_S
Distributional measures		
text	Strube and Ponzetto [18]	$.20_P$†
WLM	Milne and Witten [16]	$.69_?$
ESA	Gabrilovich and Markovitch [15]	$.75_S$
NNLM (CBOW)	Chen and de Melo [64]	$.64_S$
NNLM (CBOW)	Baroni et al. [65]	$.75_S$
NNLM (Skipgram)	Hill et al. [3]	$.655_S$
NNLM (Skipgram)	http://wordvectors.org	$.64_S$

WordNet-based algorithms.

SimLex-999 contains 6% of antonym pairs. Due to the SimLex-999 guidelines, this has adverse impact on the performance of both thesaurus-based and distributional methods as illustrated in Table 17. The number of antonyms in WIN353 is negligible.

Difficulty of the dataset. Hill et al. [3] gives as one of the incentives for designing new similarity dataset the fact that existing state-of-the-art algorithms obtain correlations close to 1.0 on WordSim353 and this dataset thus no longer provides a reliable benchmark of new algorithms. In response to this, SimLex-999 was designed to contain "*a significant number of pairs, such as [movie, theater], which are strongly associated but receive low similarity scores.*"[15] This lowers the average rating assigned to word pairs compared to WordSim353 and makes the dataset harder for word similarity algorithms.

As it follows from the previously presented analysis, WIN353 is more challenging than SimLex-999 as it contains much less synonyms and antonyms found in a thesaurus (cf. Table 13). The word pairs in WIN353 also have lower average similarity score (4.07 vs 3.18) assigned by the raters.

7.4. Benchmarks

The purpose of this section is to demonstrate that our implementations and parameter setup do not deviate substantially from what has been reported in prior research. To this end, Table 18 presents an overview of results obtained for WordSim353 and Table 19 the results for R&G and M&C datasets. We selected these datasets, since they are probably the most studied ones.

On WordSim353, our correlations for ESA and WLM correspond to the ones reported in Milne and Witten [16]. Considering WordNet measures, our results are slightly lower than those in earlier research. This might be caused by a number of factors including different information content file, different correlation measure as Pearson correlation was used in older research, handling of words not found in WordNet and different WordNet versions and WordNet subsets used (we considered only nouns).

It should be noted that our benchmark does not include the PageRank-based WordNet algorithm proposed by Agirre and Soroa [27]. This could be considered as a state-of-the-art among the WordNet measures with respect to the achieved ρ on WordSim353. However, this algorithm uses WordNet in such a way that we are unsure it can be considered as a similarity measure (the approach

[15] We conjecture that this decision might be related to the fact that SimLex-999 word pairs were sampled from free association norms and synonymy accounts for about 14% of word pairs in word association norms [24].

Table 19
Summary of earlier results reported for similarity datasets. Results for Wikipedia-based measures for RG-65 and MC-30 are sourced from Milne and Witten [16]; results for WordNet-based measures from Budanitsky and Hirst [19] (neither gives type of correlation coefficient). Results in parentheses are sourced from Agirre et al. [4]. The NNLM result for RG-65 is sourced from Baroni et al. [65]; the model was trained on a large ukWaC corpus word2vec (skip gram). The results for SimLex-666 are sourced from Banjade et al. [66]. Subscript $_P$ stands for Pearson correlation coefficient and $_S$ for Spearman respectively. ? denotes that the type of correlation coefficient is not known.

Dataset	Distributional measures				WordNet measures		
	WikiRelate!	WLM	ESA	NNLM	Resnik	Lin	JCn
MC-30	$.45_?$	$.70_?$	$.73_?$	–	$.77$ (0.81_S)	$.83_?$ (0.82_S)	$.85_?$ (0.83_S)
RG-65	$.52_?$	$.64_?$	$.82_?$	$.84_P$	$.78_?$	$.82_?$	$.78_?$
SimLex-999			$.271_S$	$.442_S$			
SimLex-666				$.452_S$	$.443_S$	$.452_S$	$.451_S$

was originally proposed for word sense disambiguation).

Our BOW model can be compared to the *text* overlap method evaluated by Strube and Ponzetto [18] on WordSim353, since both methods derive the relatedness score from comparing the texts of articles describing the input words. We attribute the large difference in the performance of both methods ($\rho = 0.66$ vs $\rho = 0.2$) to somewhat simplistic model being used for the text method. It did not involve stop word list or term pruning, term weighting was limited to normalization by text length. In contrast to our BOW representation that uses all words in the article defining the word, *distributional semantic models* (DSMs) are built based on a co-occurrence pattern, which is learnt from a window around the target word. A comprehensive evaluation of DSMs on RG-65 and WordSim353 datasets was recently performed by Lapesa and Evert [67].

Our result for word2vec ($\rho = .69$) is on par with other results reported in the literature for WordSim353. Neither for WordSim353 nor for the RG-65 dataset our results reach the current state-of-art. The higher correlations reported by Baroni et al. [65] can be explained by Baroni et al. [65] performing extensive parameter tuning and using a larger ukWaC corpus of which Wikipedia is only one part.

A benchmark of multiple algorithms on SimLex-999 and its subset was recently performed by Banjade et al. [66]. Comparison of results reported in Table 19 with our figures does not reveal substantial deviations. Banjade et al. [66] do not report on their configuration for WordNet measures. Our results show that substantial improvement can be gained by synset similarity maximization. The NNLM result reported by Banjade et al. [66] is for skip-gram model on the Google News corpus. For SimLex-666 the result reported in Banjade et al. [66] exactly matches ours. For ESA on SimLex-999, they state correlation about 0.04 higher than our, which is reported in Table 17. This difference can be possibly attributed to a more recent/larger Wikipedia snapshot used (year of snapshot not reported in the paper).

Interestingly, the maximum Spearman correlation reported in Banjade et al. [66] on SimLex-666 for a single method or a combination of methods is for the UMBC system [68]. The attained correlation 0.59 equals the correlation that we obtained with the JCn measure with the Synset Similarity Maximization option.

8. Conclusions

It is probably not possible to create a universal dataset for benchmarking word similarity algorithms which would equally well fit all applications. In the related domain of machine learning, this problem is addressed by involving large number of datasets in algorithm evaluations, with the latest benchmarking initiatives containing 100 diverse datasets [69] or even more. To advance the field of similarity computation, not only greater variety of benchmarking resources is needed, but these also need to be better understood and described in terms of the similarity definition used and their lexical content. We performed such analysis for SimLex-999 and WordSim353, commonly used datasets for benchmarking word similarity and relatedness. To contribute to the diversity of evaluation resources for word similarity computation, we propose to adapt the paradigmatic association definition of similarity, which is more permissive of antonymy as a similarity relation. We designed two guidelines based on this approach, one eliciting word interchangeability scores and the second one eliciting explicit similarity ratings.

The main limitation of the presented lexical resources is that the word interchangeability approach to measuring similarity is novel and its caveats are not yet well studied. Regarding the presented datasets, the deficiencies of WIN353 include its relatively small size and the fact that the word pairs were not selected in a transparent way in the underlying WordSim353 dataset. Also, the inclusion of raters from multiple countries contributed to the low inter-rater agreement, which may limit the strengths of the conclusions drawn from algorithm improvements above the inter-rater agreement. The biggest limitation is that we obtained low similarity scores for antonyms, which we attribute to overly strict definition of word interchangeability that we incorporated into the guidelines. A partial remedy is provided by the two other contributed datasets: the larger WinLex999cs based on lexical pairs in SimLex-999, and ES353, which was annotated with guidelines explicitly addressing similarity as antonymy.

As auxiliary resources, we present several other datasets. The re-annotation of the original SimLex-999 in Czech can help research in the multilingual domain. The crowdsourced reannotation of WordSim353 in English with raters spanning multiple countries may be found useful for some replication studies.

The paper also includes a benchmark of common WordNet-based and distributional measures, which is focused on analyzing the differences in their performance on similarity and relatedness datasets. For similarity datasets the main conclusion are that WordNet similarity measures perform equally well as state-of-the-art distributional models trained on Wikipedia. The hardest similarity dataset in the evaluation for WordNet-measures was WIN353 and the hardest for the distributional measures was SimLex-666 (noun subset of SimLex-999). For NNLM models we observed that smaller window size models similarity better. For relatedness datasets, the best performance is surprisingly obtained by the ESA algorithm, which outperforms all measures on WordSim353 including its WSRel subset. Overall, the performance of all distributional measures was quite leveled. In contrast, the correlations obtained by the WordNet measures were substantially lower and more varied – with one exception: all WordNet similarity measures obtained on the WSRel subset of WordSim353 nearly zero Spearman correlation, which indicates that the benchmarked WordNet similarity measures do not measure relatedness.

As a possible direction of future work we consider using a revised version of the interchangeability guidelines to annotate the SimLex-999 in English and part of the BLESS dataset. Word pairs in the BLESS dataset are associated with additional lexical and relationship information, which would allow for more insight into the performance of the evaluated algorithms.

This article contains all the presented datasets along with the annotation guidelines in the supplementary material. Additional resources can be found at http://ner.vse.cz/datasets/win353/. All datasets are licensed under Attribution 4.0 International (CC BY 4.0) license.

Acknowledgements

The authors would like to thank the anonymous reviewers for their feedback. This research was partially supported by the European Union's 7th Framework Programme via the LinkedTV project (FP7-287911) and by long term institutional support of research activities by Faculty of Informatics and Statistics, University of Economics, Prague.

The contributions of the authors is as follows. TK administered the annotation tasks. TK and OZ analyzed the data from crowdsourcing. TK performed the benchmarks with WordNet-based measures and the BOW, and OZ with ESA. Both authors worked on experiments with NNLM. TK authored and translated the WIN353 guidelines and OZ translated the SimLex-999 guidelines and served as the adjudicator for translation of SimLex-999. TK conceived the research and wrote the paper. TK and OZ edited the paper.

Appendix A. Supplementary data

Supplementary data related to this article can be found at https://doi.org/10.1016/j.datak.2018.03.004.

References

[1] Z. Harris, Distributional structure, Word 10 (23) (1954) 146–162.

[2] L. Finkelstein, E. Gabrilovich, Y. Matias, E. Rivlin, Z. Solan, G. Wolfman, E. Ruppin, Placing search in context: the concept revisited, ACM Trans. Inf. Syst. 20 (1) (January 2002) 116–131.

[3] F. Hill, R. Reichart, A. Korhonen, Simlex-999: evaluating semantic models with (genuine) similarity estimation, Comput. Linguist. 41 (4) (2015) 665–695.

[4] E. Agirre, E. Alfonseca, K. Hall, J. Kravalová, M. Paşca, A. Soroa, A study on similarity and relatedness using distributional and WordNet-based approaches, in: Proceedings of Human Language Technologies: the 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics. NAACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 19–27.

[5] C. Chiarello, C. Burgess, L. Richards, A. Pollock, Semantic and associative priming in the cerebral hemispheres: some words do, some words don't…sometimes, some places, Brain Lang. 38 (1) (1990) 75–104.

[6] G. Lapesa, S. Evert, S.S. im Walde, Contrasting syntagmatic and paradigmatic relations: insights from distributional semantic models, in: Proceedings of the 3rd Joint Conference on Lexical and Computational Semantics. *SEM 2014, Association for Computational Linguistics, Stroudsburg, PA, USA, 2014, pp. 160–170.

[7] P.D. Turney, P. Pantel, et al., From frequency to meaning: vector space models of semantics, J. Artif. Intell. Res. 37 (1) (2010) 141–188.

[8] A. Tversky, Features of similarity, Psychol. Rev. 84 (1977) 327–352.

[9] S. Scheible, S.S. im Walde, S. Springorum, Uncovering distributional differences between synonyms and antonyms in a word space model, in: Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013, Asian Federation of Natural Language Processing/ ACL, 2013, pp. 489–497.

[10] C. Willners, Antonyms in Context : a Corpus-based Semantic Analysis of Swedish Descriptive Adjectives, (Ph.D. thesis), Lund University, 2001.

[11] R. Rapp, The computation of word associations: comparing syntagmatic and paradigmatic approaches, in: Proceedings of the 19th International Conference on Computational Linguistics, COLING '02, vol. 1, Association for Computational Linguistics, Stroudsburg, PA, USA, 2002, pp. 1–7.

[12] I. Leviant, R. Reichart, Separated by an Un-common Language: towards Judgment Language Informed Vector Space Modeling, arXiv preprint arXiv:1508.00106, 2015.

[13] M. Sahlgren, The distributional hypothesis. from context to meaning: distributional models of the lexicon in linguistics and cognitive science, Rivista Linguist. 20 (1) (2008).

[14] P. Resnik, Using information content to evaluate semantic similarity in a taxonomy, in: Proceedings of the 14th International Joint Conference on Artificial Intelligence, IJCAI'95, vol. 1, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1995, pp. 448–453.

[15] E. Gabrilovich, S. Markovitch, Computing semantic relatedness using Wikipedia-based explicit semantic analysis, in: Proceedings of the 20th International Joint Conference on Artifical Intelligence. IJCAI'07, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007, pp. 1606–1611.

[16] D. Milne, I.H. Witten, An effective, low-cost measure of semantic relatedness obtained from Wikipedia links, in: Proceedings of the First AAAI Workshop on Wikipedia and Artificial Intelligence, 2008, pp. 25–30.

[17] G. Pirró, J. Euzenat, A feature and information theoretic framework for semantic similarity and relatedness, in: Proceedings of the 9th International Semantic Web Conference on the Semantic Web - Volume Part I. ISWC'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 615–630.

[18] M. Strube, S.P. Ponzetto, WikiRelate! computing semantic relatedness using Wikipedia, in: Proceedings of the 21st National Conference on Artificial Intelligence, AAAI'06, vol. 2, AAAI Press, 2006, pp. 1419–1424.

[19] A. Budanitsky, G. Hirst, Evaluating WordNet-based measures of lexical semantic relatedness, Comput. Linguist. 32 (1) (Mar. 2006) 13–47.

[20] D. Gentner, A.B. Markman, Structural alignment in comparison - no difference without similarity, Psychol. Sci. 5 (3) (1994) 152–158.

[21] D. Gentner, A. Markman, Structure mapping in analogy and similarity, Am. Psychol. 52 (1) (1997) 45–56.

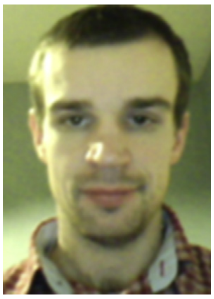
[22] D. Gentner, Structure-mapping: a theoretical framework for analogy, Cognit. Sci. 7 (2) (1983) 155–170.

[23] K. Lund, C. Burgess, R.A. Atchley, Semantic and associative priming in high-dimensional semantic space, in: Proceedings of the 17th Annual Conference of the Cognitive Science Society, 1995, pp. 660–665.

[24] K.A. Hutchison, Is semantic priming due to association strength or feature overlap? a microanalytic review, Psychonomic Bull. Rev. 10 (4) (2003) 785–813.

[25] D. Nelson, C. McEvoy, T. Schreiber, The University of South Florida free association, rhyme, and word fragment norms, Behav. Res. Meth. Instrum. Comput. 36 (3) (2004) 402–407.
[26] F. Saussure, Cours de linguistique générale, Payot, Paris, 1916.
[27] E. Agirre, A. Soroa, Personalizing PageRank for word sense disambiguation, in: Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics. EACL '09, Association for Computational Linguistics, Stroudsburg, PA, USA, 2009, pp. 33–41.
[28] EPI, EF EPI English Proficiency Index, 2014. Available from: www.ef.com.
[29] Longman, Longman Communication 3000 (Longman Dictionary of Contemporary English), 2007.
[30] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia - a crystallization point for the web of data, Web Semant 7 (3) (Sep. 2009) 154–165.
[31] M. Jarmasz, S. Szpakowicz, Roget's thesaurus and semantic similarity, in: Conference on Recent Advances in Natural Language Processing, 2003, pp. 212–219.
[32] S. Cinková, Wordsim353 for Czech, in: Text, Speech and Dialogue, Proceedings of the 19th International Conference TSD 2016. Lecture Notes in Artificial Intelligence, Springer, Berlin-Heidelberg, Germany, 2016.
[33] G. Halawi, G. Dror, E. Gabrilovich, Y. Koren, Large-scale learning of word relatedness with constraints, in: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '12, ACM, New York, NY, USA, 2012, pp. 1406–1414.
[34] E. Bruni, N.-K. Tran, M. Baroni, Multimodal distributional semantics, J. Artif. Intell. Res. (JAIR) 49 (2014) 1–47.
[35] H. Rubenstein, J.B. Goodenough, Contextual correlates of synonymy, Commun. ACM 8 (10) (Oct. 1965) 627–633.
[36] G.A. Miller, W.G. Charles, Contextual correlates of semantic similarity, Lang. Cognit. Process. 6 (1) (1991) 1–28.
[37] D. Milne, I.H. Witten, An open-source toolkit for mining Wikipedia, Artif. Intell. 194 (2013) 222–239.
[38] R. Feldman, J. Sanger, The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data, Cambridge University Press, Dec. 2006.
[39] J.J. Jiang, D.W. Conrath, Semantic similarity based on corpus statistics and lexical taxonomy, in: Proceedings of the International Conference on Research in Computational Linguistics, 1997, pp. 19–33.
[40] D. Lin, An information-theoretic definition of similarity, in: Proceedings of the 15th International Conference on Machine Learning, Morgan Kaufmann, 1998, pp. 296–304.
[41] G. Pirró, A semantic similarity metric combining features and intrinsic information content, Data Knowl. Eng. 68 (11) (2009) 1289–1308.
[42] E. Gabrilovich, S. Markovitch, Wikipedia-based semantic interpretation for natural language processing, J. Artif. Int. Res. 34 (1) (Mar. 2009) 443–498.
[43] T. Mikolov, I. Sutskever, K. Chen, G.S. Corrado, J. Dean, Distributed representations of words and phrases and their compositionality, in: Advances in Neural Information Processing Systems, 2013, pp. 3111–3119.
[44] M. Sahlgren, The Word-space Model: Using Distributional Analysis to Represent Syntagmatic and Paradigmatic Relations between Words in High-dimensional Vector Spaces, (Ph.D. thesis), Stockholm University, Sweden, 2006.
[45] Y. Peirsman, K. Heylen, D. Geeraerts, Size matters: tight and loose context definitions in English word space models, in: Proceedings of the ESSLLI Workshop on Distributional Lexical Semantics Pages. Hamburg, Germany, August 2008, pp. 34–41.
[46] T. Landauer, S. Dumais, A solution to Plato's problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge, Psychol. Rev. 104 (2) (1997) 211–240.
[47] R. Rapp, Word sense discovery based on sense descriptor dissimilarity, in: Proceedings of the 9th Machine Translation Summit, 2003, pp. 315–322.
[48] J. Bullinaria, J. Levy, Extracting semantic representations from word co-occurrence statistics: a computational study, Behav. Res. Meth. 510 (3) (2007).
[49] K. Radinsky, E. Agichtein, E. Gabrilovich, S. Markovitch, A word at a time: computing word relatedness using temporal semantic analysis, in: Proceedings of the 20th International Conference on World Wide Web. WWW '11, ACM, New York, NY, USA, 2011, pp. 337–346.
[50] M.D. Lee, M. Welsh, An empirical evaluation of models of text document similarity, in: CogSci2005, Erlbaum, 2005, pp. 1254–1259.
[51] A. Panchenko, O. Morozova, A study of hybrid similarity measures for semantic relation extraction, in: Proceedings of the Workshop on Innovative Hybrid Approaches to the Processing of Textual Data. HYBRID '12, Association for Computational Linguistics, Stroudsburg, PA, USA, 2012, pp. 10–18.
[52] M. Baroni, A. Lenci, How we blessed distributional semantic evaluation, in: Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics. GEMS '11, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, pp. 1–10.
[53] N. Neubauer, N. Haldenwang, O. Vornberger, Differences in semantic relatedness as judged by humans and algorithms, in: Proceedings of the 6th Language & Technology Conference, 2013.
[54] C. Biemann, Turk bootstrap word sense inventory 2.0: a large-scale resource for lexical substitution, in: Language Resources and Evaluation (LREC), 2012, pp. 4038–4042.
[55] D. Bär, T. Zesch, I. Gurevych, Composing Measures for Computing Text Similarity, Technical report, TU Darmstadt, 2015.
[56] R. Snow, B. O'Connor, D. Jurafsky, A.Y. Ng, Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2008, pp. 254–263.
[57] S. Sen, M.E. Giesel, R. Gold, B. Hillmann, M. Lesicko, S. Naden, J. Russell, Z.K. Wang, B. Hecht, Turkers, scholars, "Arafat" and "peace": cultural communities and algorithmic gold standards, in: Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work and Social Computing. CSCW '15, ACM, New York, NY, USA, 2015, pp. 826–838.
[58] T. Zesch, I. Gurevych, Automatically creating datasets for measures of semantic relatedness, in: Proceedings of the Workshop on Linguistic Distances, Association for Computational Linguistics, 2006, pp. 16–24.
[59] A. Panchenko, D. Ustalov, N. Arefyev, D. Paperno, N. Konstantinova, N. Loukachevitch, C. Biemann, Human and machine judgements for Russian semantic relatedness, in: International Conference on Analysis of Images, Social Networks and Texts, Springer, 2016, pp. 221–235.
[60] D. Crystal, English as a Global Language, second ed., Cambridge Univ. Press, Cambridge, 2003.
[61] E.A. Anchimbe, The Native-speaker Fever in English Language Teaching (ELT): Pitting Pedagogical Competence against Historical Origin, Linguistik online (1), 2006.
[62] H.T. Ng, C. Yong, K.S. Foo, A case study on inter-annotator agreement for word sense disambiguation, in: Proceedings of the ACL SIGLEX Workshop on Standardizing Lexical Resources (SIGLEX99). College Park, Maryland, 1999, pp. 9–13.
[63] M. Thelwall, K. Buckley, Topic-based sentiment analysis for the social web: the role of mood and issue-related words, J. Am. Soc. Inf. Sci. Technol. 64 (8) (2013) 1608–1617.
[64] J. Chen, G. de Melo, Semantic information extraction for improved word embeddings, in: Proceedings of the NAACL Workshop on Vector Space Modeling for NLP, 2015.
[65] M. Baroni, G. Dinu, G. Kruszewski, Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors, in: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, vol. 1, Association for Computational Linguistics, Baltimore, Maryland, June 2014, pp. 238–247, (Long Papers).
[66] R. Banjade, N. Maharjan, N. Niraula, V. Rus, D. Gautam, Lemon and tea are not similar: measuring word-to-word similarity by combining different methods, in: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing, Lecture Notes in Computer Science, vol. 9041, Springer International Publishing, 2015, pp. 335–346, https://doi.org/10.1007/978-3-319-18111-0_25.
[67] G. Lapesa, S. Evert, A large scale evaluation of distributional semantic models: parameters, interactions and model selection, Trans. Assoc. Comput. Linguist. 2 (2014) 531–545.
[68] L. Han, A.L. Kashyap, T. Finin, J. Mayfield, J. Weese, UMBC EBIQUITY-CORE: semantic textual similarity systems, in: Proceedings of the Second Joint Conference on Lexical and Computational Semantics, Association for Computational Linguistics, June 2013.
[69] B. Bischl, G. Casalicchio, M. Feurer, F. Hutter, M. Lang, R.G. Mantovani, J.N. van Rijn, J. Vanschoren, Openml Benchmarking Suites and the Openml100, arXiv preprint arXiv:1708.03731, 2017.

Tomáš Kliegr earned a Ph.D. from the University of Economics, Prague for research on the use of WordNet and Wikipedia-based measures in entity classification (2012), and a Ph.D. from the School of Computer Science, Queen Mary University of London for research on the role of cognitive biases in interpretability of rule learning results (2017). His joint work with Ondřej Zamazal and Václav Zeman on the Linked Hypernyms Dataset received the first DBpedia Text Extraction Challenge prize in 2017. Dr. Kliegr is serving, or has served, as a program committee member of a number of conferences in the field of machine learning, artificial intelligence, and semantic data processing, such as ECML/PKDD, IJCAI/ECAI, ISWC (PD), ESWC (PD) and RuleML.



Ondřej Zamazal is Researcher and Lecturer at the University of Economics, Prague (UEP), Department of Information and Knowledge Engineering, where he also obtained the PhD degree in 2010. His main research topics are ontology engineering and ontology matching. He participated in EU projects such as K-Space, Knowledge Web, LOD2, LinkedTV and OBEU and undertook research internships at INRIA Rhone-Alps, France and University of Mannheim, Germany. He is holder of the Josef Hlávka Award (2006), author of about 50 refereed publications and co-organiser of the OAEI initiative, PC member of a number of conferences, including top-class ones such as ISWC or EKAW.

Appendix D: EntityClassifier.eu: real-time classification of entities in text with Wikipedia

C1 Dojchinovski M., Kliegr T. (2013) EntityClassifier.eu: Real-Time Classification of Entities in Text with Wikipedia. In: Blockeel H., Kersting K., Nijssen S., Železný F. (eds) Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2013. Lecture Notes in Computer Science, vol 8190. Springer, Berlin, Heidelberg. ISBN 978-3-642-40993-6. DOI `https://doi.org/10.1007/978-3-642-40994-3_48`

Entityclassifier.eu: Real-Time Classification of Entities in Text with Wikipedia

Milan Dojchinovski[1,2] and Tomáš Kliegr[2]

[1] Web Engineering Group
Faculty of Information Technology
Czech Technical University in Prague
milan.dojchinovski@fit.cvut.cz

[2] Department of Information and Knowledge Engineering
Faculty of Informatics and Statistics
University of Economics, Prague, Czech Republic
tomas.kliegr@vse.cz

Abstract. Targeted Hypernym Discovery (THD) performs unsupervised classification of entities appearing in text. A hypernym mined from the free-text of the Wikipedia article describing the entity is used as a class. The type as well as the entity are cross-linked with their representation in DBpedia, and enriched with additional types from DBpedia and YAGO knowledge bases providing a semantic web interoperability. The system, available as a web application and web service at `entityclassifier.eu`, currently supports English, German and Dutch.

1 Introduction

One of the most significant challenges in text mining is the dimensionality and sparseness of the textual data. In this paper, we introduce Targeted Hypernym Discovery (THD), a Wikipedia-based entity classification system which identifies salient words in the input text and attaches them with a list of more generic words and concepts at varying levels of granularity. These can be used as a lower dimensional representation of the input text.

In contrast to the commonly used dimensionality reduction techniques, such as PCA or LDA, which are sensitive to the amount of data, THD provides the same quality of output for all sizes of input text, starting from just one word. Since THD extracts these types from Wikipedia, it can also process infrequent, but often information-rich words, such as named entities. Support for live Wikipedia mining is a unique THD feature allowing coverage of "zeitgeist" entities which had their Wikipedia article just established or updated.

THD is a fully unsupervised algorithm. A class is chosen for a specific entity as the one word (concept) that best describes its type according to the consensus of Wikipedia editors. Since the class (so as the entity) is mapped to DBpedia, the semantic knowledge base, one can traverse up the taxonomy to the desired class granularity. Additionally, the machine-readable information obtainable on the disambiguated entity and class from DBpedia and YAGO can be used for feature enrichment.

H. Blockeel et al. (Eds.): ECML PKDD 2013, Part III, LNAI 8190, pp. 654–658, 2013.

2 Architecture

THD is implemented in Java on top of the open source GATE framework[1].

Entity extraction module identifies entity candidates (noun phrases) in the input text. Depending on setting, entities can be restricted to named entities ("Diego Maradona") or common entities ("football").

Disambiguation module assigns entity candidate with a Wikipedia entry describing it. This module combines textual similarity between the entity candidate and article title with the importance of the article.

Entity classification module assigns each entity with one or more hypernyms. The hypernyms are mined with the THD algorithm (see Sec. 3) from the Wikipedia articles identified by the Disambiguation module. This mining is performed either on-line from live Wikipedia or from a Wikipedia mirror. The default option is to use the Linked Hypernyms Dataset, which contains 2.5 million article-hypernym pairs precomputed from a Wikipedia mirror.

Semantization module maps the entity as well as the class to `DBpedia.org` concepts. A "semantic enrichment" is also performed: once the entity is mapped, additional types are attached from DBpedia [1] and YAGO [2], the two prominent semantic knowledge bases. The final set of types returned for an entity thus contains the "linked hypernym" (hypernym mapped to DBpedia obtained with THD), and a set of DBpedia and YAGO types.

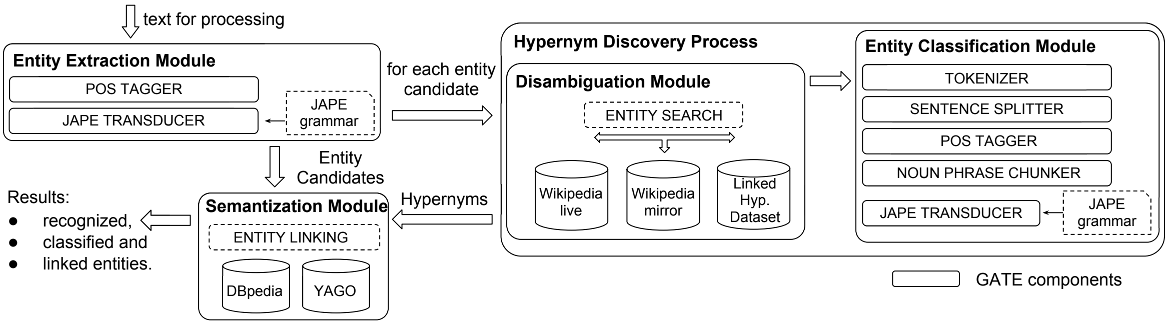


Fig. 1. Architecture overview

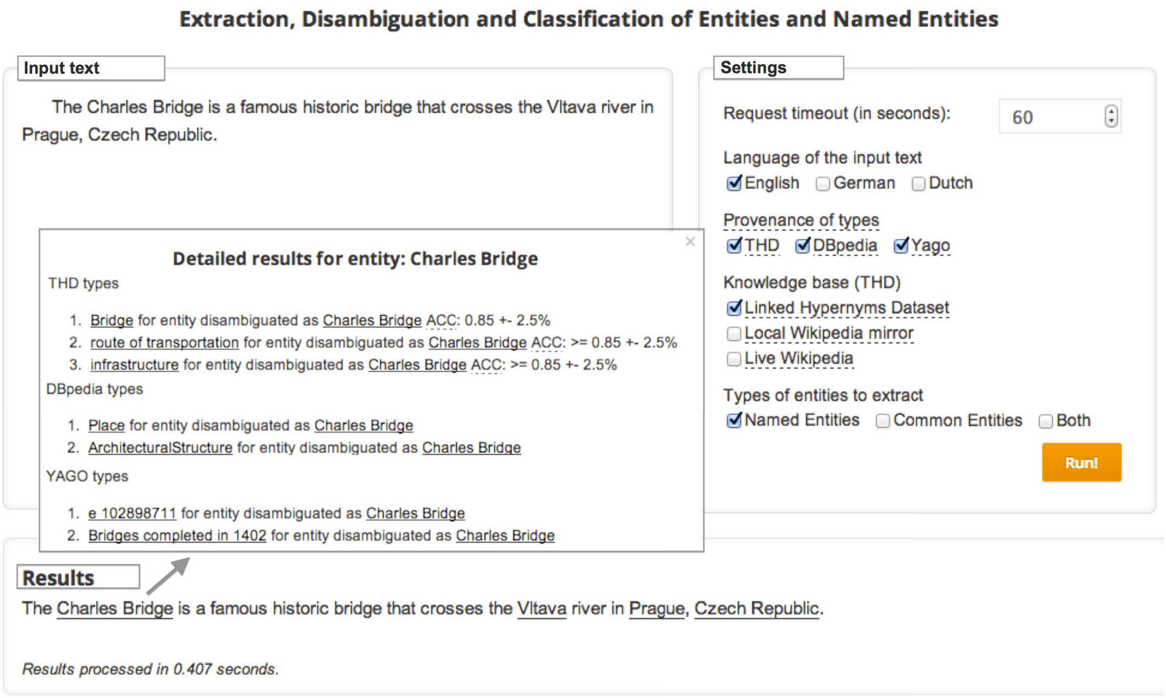
3 Hypernym Discovery Algorithm and Benchmark

Hypernym discovery is performed with hand-crafted lexico-syntactic patterns. These were in the past primarily used on larger text corpora with the intent to discover all word-hypernym pairs in the collection [7]. With *Targeted* Hypernym Discovery we apply lexico-syntactic patterns on a *suitable document* (Wikipedia article) with the intent to extract *one hypernym* at a time (details in [3,4]).

THD performance was measured on the following benchmarks independent on the input text: a) discovering correct hypernym given a Wikipedia article, b) linking hypernym to a semantic web identifier. The outcome of the evaluation[2]

[1] `http://gate.ac.uk`

[2] The results and the "High accuracy dataset" are available at `http://ner.vse.cz/datasets/linkedhypernyms/`.



Extraction, Disambiguation and Classification of Entities and Named Entities

Input text

The Charles Bridge is a famous historic bridge that crosses the Vltava river in Prague, Czech Republic.

Settings

Request timeout (in seconds): 60

Language of the input text

☑ English □ German □ Dutch

Provenance of types

☑ THD ☑ DBpedia ☑ Yago

Knowledge base (THD)

☑ Linked Hypernyms Dataset

□ Local Wikipedia mirror

□ Live Wikipedia

Types of entities to extract

☑ Named Entities □ Common Entities □ Both

Detailed results for entity: Charles Bridge

THD types

1. Bridge for entity disambiguated as Charles Bridge ACC: 0.85 +- 2.5%
2. route of transportation for entity disambiguated as Charles Bridge ACC: >= 0.85 +- 2.5%
3. infrastructure for entity disambiguated as Charles Bridge ACC: >= 0.85 +- 2.5%

DBpedia types

1. Place for entity disambiguated as Charles Bridge
2. ArchitecturalStructure for entity disambiguated as Charles Bridge

YAGO types

1. e 102898711 for entity disambiguated as Charles Bridge
2. Bridges completed in 1402 for entity disambiguated as Charles Bridge

Results

The Charles Bridge is a famous historic bridge that crosses the Vltava river in Prague, Czech Republic.

Results processed in 0.407 seconds.

Fig. 2. Screenshot of the system (edited to fit the page)

altogether on 16.500 entity articles (English, German, Dutch) is reported in [3]. The best results were obtained for the German person subset, with precision 0.98 and recall 0.95. This is on par with the the best results in the respective metrics recently reported in [5]: 0.97 precision for lexico-syntactic patterns and 0.94 recall for Syntactic-Semantic Tagger. The overall accuracy of discovering plain text (linked) hypernyms for English is 0.95 (0.85), for Dutch 0.93 (0.88) and German 0.95 (0.77). These numbers provide a lower bound on the error of THD, since they do not include the entity recognition error and particularly the disambiguation error (matching entity with a Wikipedia article).

4 Comparison with Related Systems

While techniques for Named Entity Recognition and classification (NER) are well-researched, NER classifiers typically need to be trained on large labeled document corpora, which generally involve only several labels, making them unsuitable for dimensionality reduction. Replacement of “Maradona” with “Person” loses too much meaning for most applications. The recent shift from human-annotated corpora to Wikipedia in some systems allows to provide types with finer granularity, and also broadening of the scope to “common” entities. In this section (and accompanying screencasts), we present a comparison with two best-known academic systems DBpedia Spotlight [6] and AIDA [8].

Real-time Mining. THD directly incorporates a text mining algorithm. Once an entity is disambiguated to a Wikipedia article, the system retrieves the article from Wikipedia and extracts the hypernym from its free text. The mining speed is about 1 second per entity including network overhead. This allows to discover types for entities, which had their article only recently added to Wikipedia, or adapt to changes in Wikipedia. The authors are not aware of any other system that incorporates query-time Wikipedia mining. AIDA and DBpedia Spotlight lookup the disambiguated entity in a database of types.

Complementarity to other Systems. Since THD extracts the types from *free text*, the results are largely complementary to types returned by other Wikipedia-based systems. These typically rely on DBpedia or YAGO knowledge-bases, which are populated from article *categories* and *"infoboxes"*, the semistructured information in Wikipedia. As a convenience, THD returns types from DBpedia and YAGO in addition to the mined hypernym. The complementary character of the results can be utilized for classifier fusion.

Right Granularity. For many entities DBpedia and YAGO-based systems provide a long list of possible types. For example, DBpedia assigns Diego Maradona with 40 types including `dbpedia-owl:SoccerManager`, `foaf:Person` as well as the highly specific `yago:1982FIFAWorldCupPlayers`. THD aids the selection of the "right granularity" by providing the most frequent type, as selected by Wikipedia editors for inclusion into the article's first sentence. For Maradona, as of time of writing, THD returns "manager".[3]

Multilinguality. System currently supports English, Dutch and German, extensibility to a new language requires only providing two JAPE grammars and plugging in correct POS tagger (ref. to Fig. 2). DBpedia Spotlight and AIDA support only English.

Acknowledgements. This research was supported by the European Union's 7th Framework Programme via the LinkedTV project (FP7-287911) and CTU in Prague grant (SGS13/100/OHK3/1T/18).

References

1. Bizer, C., et al.: DBpedia - a crystallization point for the web of data. Web Semant 7(3), 154–165 (2009)
2. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: YAGO2: A spatially and temporally enhanced knowledge base from Wikipedia. Artificial Intelligence 194, 28–61 (2013)
3. Kliegr, T., Dojchinovski, M.: Linked hypernyms: Enriching DBpedia with Targeted Hypernym Discovery (Submitted)
4. Kliegr, T., et al.: Combining captions and visual analysis for image concept classification. In: MDM/KDD 2008. ACM (2008)
5. Litz, B., Langer, H., Malaka, R.: Sequential supervised learning for hypernym discovery from Wikipedia. In: Fred, A., Dietz, J.L.G., Liu, K., Filipe, J. (eds.) IC3K 2009. CCIS, vol. 128, pp. 68–80. Springer, Heidelberg (2011)

[3] As demonstrated in [4], the algorithm used can also return multi-word hypernyms ("soccer manager"). This feature is not yet available in THD.

6. Mendes, P.N., Jakob, M., Garcia-Silva, A., Bizer, C.: DBpedia spotlight: Shedding light on the web of documents. In: I-Semantics (2011)
7. Snow, R., Jurafsky, D., Ng, A.Y.: Learning syntactic patterns for automatic hypernym discovery. In: Advances in Neural Information Processing Systems, vol. 17, pp. 1297–1304. MIT Press, Cambridge (2005)
8. Yosef, M.A., et al.: AIDA: An online tool for accurate disambiguation of named entities in text and tables. PVLDB 4(12), 1450–1453 (2011)

Part III.

Rule Learning

Appendix E: Learning business rules with association rule classifiers

C2 Kliegr T., Kuchař J., Sottara D., Vojíř S. (2014) Learning Business Rules with Association Rule Classifiers. In: Bikakis A., Fodor P., Roman D. (eds) Rules on the Web. From Theory to Applications. RuleML 2014. Lecture Notes in Computer Science, vol 8620. Springer, Cham. ISBN 978-3-319-09869-2. DOI https://doi.org/10.1007/978-3-319-09870-8_18

Learning Business Rules with Association Rule Classifiers

Tomáš Kliegr[1,4], Jaroslav Kuchař[1,2], Davide Sottara[3], and Stanislav Vojíř[1]

[1] Department of Information and Knowledge Engineering,
Faculty of Informatics and Statistics,
University of Economics, Prague, Czech Republic
first.last@vse.cz

[2] Web Engineering Group, Faculty of Information Technology,
Czech Technical University in Prague, Czech Republic

[3] Biomedical Informatics Department,
Arizona State University, Phoenix, AZ, USA
dsottara@asu.edu

[4] Multimedia and Vision Research Group,
Queen Mary, University of London, UK

Abstract. The main obstacles for a straightforward use of association rules as candidate business rules are the excessive number of rules discovered even on small datasets, and the fact that contradicting rules are generated. This paper shows that Association Rule Classification algorithms, such as CBA, solve both these problems, and provides a practical guide on using discovered rules in the Drools BRMS and on setting the ARC parameters. Experiments performed with modified CBA on several UCI datasets indicate that data coverage rule pruning keeps the number of rules manageable, while not adversely impacting the accuracy. The best results in terms of overall accuracy are obtained using minimum support and confidence thresholds. Disjunction between attribute values seem to provide a desirable balance between accuracy and rule count, while negated literals have not been found beneficial.

Keywords: association rules, rule pruning, business rules, Drools.

1 Introduction

Association rule learning cannot be directly used for learning business rules, due to the excessive number of rules generated even for small datasets, and the lack of a rule conflict resolution strategy. However, if several techniques originally developed for association rule classification (ARC) are adopted, association rules can be used as classification business rules. ARC algorithms contain a rule pruning step, which significantly reduces the number of rules, and define a conflict resolution strategy for cases when one object is matched by multiple rules.

This paper has two focus areas. Due to the limited amount of prior work, in the first part of the paper we evaluate to what degree ARC algorithms meet the requirements of the business rule learning task and demonstrates how the

A. Bikakis et al. (Eds.): RuleML 2014, LNCS 8620, pp. 236–250, 2014.

discovered rules can be used in a Drools Business Rule Management System (BRMS) system. The second part of the paper describes our implementation and experimental evaluation of a business rule learning system. In contrast to mainstream ARC algorithms, the system allows to learn disjunctive and negative rules. We hypothesize that the additional expressiveness could result in a rule set which is smaller, and thus more intelligible for the business user. Another modification is a simplification of the rule pruning phase.

This paper is organized as follows. Section 2 reviews related research. Section 3 presents a set of requirements on business rule learning algorithm and contrasts it with what ARC algorithms provide. Section 4 describes how rules learnt from data can be used in the Drools. Section 5 presents our experimental business rule learning system *brCBA*. Section 6 presents experimental evaluation on several datasets. Finally, Section 7 summarizes our findings, gives limitations of the presented work and outlines viable directions of future research.

2 Related Work

There is a very limited amount of prior work on learning business rules from data. This paper is restricted to what we call *classification* business rules i.e. rules that assign a class (a type) to an object whenever its description matches the conditions contained in the rule's body. This corresponds to what is known in the rule learning literature as *classification rule* or *predictive rule*.

Association rule learning algorithms such as *apriori* [1] or FP-growth [3] can be used to learn conjunctive classification rules from data if the mining setup is constrained so that only the target class values can occur in the consequent of the rules. The GUHA method [7] is an alternative approach to mine association rules, which allows to learn also rules featuring negation and disjunction between attribute values.

The main obstacles for a straightforward use of association rules as candidate business rules are the excessive number of rules discovered even on small datasets, and the fact that contradicting rules are generated. Association Rule Classifier (ARC) algorithms provide an extension over association rule learning algorithms which address exactly these issues. These algorithms contain a rule pruning step, which significantly reduces the number of rules, and define a conflict resolution strategy for cases when one object is matched by multiple rules.

The first ARC algorithm dubbed CBA (Classification based on Associations) was introduced in 1998 by Liu et al. [5]. While there were multiple follow-up algorithms providing incremental improvements in classification performance (e.g. CPAR [15], CMAR [4] and MMAC [10]), the structure of most ARC algorithms follows that of CBA [13]: 1) learn association rules, 2) prune the set of classification rules, 3) classify new objects. Our proposed brCBA algorithm also follows this structure. It differs from CBA and other algorithms by using a GUHA-based algorithm in the "learn association rules" phase, which allows us to explore the effects of disjunction and negation on classification performance. To the best of our knowledge, the impact of the increased expressiveness added by these connectives on ARC performance has not yet been reported.

The output of association rule learning algorithms is determined typically by two parameters: minimum confidence and support thresholds on the training data. The confidence of a rule is defined as $a/(a+b)$, where a is the number of correctly classified objects, i.e. those matching rule antecedent as well rule consequent, and b is the number of misclassified objects, i.e. those matching the antecedent, but not the consequent. The support of a rule is defined as a/n, where n is the number of all objects (relative support), or simply as a (absolute support). The confidence threshold can be used to control the quality of the resulting classifier. While the authors of ARC classifiers report the confidence threshold used in their experimental setups (0.3 [10], 0.4 [9], 0.5 [5]), the impact of varying the value of this threshold on classifier performance has not yet been studied (to the best of our knowledge). To help guide the setting of ARC algorithms, we provide a detailed study of the effect of confidence threshold and support thresholds on the classification accuracy and rule count.

There is also a very limited work on effects of rule pruning. A qualitative review of rule pruning algorithms used in ARC are given e.g. in [13,8]. The effect of pruning on the size of the rule set is reported in [5], which presents evaluation on 26 UCI datasets. The average number of rules per dataset without pruning was 35,140, with pruning the average number of rules was reduced to 69. However, this paper focuses on the evaluation of less commonly employed pessimistic pruning. We focus on evaluation of data coverage pruning, which is the most commonly used pruning algorithm (present, with some modifications, in CBA, CMAR and MMAC).

3 Business Rule Learning Requirements

The business rule learning workflow imposes some specific demands on the selection of a suitable rule learning algorithm. In this section, we discuss the compliance of ARC algorithms with some of the requirements that we have identified.

BRMS Supported Rule Expressiveness. The rules learnt are composed of a conjunction of constraints on attribute values in the antecedent, and a single value for the class attribute in the consequent. The operations performed by later steps in ARC execution, such as pruning or ranking, do not change the internal structure of the rules.

Example 1. Rule learnt on the Iris dataset.

$\ulcorner$petalLength=$\langle 3.95; 4.54)$ $\wedge$ petalWidth=$\langle 1.3; 1.54)$ $\rightarrow_{1,0.14}$ Class=Iris-versicolor$\urcorner$, where 1 is rule confidence and 0.14 (relative) rule support.

Rules, such as the one depicted in Example 1, can be translated into technical rule languages for execution inside a rule engine. In our earlier work [14] we presented the mapping to DRL, the format used by the open source BRMS system Drools.

Small number of output rules. Perhaps the biggest challenge in converting association rules to business rules is the fact that the number of discovered rules is often too large to be presented to a user. The two common strategies to solve this problem are rule grouping and rule pruning.

Rule grouping algorithms cluster the rules according to a predefined distance measure [12]. Most ARC algorithms use rule pruning. The details of the individual types of pruning algorithms is given e.g. in [13,11,8]. The most commonly used method according to these survey papers is *Data Coverage Pruning* (see Subs. 5.2).

Exhaustive set of rules. Most ARC algorithms use an exact association rule learning algorithm, either based on apriori or FP-Growth. These algorithms learn exhaustive set of rules matching predefined minimum confidence and minimum support thresholds [13].

However, some rules are removed in the pruning phase. Since pruning[1] removes only rules which cover objects which are already covered by another higher priority rule, the pruning typically affects only rules that would be viewed by the user as redundant.

Rule conflict resolution. Once association rules are generated and pruned, ARC algorithms use them to classify new objects. There are two fundamental approaches: *single rule* and *multiple rule* classification [13], depending on the number of rules that are involved in assigning a class to an object. The *single rule* classification used in CBA is described in Section 5.3 and subject to experimental evaluation as part of our implementation in Section 6. An overview of possible implementation in the Drools Rule Engine is present in Section 4.

Ability to control rule quality. The rule quality can be controlled by setting the minimum confidence (and support) thresholds. It should be noted that ARC algorithms try to cover every training object with at least one rule, for example, CBA ensures this by adding a default rule to the rule set. The default rule insertion needs to be omitted (ref. to Subs. 5.2) in order to allow the user to control the overall quality of the rule set.

4 Drools-Based Rule Engine

The learning algorithm generates association rules which establish an implication between the antecedent and the consequent. In the case of classification rules, the consequent is the type of an individual object whose features have been matched by the antecedent. So, they can naturally be reinterpreted as business rules with the semantics of production rules. This allows to decouple recognition from decision making, resulting in more robust knowledge bases. Moreover, (production) rule engines can be considered commodity components: in particular, we have used the popular open source business logic platform Drools[2]. Drools is written in Java and relies on an object-oriented rule engine inspired from the RETE algorithm.

[1] Referring to the “database coverage” algorithm.

[2] `http://drools.jboss.org`

Listing 1.2. A Conflict Resolution Meta-Rule in Drools

```
rule 'Block by confidence'  @Direct
  when
    $m1 : Match( associationRole == 'premise', $t : tuple )
    $m2 : Match( this != $m1, associationRole == 'premise', tuple == $t,
                 confidence > $m1.confidence ||
                 confidence == $m1.confidence && support > $m1.support ||
                 antecedent < $m1.antecedent )
  then
    kcontext.cancelMatch( $m1 );
  end
```

In our implementation, we have created a simple, generic data model with two classes to model attributes and inferred types: `DrlObject` and `DrlAR` respectively. This allows to write rules such as the one in Listing 1.1.

Listing 1.1. A Sample Classification Rule in Drools

```
rule "rule_1" @associationRole(premise)
    @antecedent(4) @confidence(1) @support(0.06)
  when
    DrlObj( name == "petalLength", numVal >= 1 && < 1.59 )
    DrlObj( name == "petalWidth",  numVal >= 0.1 && < 0.34 )
    DrlObj( name == "sepalLength",
            numVal >= ( 4.3 && < 4.66 ) || ( >= 4.66 && < 5.02 ) )
    DrlObj( name == "sepalWidth", numVal >= 2.96 && < 3.2) )
  then
    DrlAR $type = new DrlAR( "rule_1", "Iris_Setosa", 4, 1, 0.06 );
    insertLogical( $type );
end
```

The rules are generated automatically from the output of the rule learner. Since the learner produces XML, we have applied an XSLT transformation to generate DRL, the Drools technical rule language. Notice that information such as confidence and support is retained as metadata and modelled using Java-like @annotations.

In order to implement the conflict resolution strategies mentioned in Section 5.2, we have exploited the "declarative agenda" feature of the rule engine. In a production rule engine, whenever one or more facts match the left-hand side of a rule, a rule activation is created and queued into an agenda. Activations are then consumed and the actions in the right-hand side are executed by the engine. Drools' declarative agenda allows to define rules that match and process the activations queued in the agenda itself. Such "meta-rules" are deployed into the same rule base as the standard rules. More specifically, entries in the agenda are instance of the class `Match`, which holds references to the rule that was activated as well as the tuple that caused the activation. Any metadata that is attached to the original rule is exposed by the engine as a virtual property of the activation, so that the meta-rule can constrain their value. Thanks to these capabilities, any conflict resolution strategy can be implemented with a single meta-rule, as shown in Listing 1.2. In our case, the activation of a rule with higher priority will cancel the activation of a rule with a lower priority for the same tuple.

5 brCBA - CBA for Business Rule Learning

In this section, we describe the setup used to perform the experimental evaluation. The implementation comes out of the seminal CBA algorithm. However, there are minor differences in individual steps, which are summarized in Table 1 and explained in the remainder of this section. Most importantly, brCBA uses for rule learning the LISp-Miner system[3], an implementation of the GUHA method, instead of the apriori algorithm.

Table 1. Comparison of CBA and brCBA

stage	**CBA [5]**	**brCBA**
learning	conjunctive rules (apriori)	conj. rules, disjunctions between attribute values, negations (GUHA method)
pruning	pessimistic pruning (optional), data coverage, default rule replacement	no pruning, data coverage pruning
classification	complete	partial

5.1 Rule Expressiveness

The mainstream systems for mining association rules employed in ARC, including CBA, output conjunctive association rules. The basic building block of an association rule is a literal.[4]

Definition 1. ***(literal)*** *A literal p is an attribute-value pair, taking the form of (A_i, v) in which A_i is an attribute and v a value. An object o satisfies a literal $p = (A_i, v)$ if and only if $o_i = v$, where o_i is the value of the i^{th} attribute of o.*

Definition 2. ***(rule)*** *A rule r, which takes the form of "$l_1 \wedge l_2, \wedge \ldots \wedge l_m \rightarrow c$", consists of a conjunction of literals $l_1, l_2, \ldots, l_m$, associated with a class label c. An object satisfies rule r's body if and only if it satisfies every literal in the rule. If object satisfies r's body, r predicts that the object is of class c. If a rule contains zero literal, its body is satisfied by any object.*

In brCBA we extend the original notion of literal present in Def. 1 to allow for disjunction between attribute values (dynamic binning) and negated literals.

Dynamic Binning (disjunctions between attribute values). Typically value binning is performed during the preprocessing step, creating a modified data table which contains a smaller number of merged values. This approach may negatively impact the quality of the rule learning if the bins created are too narrow or too broad. In brCBA we extend the definition of literal to allow for dynamic binning, which merges multiple values during *rule learning* into a value range (an enumeration of values or an interval).

[3] `http://lispminer.vse.cz`

[4] We introduce the definition of literal and an association rule from [15] substituting the machine learning term “tuple” by term “object” common in the BRMS field.

Definition 3. ***(positive literal)*** *A positive literal p is an association of an attribute with a value range, taking the form of (A_i, V) in which A_i is an attribute and V is a value range. An object o satisfies a positive literal $p = (A_i, V)$ if and only if $o_i \in V$, where o_i is a value of the i^{th} attribute of object o.*

From the options offered by the LISp-Miner system, we consider two types of dynamic binning: **Subset** binning merges up to a prespecified number of values, while **Sequence** (Interval) binning merges up to a prespecified number of *adjacent* values [7]. Subset binning is typically applied on on nominal attributes, while adjacent value binning on numerical or ordinal attributes.

The maximum number of values to be merged is set by parameter λ (for both methods). The result of dynamic binning on an attribute is a set of literals. Unlike some greedy algorithms (such as the algorithm for grouping values in C4.5 [6]), the dynamic binning operator is exhaustive. For an attribute A_i with n distinct values, assuming that $n \geq \lambda$, sequence binning creates $\sum_{j=1}^{\lambda} n - j + 1$ literals, while subset binning $\sum_{j=1}^{\lambda} \binom{n}{j}$ literals.

Example 2. Binning
The discretization on the petalLength attribute from the Iris dataset was performed by creating equidistant bins during preprocessing[a]: $[1; 1.59)$, $[1.59; 3.95)$, $[3.95; 4.54)$, $[4.54; 5.13)$, $[5.13; 5.72)$. Interval binning set to maximum length λ=2 will create 9 literals: five literals corresponding the original values plus the following four: $[1; 1.59) \vee [1.59; 3.95)$, $[1.59; 3.95) \vee [3.95; 4.54]$, $[3.95; 4.54) \vee [4.54; 5.13)$, $[4.54; 5.13) \vee [5.13; 5.72)$.
An example rule featuring dynamically binned intervals: $\ulcorner$petalLength = $[4.54; 5.13) \vee \langle 5.13; 5.72) \rightarrow_{0.77,0.33}$ Class=Iris-versicolor$\urcorner$,

[a] Merging bins with too small support count into one bin.

Negation. Considering negative literals in addition to the positive ones during rule mining produces a richer set of rules. It was previously conjectured that this could benefit the performance of ARC [2].

Definition 4. ***(negative literal)*** *A negative literal n is an association of an attribute with a value range, taking the form of (A_i, V) in which A_i is an attribute and V is a value range. An object o satisfies a negative literal $n = (A_i, V)$ if and only if $o_i \notin V$, where o_i is a value of the i^{th} attribute of o.*

Example 3. Rule with a negative literal
$\ulcorner\neg$petalLength=$[1; 1.59) \wedge$ petalWidth $[0.1; 0.34) \rightarrow_{1,0.05}$ Class=Iris-setosa$\urcorner$

5.2 Rule Pruning

CBA and brCBA use the *data coverage* rule pruning algorithm. This algorithm applies to a sorted list of ranked rules. Each rule is matched against the training

Algorithm 1. Data Coverage

Require: rules – sorted list of rules, T – set of objects in the training dataset
Ensure: rules – pruned list of rules

```
rules := sort rules according to criteria on Fig. 1
for all $rule \in rules$ do
    $matches$:= set of objects from $T$ that match both rule ant. and conseq.
    if matches==$\emptyset$ then
        remove $rule$ from $rules$
    else
        remove $matches$ from $T$
    end if
end for
return $rules$
```

data. If a rule does not correctly classify any object, it is discarded. Otherwise, the rule is kept, and the objects correctly classified are removed (ref. to Alg. 1).

The output of rule pruning is a reduced set of rules, where the redundant rules have been removed. If there are two rules matching one training object, the weaker rule (acc. to Fig. 1) will be removed.

1. r_a is ranked higher if confidence of r_a is greater than that of r_b,
2. r_a is ranked higher if confidence of r_a is the same as confidence of r_b, but support of r_a is greater than that of r_b,
3. r_a is ranked higher if r_a has shorter antecedent (fewer conditions) than r_b.

Fig. 1. Rule ranking criteria. Tie-breaking conditions applied if antecedents of two rules r_a and r_b match the same object.

It should be noted that the original CBA classifier contains two additional pruning steps: a) pessimistic pruning and b) replacement of rules performing worse than the majority class baseline with the default rule predicting the majority class. Pessimistic pruning is not featured in our setup, since it was not found to improve performance [5]. The omission of the default rule pruning in brCBA gives the user the control over the quality of the rule set, which can be influenced by the minimum confidence parameter, obtaining a *partial classifier* (not all objects may be labeled).

5.3 Classification and Rule Conflict Handling

If an input object matches exactly one rule, the classification step is very simple – the class contained in the consequent of the rule is assigned to the object. However, the output of association rule learning contains all too often an excessive number of redundant and conflicting rules. Employing rule pruning alleviates the number of conflicts since the number of redundant rules is reduced. Nevertheless pruning does not ensure that rule conflict will not emerge.

Rule conflict occurs if for a given object, there are at least two rules r_a and r_b, whose antecedents match the object. In practical terms, handling rule conflict is of importance if the consequents of these two rules are different, i.e. the rules assign a different class.

Association rules readily come with several scores that could be used to define a priority. These are primarily confidence and support, however additional measures such as chi-square or lift can be computed. The problem is thus to select, or combine these metrics into a total order, which would allow to solve ties between individual rules. brCBA uses the same method as CBA. In the first step, rules are sorted according to confidence, support and rule length – in the same way as in the data coverage pruning (see Fig. 1). The conflict is resolved by selecting the consequent of the top-ranked rule matching the object.

6 Experiments

The purpose of the experimental evaluation was to assess the impact of the following settings of association rule classifiers in the context of partial classification: data coverage rule pruning, dynamic binning, negated literals, and confidence/support thresholds.

6.1 Setup

Datasets. Experiments were performed on Iris, Balance Scale and Glass datasets from the UCI repository[5], which are frequently used for benchmarking classification systems. The use of a smaller number of datasets than in most related work allows us to present a detailed qualitative analysis of the results.
Preprocessing. Numerical attributes were discretized using equidistant binning with custom merging of bins with small support.
Rule Learning. To perform the experiments, we used the LISp-Miner system[6] for learning association rules. LISp-Miner allows to perform learning of negative and disjunctive rules. Disjunctive rules (dynamic binning) are learnt through the setting of the LISp-Miner coefficient feature on individual input attributes to *subset* or, respectively, *sequence* type. The maximum length parameter λ was set to 2.[7]
Rule Pruning. To perform rule pruning we used our Java implementation of the data coverage algorithm. This algorithm does not have any parameters.
Conflict Resolution. We used the conflict resolution according to Fig. 1.

[5] http://archive.ics.uci.edu/ml/

[6] http://lispminer.vse.cz

[7] The system allows to enter also the minimum length parameter, which was left set to 1. For experiments involving negative rules, the system was set to consider both positive and negative version for each literal. The remaining parameters of the LISp-Miner system were left at their default values.

6.2 Results

The experimental results achieved on individual datasets are depicted on Table 2-5 in terms of accuracy and rule count. Accuracy is computed as *correct*/N, where *correct* is the number of correct predictions and N the total number of objects.

Since brCBA is a partial classifier, it may not assign a label to all objects. For this reason, we also provide complementary results using precision, which we compute as *correct*/N_{cov}, where N_{cov} is the number of covered (classified) objects. The plots depicted on Figure 2-5 provide accuracy and precision at minimum confidence levels 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0 along with the average number of unclassified objects ($N - N_{cov}$).

All results are reported using ten fold cross validation with macro averaging.

Table 2. Dataset: Iris, minimum support threshold: 7 objects (5.18%)

	not pruned				pruned			
	without binning		sequence 1-2		without binning		sequence 1-2	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.5	96	0.940	972.2	0.940	20	0.920	17	**0.953**
0.6	87	0.940	903.6	0.940	19	0.920	17	**0.953**
0.7	83	0.940	839.6	0.940	17	0.920	17	**0.953**
0.8	76	0.940	734.7	0.940	17	0.920	15	0.947
0.9	68	0.900	603.2	0.940	15	0.880	14	0.940

Table 3. Dataset: Balance Scale, minimum support threshold: 10 objects (1.78%)

	not pruned				pruned			
	without binning		subset 1-2		without binning		subset 1-2	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.6	124	**0.891**	11947	0.758	78	0.870	153	0.779
0.7	86	0.875	8462	0.826	70	0.864	153	0.779
0.8	50	0.790	4881	0.838	50	0.782	153	0.779
0.9	24	0.547	2193	0.838	24	0.547	153	0.779
1.0	1	0.047	1001	0.811	1	0.047	99	0.758

Minimum Support and Confidence Thresholds. Experimental results show that the lower minimum support threshold is generally associated with improved accuracy. This is demonstrated on Table 5.

For Iris and Balance Scale datasets the precision and accuracy do not react to an increase of minimum confidence within a certain interval (Figure 2-4). This phenomenon is encountered without respect to whether the pruning is turned on or off. This can be explained by the fact that the mining output for a given minimum confidence threshold contains also the higher confidence rules. If these higher confidence rules cover all test objects that are covered by the lower confidence rules, due to the conflict resolution strategy used the lower confidence

Table 4. Dataset: Glass, minimum support threshold: 10 objects (5.18%)

	not pruned				pruned			
	positive only		with negations		positive only		with negations	
confidence	rules	accuracy	rules	accuracy	rules	accuracy	rules	accuracy
0.5	58.3	0.529	1418.8	0.492	25.8	**0.534**	44.3	0.519
0.6	31.8	0.464	838.5	0.492	21.1	0.464	42.4	0.492
0.7	10.3	0.290	416.7	0.449	8.4	0.286	29.3	0.444
0.8	2.4	0.117	195.6	0.225	1.8	0.117	11.9	0.225
0.9	0.4	0.010	63.8	0.071	0.2	0.010	1.8	0.071

Table 5. Impact of miminum support treshold. minimum confidence 0.6.

		not pruned		pruned	
Dataset, task	support	rules	accuracy	rules	accuracy
iris	7 (4.7%)	87	0.940	19	0.920
”	2 (1.3%)	168	0.947	21	0.913
”	1 (0.7%)	291	**0.967**	23	0.927
iris, sequence 1-2	7 (4.7%)	904	0.940	17	0.953
”	2 (1.3%)	1661	0.953	19	**0.960**
”	1 (0.7%)	2653	**0.960**	19	**0.960**
glass	10 (4.7%)	32	0.464	21	0.464
”	2 (0.9%)	2374	**0.622**	68	0.608
balance scale	10 (1.7%)	124	**0.891**	78	0.870
”	2 (0.4%)	558	0.841	216	0.714
balance scale, subset 1-2	10 (1.7%)	11947	0.758	153	**0.779**

rules are never applied. The minimum confidence threshold thus starts to have effect once it removes rules which cover objects uncovered by any other higher confidence rule.

A Similar effect can be observed for the minimum support threshold. An optimal support threshold of 1% is reported in [5], [9] gives 2%, while [10] suggests 2% or 3%. Our results indicate that the best results are obtained with support threshold set to 1 object.[8]

Pruning. Experimental results show that pruning is an effective tool for reducing the number of rules without significantly affecting classification accuracy and precision. Without pruning, confidence and support thresholds need to be carefully chosen in order to balance number of rules and performance (Table 2-5). Pruning ensures a manageable number of rules even for low threshold values. For example, the best performing setup on iris dataset achieves accuracy of 0.967 with 291 rules, no test object is left unclassified. Pruning reduces the number of rules to only 23 with a slight drop in accuracy due to an increase in the number of unclassified objects (Fig. 2).

Negation and Dynamic Binning. Experiments performed on the Glass and Iris datasets explore the effect of negation (ref. to Table 4 and Fig. 4). The results

[8] This setup is referred to in the literature as “no support” mining.

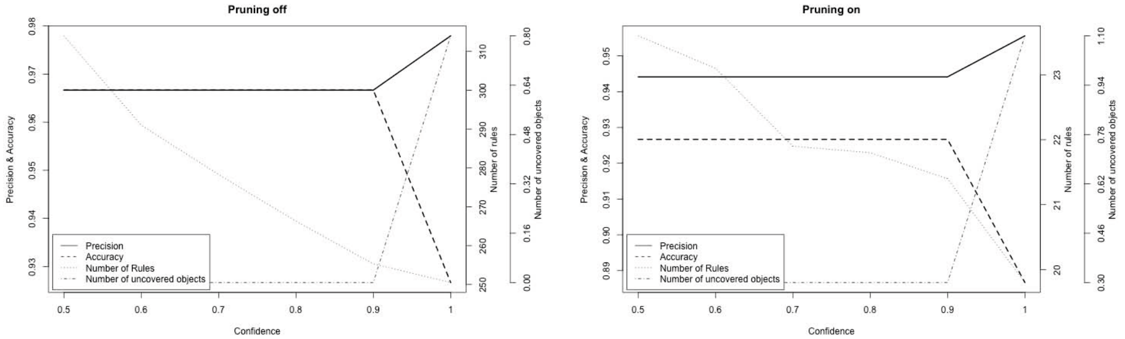


Fig. 2. Effect of pruning. Setting: Iris dataset, minimum support threshold 1.

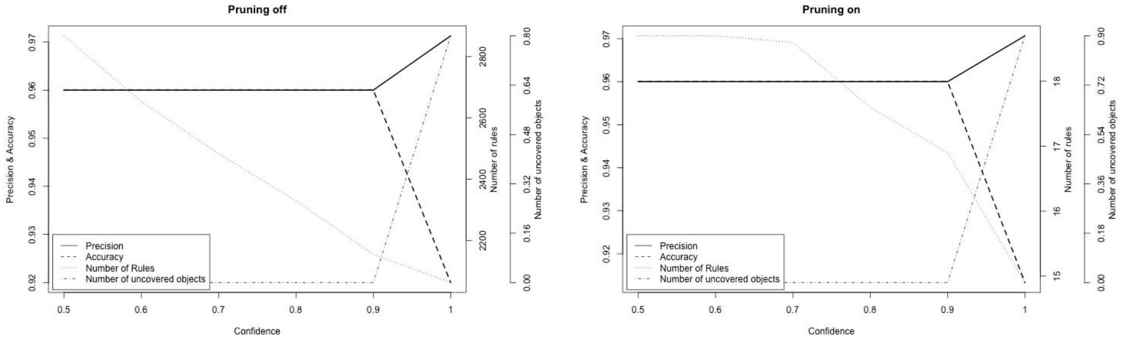


Fig. 3. Effect of dynamic binning on numerical attributes (sequence of length 2). Setting: Iris dataset, minimum support 1, dynamic binning on.

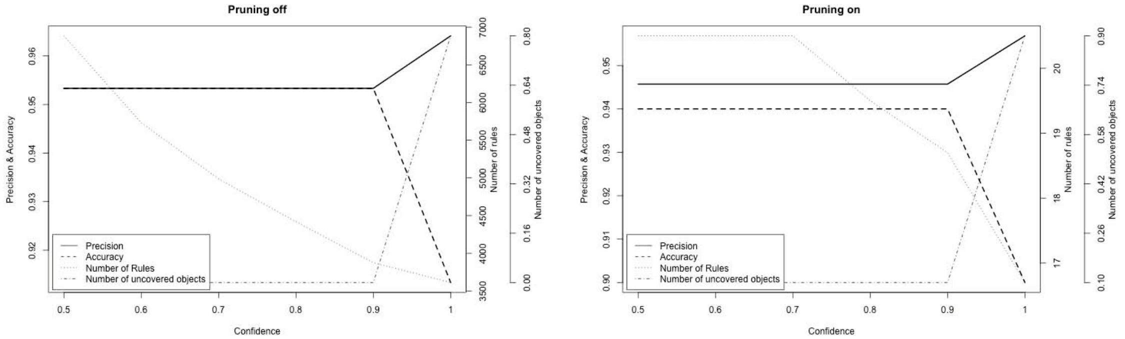


Fig. 4. Effect of including negative literals. Setting: Iris dataset, minimum support threshold 1.

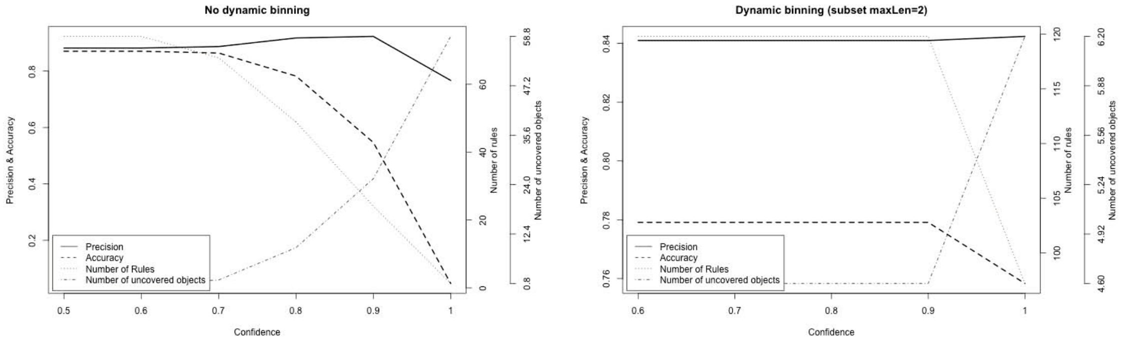


Fig. 5. Effect of dynamic binning on nominal attributes (subset of length 2). Setting: minimum support threshold 10, pruning on, Balance Scale dataset.

show that involving negation in rule learning phase significantly increases the computational demands of the rule learner used, while the results are generally unaffected in terms of accuracy, and inflated in terms of rule count.

Sequence binning was performed on the Iris dataset, which contains only numerical attributes. The results for a higher minimum support thresholds indicate that sequence binning slightly improves performance (Table 2) while simultaneously decreasing rule count. While overall the best accuracy of 0.967 is achieved without binning (Table 5), the result obtained with a pruned set of rules featuring dynamically created bins (0.960) is only slightly worse, but is composed of a much smaller set of rules (19 vs 291). For the Balance Scale dataset, which contains nominal attributes, subset binning was performed. This highly computationally intensive operation did not provide accuracy improvement (Table 3).

Comparison with Other Algorithms. To compare with earlier reported results for CBA, the first two brCBA columns report results from runs, which were generated with similar rule learning settings of 50% min. confidence and 1% min. support thresholds, no dynamic binning and no negation. There is, however, some difference in data preprocessing of numerical attributes – with brCBA we used equidistant binning (see Example 2).

The results depicted on Table 6 indicate that the in terms of accuracy, brCBA with no pruning gives the best performance by thin margin on the iris dataset, but lags behind significantly on the glass dataset. Comparing runs with pruning, the additional pruning steps in the "full" CBA provide better accuracy. And, according to the comparison with the rule count reported in [5], even smaller rule count.

It should be emphasized that the conclusions drawn above are only indicative due to a small number of datasets involved in the benchmark.

Table 6. Comparison with other systems – accuracy

	previous results [4,15]					brCBA	
dataset	c4.5	ripper	cmar	cpar	cba	not pr.	pruned
iris	0.953	0.940	0.940	0.94.7	0.947	**0.967**	0.927
glass	0.687	0.691	0.701	**0.744**	0.739	0.622	0.612

7 Conclusion

This paper investigated the possibility of learning classification business rules from data using association rule learning algorithms.

We introduced brCBA, a modification of the CBA algorithm, which omits the default rule classification. This enabled us to demonstrate the sensitivity of rule count and accuracy on the minimum confidence and support thresholds. Also, our modified implementation used a more expressive rule learning system, which allowed to study the effect of involving rules with disjunction and negations.

Our experimental evaluation on several UCI datasets lead to the following recommendations for business rule learning with ARC algorithms:

- The lowest confidence and support thresholds produce the best results. Since low threshold values have adverse effect on computational tractability, the setting of these thresholds is constrained by the available computational resources.
- Omission of important rules by pruning is a marginal, if any, issue, since pruned rule set maintains the accuracy of the original rule set on test data. Since pruning was at the same time found to significantly reduce the rule count, it is suitable for a business rule pruning setup.
- Involving higher expressiveness rules is not recommended given the substantial increase in computational demands and a negligible positive effect on accuracy and rule count (as opposed to default run with pruning).

It should be noted that the applicability of these recommendation is limited by the small number of the datasets involved in the experimental evaluation. Additionally, we have shown that the rule ranking algorithm used in CBA can be easily implemented as a rule conflict handling method in the Drools BRMS system, providing a complete workflow from data to actionable business rules.

As a future work, we plan to create an experimental web-based system that would allow to perform business rule learning with ARC algorithms. Also, we would like to further explore the topic of dynamic binning (disjunctions between values of one attribute), which provided promising results. It would be also interesting to perform additional experiments on a larger number of datasets.

Acknowledgment. The authors would like to thank the anonymous reviewers for their insightful comments. This work is supported by the European Union under grant no. FP7-ICT-2011-7 LinkedTV (Television Linked To The Web) and by the University of Economics in Prague by grants no. IGA 20/2013 and institutional support (IP 400040).

References

1. Agrawal, R., Imielinski, T., Swami, A.N.: Mining association rules between sets of items in large databases. In: SIGMOD, pp. 207–216. ACM Press (1993)
2. Antonie, M.-L., Zaïane, O.R.: Mining positive and negative association rules: An approach for confined rules. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 27–38. Springer, Heidelberg (2004)
3. Han, J., Pei, J., Yin, Y., Mao, R.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Min. Knowl. Discov. 8(1), 53–87 (2004)
4. Li, W., Han, J., Pei, J.: CMAR: accurate and efficient classification based on multiple class-association rules. In: ICDM 2001, pp. 369–376 (2001)
5. Liu, B., Hsu, W., Ma, Y.: Integrating classification and association rule mining. In: KDD 1998, pp. 80–86 (1998)
6. Ross Quinlan, J.: C4.5: Programs for Machine Learning. Morgan Kaufmann (1993)
7. Rauch, J., Šimůnek, M.: An alternative approach to mining association rules. Foundation of Data Mining and Knowl. Discovery 6, 211–231 (2005)

8. Thabtah, F.: Pruning techniques in associative classification: Survey and comparison. Journal of Digital Information Management 4(3) (2006)
9. Thabtah, F., Cowling, P., Peng, Y.: The impact of rule ranking on the quality of associative classifiers. In: Bramer, M., Coenen, F., Allen, T. (eds.) Research and Development in Intelligent Systems XXII, pp. 277–287. Springer, London (2006)
10. Thabtah, F., Cowling, P., Peng, Y.: Multiple labels associative classification. Knowledge and Information Systems 9(1), 109–129 (2006)
11. Thabtah, F.A.: A review of associative classification mining. Knowledge Eng. Review 22(1), 37–65 (2007)
12. Toivonen, H., Klemettinen, M., Ronkainen, P., Htnen, K., Mannila, H.: Pruning and grouping discovered association rules. In: ECML 1995 Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases, pp. 47–52 (1995)
13. Vanhoof, K., Depaire, B.: Structure of association rule classifiers: a review. In: 2010 International Conference on Intelligent Systems and Knowledge Engineering (ISKE), pp. 9–12 (November 2010)
14. Vojíř, S., Kliegr, T., Hazucha, A., Skrabal, R., Šimunek, M.: Transforming association rules to business rules: Easyminer meets drools. In: Fodor, P., Roman, D., Anicic, D., Wyner, A., Palmirani, M., Sottara, D., Lévy, F. (eds.) RuleML (2). CEUR Workshop Proceedings, vol. 1004. CEUR-WS.org (2013)
15. Yin, X., Han, J.: CPAR: Classification based on predictive association rules. In: Proceedings of the SIAM International Conference on Data Mining, pp. 369–376. SIAM, San Franciso (2003)

Appendix F: Web framework for interpretable machine learning based on rules and frequent itemsets

J4 Stanislav Vojíř, Václav Zeman, Jaroslav Kuchař, Tomáš Kliegr, EasyMiner.eu: Web framework for interpretable machine learning based on rules and frequent itemsets, Knowledge-Based Systems, 2018, , ISSN 0950-7051, `https://doi.org/10.1016/j.knosys.2018.03.006`.

Knowledge-Based Systems 150 (2018) 111–115

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

EasyMiner.eu: Web framework for interpretable machine learning based on rules and frequent itemsets

Stanislav Vojíř[a], Václav Zeman[a], Jaroslav Kuchař[a,b], Tomáš Kliegr[a,*]

[a] Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics, Prague, nám. W Churchilla 4, Prague, Czech Republic
[b] Web Intelligence Research Group, Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, Prague, 160 00, Czech Republic

ARTICLE INFO

Article history:
Received 26 October 2017
Revised 28 February 2018
Accepted 2 March 2018
Available online 9 March 2018

Keywords:
Association rules
Classification
Web service
Web application
Prediction API
Machine learning
Data mining

ABSTRACT

EasyMiner (http://www.easyminer.eu) is a web-based system for interpretable machine learning based on frequent itemsets. It currently offers association rule learning (apriori, FP-Growth) and classification (CBA). EasyMiner offers a visual interface designed for interactivity, allowing the user to define a constraining pattern for the mining task. The CBA algorithm can also be used for pruning of the rule set, thus addressing the common problem of “too many rules” on the output, and the implementation supports automatic tuning of confidence and support thresholds. The development version additionally supports anomaly detection (FPI and its variations) and linked data mining (AMIE+). EasyMiner is dockerized, some of its components are available as open source R packages.

1. Introduction

Rules are one of the most accessible forms of knowledge that can be derived from data, and can thus serve as a basis for a machine learning framework focused on generation of interpretable models. In order to ensure scalability, the presented system relies on association rule learning, which uses efficient algorithms for frequent itemset mining proven to work on large datasets [1]. While association rules were originally devised for exploratory data mining, they can also be turned to a classifier and also serve as a basis for interpretable anomaly detection [2]. The EasyMiner framework contains a carefully curated selection of algorithms based on association rules and their “building blocks” – the frequent itemsets. These cover some of the most common machine learning problems while fostering interpretability by adhering to one type of symbolic knowledge representation.

Association rule learning can be informally described as a task of finding all rules in the input dataset of the form: *antecedent* ⇒ *consequent*, which meet predefined statistical measures of interest. When the input for association rule learning is a transaction database as originally expected by the `apriori` algorithm, the first approach for mining association rules [3], the discovered association rules are composed of *items*. Example of such rule is: *onion, potato* ⇒ *meat*. In EasyMiner, the input for association rule learning is a flat file containing *multinominal attributes*, as in the standard classification task. This corresponds to output association rules such as $district{=}Prague \wedge salary{=}Low \Rightarrow rating{=}C$. Each rule is associated with interest measures, such as *support*, defined as the number of data rows (instances) matching the entire rule, and *confidence* that expresses how many percent of instances matching the antecedent also match the consequent.

Algorithms for classification that are based on association rules take the list of rules output by association rule learning on the input and process it into a rule-based classifier. Classification based on Associations (CBA) algorithm proposed by Liu et al. [4] is considered as the reference algorithm for this group of classification algorithms. The main steps in CBA are removal or redundant rules and inclusion of a default rule, which ensures that every test instance is covered. While CBA, proposed in 1998, is a relatively old algorithm, we included it into EasyMiner. The output of CBA is more user friendly than of its successors, while the difference between the CBA’s accuracy and the accuracy of the state-of-the-art association rule classification algorithms is very small [5]. CBA also helps to address one of the main problems with association rule learning (as an exploratory data mining task), which is the high

[*] Corresponding author.
E-mail addresses: Stanislav.vojir@vse.cz (S. Vojíř), Vaclav.Zeman@vse.cz (V. Zeman), Jaroslav.Kuchar@vse.cz, Jaroslav.Kuchar@fit.cvut.cz (J. Kuchař), tomas.kliegr@vse.cz (T. Kliegr).

https://doi.org/10.1016/j.knosys.2018.03.006
0950-7051/© 2018 Elsevier B.V. All rights reserved.

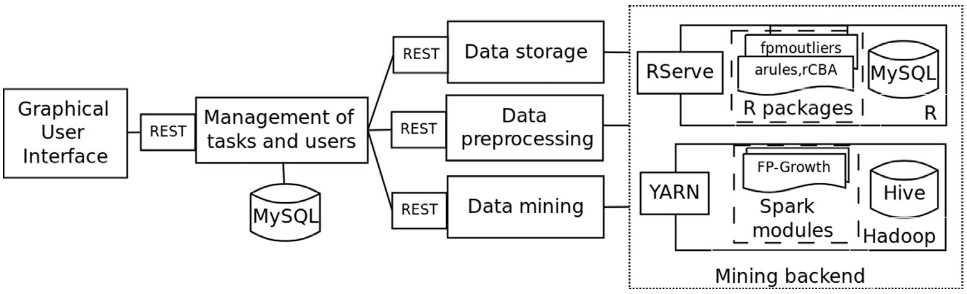


Fig. 1. Architecture of the EasyMiner system.

number of rules that can be generated. Since CBA only removes rules from the original list, it can be used for pruning the set of association rules.

As two additional types of task, the development version of EasyMiner integrates anomaly detection and extraction of association rules from linked data. Anomaly detection is based on the frequent itemset-based outlier detection approach [6,7], which assumes that if an instance is covered by multiple frequent itemsets, it means that this data instance is unlikely to be an anomaly. The linked data support is based on implementation of the AMIE+ algorithm for rule mining in ontological knowledge bases [8].

2. Problems and background

The presented system offers an open source web-based framework for machine learning. Its main functionality covered in this article is association rule learning and building of classifiers composed of association rules. The difference between EasyMiner and main-stream open source machine learning toolboxes, such as Scikit-learn or R (http://scikit.ml/, https://www.r-project.org/), or specialized toolboxes such as spmf (http://www.philippe-fournier-viger.com/spmf/) that also contain some of the algorithms available in EasyMiner, is that EasyMiner focuses on ease of use for the end user, who is not required to have any programming skills, providing out-of-the-box graphical user interface and Predictive REST API (PAPI).

There are several extensions that add REST APIs or visual interfaces to the open source toolboxes listed above. An example of such a system is Shiny (https://shiny.rstudio.com/) for R. The difference between EasyMiner and Shiny is that Shiny provides a generic, one-for-all approach. In contrast, EasyMiner is an integrated bundle, which was crafted to support machine learning workflows based on association rules and frequent itemsets. This focus on a specific type of model allowed us to make architectural choices that improve user experience as well as performance of the system as a whole.

3. Software framework

EasyMiner is composed of several microservices, which communicate via REST APIs (ref. to Fig. 1). The application has two logical layers. *Frontend* provides user interface, management of users and tasks and integration of backend services. *Backend* handles the data processing. The data processing itself is composed of three independent components: data storage, data preprocessing and data mining. Fig. 1 gives a more detailed view of the core data mining layer. EasyMiner historically supports three different mining backends. This article covers the latest stable version of EasyMiner, which uses for data mining operations the popular R framework (http://r-project.org/). The first mining backend in EasyMiner, implemented on top of the LISp-Miner system [9], was to our knowledge the first web-based application for exploratory data mining. There is also a proprietary backend on top of Apache Spark/Hadoop, a demo of which is accessible from the EasyMiner website (http://www.easyminer.eu).

3.1. Comparison with other systems

In this section, we present a brief comparison with related open source or academic software.

Web-based systems. We are not aware of any open-source web-based system for association rule learning and building of classification models out of association rules. Closest to our approach is perhaps the MIME framework [10], a desktop application apparently no longer actively developed, allowing interactive frequent pattern mining.

R framework. Our *rCBA* package used in EasyMiner/R was the first open implementation for R. Currently, there are two other CBA implementations: the *arc* package[1] and the recently introduced *arulesCBA* package.[2] The *fpmoutliers* package[3] in EasyMiner is, to our knowledge, the first open implementation of frequent pattern-based anomaly detection available in R.

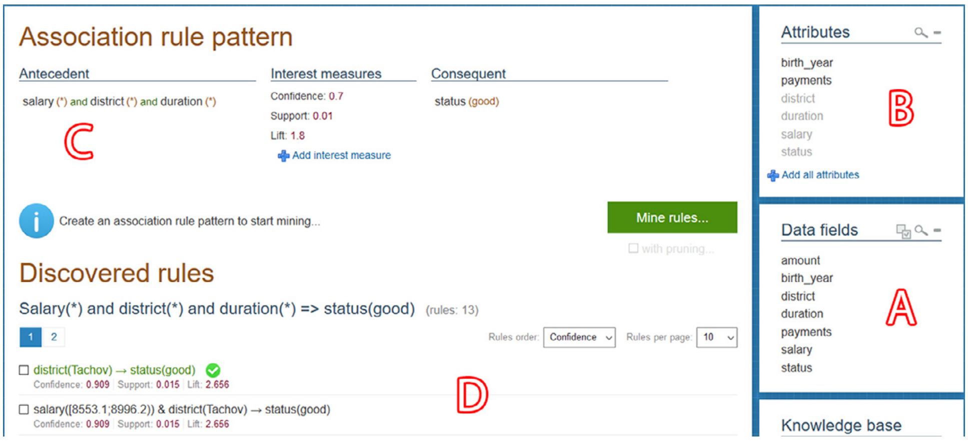
Association rule learning is a standard machine learning task which is supported in most machine learning toolboxes, both open source and commercial. We focus our comparison on the *arules* package [11], which is the most widely used association rule learning package in the R framework. EasyMiner advantages compared to direct use of *arules*:

- *Attribute-value pairs allowed in antecedent and consequent are set visually* (Fig. 2C) on per attribute level or attribute-value pair level. In order to constraint the consequent to a specific attribute directly in *arules*, one has to explicitly include all the attribute value pairs in the `apriori` command. This can be tedious for attributes with many values. In EasyMiner, this common task translates to dropping an attribute to the consequent part of the rule. EasyMiner also allows the user to easily fix the attribute to a specific attribute value.
- *No need to explicitly convert flat dataset to the flat file transaction format.* When analyzing multi-column data in *arules* one has to transform attribute values to items. For example, value *Prague* of attribute *district* needs to be transformed to `district=Prague`. In EasyMiner, this process is completely transparent.
- *Displays intermediate results as mining progresses on next rule length level.* Internally, this is performed by initiating the

[1] https://CRAN.R-project.org/package=arc.

[2] https://CRAN.R-project.org/package=arulesCBA.

[3] https://CRAN.R-project.org/package=fpmoutliers.



Association rule pattern

Antecedent

salary (*) and district (*) and duration (*)

Interest measures

Confidence: 0.7

Support: 0.01

Lift: 1.8

Consequent

status (good)

Create an association rule pattern to start mining...

□ with pruning...

Discovered rules

Salary(*) and district(*) and duration(*) => status(good) (rules: 13)

Rules order: Confidence

Rules per page: 10

□ district(Tachov) → status(good) ✓
Confidence: 0.909 | Support: 0.015 | Lift: 2.656

□ salary([8553.1;8996.2)) & district(Tachov) → status(good)
Confidence: 0.909 | Support: 0.015 | Lift: 2.656

Attributes

birth_year
payments
district
duration
salary
status

Data fields

amount
birth_year
district
duration
payments
salary
status

Knowledge base

Fig. 2. User interface of EasyMiner/R.

arules' `apriori` method separately for each rule length level and returning the results to the user interface when mining finishes. This ensures that the user gets first results very quickly.

- *Built-in support for rule pruning*. With CBA rule pruning, the number of rules in the rule set is often reduced by one or more orders of magnitude, while it is ensured that every instance is covered by at least one rule. In [12] we have demonstrated on several datasets that pruning does not negatively impact the predictive power of the rule set.
- *The user interface can be operated by mouse only*. Knowledge of the R language is not required.

4. Implementation and empirical results

4.1. Software architecture

This paper describes EasyMiner/R, which uses the R framework for performing machine learning tasks. The association rule learning step in CBA is performed by implementation of the apriori algorithm in C introduced in [13], which is wrapped into the R's *arules* package [11]. The pruning has been partly implemented in Java and wrapped as a standalone R package.[4] The use of R facilitates further extensions of the system, for example with additional preprocessing algorithms. The choice of the most suitable frequent pattern mining algorithm depends on the characteristics of the dataset [14]. By decoupling rule mining from pruning, the system can use several frequent-pattern mining backends, allowing the user to choose a suitable backend for given data, such as FP-Growth. The R computation backend is wrapped into a web REST service. This wrapper transforms declarative task definitions to R scripts and forwards them to the R environment relying on the `Rserve` server. It accepts mining tasks in a modification of the PMML format supporting standard association rules mined by apriori-like algorithms as well as the more expressive GUHA rules mined by LISp-Miner [15], which was the first backend supported by EasyMiner [9]. Note that a new component available in the development version of the R backend is the *fpmoutliers* R package, which implements several algorithms for frequent-pattern based outlier detection.

The frontend layer is implemented in PHP using Nette Framework (https://nette.org) and JavaScript. It also exposes a high-level web service interface (a Prediction API). The backend layer is implemented in the Scala language. To handle concurrent connections it relies on the Akka (https://akka.io) framework with Spray (http://spray.io).

EasyMiner supports two extension paradigms: RESTful web services and Akka actors. Integrating a new web service requires writing a new driver by extending one of the existing interfaces, depending on the purpose of algorithm added. If the newly integrated algorithm is in Scala or Java language, it is also possible and more resource effective to add it using the Akka framework. A microservice can also be created for the actor usage. Technical details for extending EasyMiner are present in the developer documentation.[5] This includes a use case describing the integration of our implementation of AMIE+ algorithm.

4.2. Benchmarks

The purpose of this evaluation is to i) complement earlier published benchmarks on CBA in Alcala-Fdez et al. [16], which focused on recently proposed association rule and fuzzy classifiers with comparison involving tried symbolic classifiers supported in a number of open source and commercial systems, ii) demonstrate that our system can cope with a wide variety of datasets, iii) evaluate effect of automatic parameter tuning in CBA as opposed to default setting, iv) by involving standard benchmark datasets allow comparison of our CBA results with other CBA implementations.

We evaluated EasyMiner on 36 UCI[6] datasets (13 with only nominal attributes and 27 including also numeric attributes). Each dataset is divided to train and test sets using a 10-fold stratified cross-validation.[7] Our benchmark also involves a comparison with several related symbolic algorithms as implemented in common machine learning software packages: DecisionTree classifier from Scikit-learn Machine Learning in Python (PDT) and Weka implementations (http://www.cs.waikato.ac.nz/ml/weka/) of C4.5 (J48), PART, RIPPER (RIP) and our CBA implementation.

For CBA, we involved two setups. First, CBA$_d$ uses default values for metaparameters, which were proposed by [4] and used in subsequent evaluations (e.g. Alcala-Fdez et al. [16]). The second setup is our CBA automatic tuning approach (CBA$_a$), which automatically finds confidence and support thresholds. Detailed benchmark setup and results can be found at http://www.easyminer.eu/benchmarks,

[4] https://CRAN.R-project.org/package=rCBA.

[5] http://www.easyminer.eu/developers.

[6] Available from https://archive.ics.uci.edu/ml/datasets.html. This collection is commonly used for evaluation of machine learning algorithms.

[7] Folds for crossvalidation were generated with Scikit-learn (http://scikit-learn.org).

Table 1
Counts of wins, losses and ties for CBA_d.

Dataset	CBA_d won	Tie	Loss	Omitted	p
J48 auto	15	3	17	2	0.56
PART auto	13	3	17	1	0.37
RIPPER auto	15	5	15	1	0.60
Python decision tree auto	24	2	9	1	0.0

which also contains link to the evaluation framework used. The results are summarized by counts of wins, losses an ties [17] reported for CBA_d in Table 1.

The benchmarks show that our CBA is competitive to other public implementations of symbolic classifiers, providing the best overall result for 10 of the 36 datasets (CBA_d). Our CBA implementation outperforms the Scikit-learn DecisionTree (PDT) and is on part with the other involved symbolic learners. The worse result for PDT can be possibly attributed to the fact that this learner does not directly support nominal attributes, which were converted to dummy variables during our preprocessing. Contrary to our earlier preliminary results [18], CBA generated larger models than other algorithms. The comparison between CBA with default parameters and our automatically tuned version shows that neither performs consistently better than the other. Overall, our results confirm that CBA provides stable results with default meta-parameter values.

Regarding the size of the produced models, the number of rules generated by CBA is comparable to the other symbolic rule learners, but on nearly all datasets CBA produces larger rule lists. Overall, the current CBA implementation in EasyMiner may not therefore be the best option for applications requiring the lowest possible rule list size.

5. Illustrative examples

In this section, we will cover the entire workflow of analyzing dataset with EasyMiner. A video file demonstrating this process on a particular dataset is contained in the supplementary material.

First, the user has to log in using a local account or a social network account. After authentication, the user uploads the dataset in CSV or zipped CSV, there is also the option to reuse an already uploaded data file. Once the data are uploaded, the user selects which data fields will be used by dragging a field from the *Data fields palette* to the *Attributes palette* (Fig. 2A,B) and selecting a preprocessing type. This creates an *attribute* that can be used in the *Rule pattern*, out of a field in the input CSV file.

To define the task, the user drags an attribute from the Attributes palette (Fig. 2B) and drops it into the antecedent or consequent part of the Rule pattern (Fig. 2C). This defines the pattern (template) for discovered association rules and constraints the search space. At this point, the user can also constraint the set of distinct values of the attribute that will be considered in the task to only a specific value. Finally, the user executes the task. This sends the task definition to the mining backend. For tasks with a single attribute in the consequent, the user can opt to perform pruning with the CBA algorithm.

A useful feature for computationally intensive tasks is the fact that rules are returned gradually, as the mining algorithm progresses through the search space. The user can decide any time to cancel the mining and work with the results collected so far. The discovered rules (Fig. 2D) can be sorted by values of interest measures. Selected rules can be stored in the *Rule Clipboard* (not shown), the contents of which persists across multiple tasks on the same dataset, or in the *Knowledge Base*, which persists across multiple datasets.

6. Conclusions

EasyMiner/R available at http://www.easyminer.eu is an open source framework for interpretable machine learning. For association rule learning and classification, it offers an interactive web-based interface. The user visually constructs a “query” in the browser by defining a rule template. Since the mining proceeds incrementally from shorter to longer rules, the user is served the shortest, and typically most satisfying rules first before the mining has finished. If there are too few or too many results or the rules contain different attributes or values than desired, the user can easily change the rule template or the minimum thresholds, staying within the same screen.

The EasyMiner workflow also opens up new opportunities for utilizing the discovered rules. One example is a pilot EasyMiner extension for identifying rules novel with respect to existing domain knowledge. In [12] we have shown that the CBA algorithm meets the requirements imposed on learning of business rules from data. In [9] we review applied research in this direction, including a proof-of-concept integration between EasyMiner and Drools (http://drools.org). EasyMiner/R is designed to be used as a complete integrated system, however, its individual components can also be used independently within the R ecosystem. EasyMiner has been used to complete assignments by estimated 1000 students at the University of Economics, Prague over the course of several years. The system is also used to explain association rule mining in a recent text book on Business Intelligence [19]. The code base of the EasyMiner system exceeds 100.000 lines. The system was continuously improved since its first release in 2012 [9]. The current version consists of multiple web services, evaluation framework, user interfaces, algorithms, docker installer (https://www.docker.com) and continuous integration (https://travis-ci.org).

Limitations and future work. Compared to the previous EasyMiner version based on the LISp-Miner association rule learner, the R-based version of EasyMiner does not allow to use expressive constructs such as disjunctions and negations when defining the rule template as well as several other interest measures. Limiting the expressiveness allows for substantially faster mining times [15]. Users needing these extra features can use LISp-Miner (http://lispminer.vse.cz).

The apriori/frequent pattern mining (fpm) approach adopted in EasyMiner does not take into account the relative importance and non-binary occurrence of items. If these limitations are salient, the user should turn to algorithms implementing *high-utility itemset mining* [20]. One of the most comprehensive selection of such algorithms is provided by the *spmf* library.

EasyMiner provides support for streaming upload, but the actual mining is performed on a static copy of the data. For analysis of continuous data where the importance of instances fluctuates in time, algorithms specifically proposed for streams should be utilized. MPM [21] is a recently developed algorithm that applies a state-of-the-art high-utility itemset mining approach to the stream environment. An interesting area of future work would be integration of an MPM-like algorithm into EasyMiner.

Required metadata

(Tables 2 and 3)

Table 2
Software metadata (optional).

Nr.	(executable) Software metadata description	Please fill in this column
S1	Current software version	2.4
S2	Permanent link to executables of this version	https://hub.docker.com/r/kizi/easyminer-frontend/ (tag v2.4)
S3	Legal software license	Apache license V2
S4	Computing platform/Operating system	Linux
S5	Installation requirements & dependencies	Docker
S6	Link to user manual	http://www.easyminer.eu/tutorial
S7	Support email for questions	vaclav.zeman@vse.cz

Table 3
Code metadata (mandatory).

Nr.	Code metadata description	Please fill in this column
C1	Current code version	2.4
C2	Permanent link to code/repository used of this code version	https://github.com/KIZI/EasyMiner/releases/tag/v2.4
C3	Legal code license	Apache license V2
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	Scala, Java, R, PHP, JavaScript
C6	Compilation requirements, operating environments & dependencies	Scala 2.11, Java 8, R>=3.3.0
C7	Link to developer documentation/manual	NA
C8	Support email for questions	stanislav.vojir@vse.cz (frontend), vaclav.zeman@vse.cz (backend)

Acknowledgments

We would like to thank Jan Rauch and Milan Milan Šimůnek for helpful discussions. The research and development of EasyMiner was supported by the European Union via the LinkedTV project (no. FP7-287911) and the OpenBudgets.eu project (No. H2020-645833) and by the University of Economics, Prague (grants IGA 15/2010, 26/2011, 21/2013 and 29/2016 and institutional support for long term research of SV and TK). The development of the CBA component was supported by CESNET grant no. 540/2014.

The contributions of the authors are as follows. Referring to Fig. 1, SV implemented the graphical user interface and the outer REST interface including management of tasks/users. VZ implemented the REST services and the service exposing RServe. JK implemented the rCBA and fpmoutliers R packages. TK conceived and managed the project and wrote the paper. The authors would also like to acknowledge the contributions of numerous students who helped to advance the system with their theses.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.knosys.2018.03.006.

References

[1] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning, Springer Series in Statistics, Springer, New York, NY, USA, 2001.
[2] M. Kopp, M. Holena, Evaluation of association rules extracted during anomaly explanation, in: ITAT, 2015, pp. 143–149.
[3] J. Fürnkranz, T. Kliegr, A brief overview of rule learning, in: International Symposium on Rules and Rule Markup Languages for the Semantic Web. RuleML 2015, Springer, 2015, pp. 54–69.
[4] B. Liu, W. Hsu, Y. Ma, Integrating classification and association rule mining, in: KDD'98: Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, 1998, pp. 80–86.
[5] T. Kliegr, Quantitative CBA: small and comprehensible association rule classification models, arXiv:1711.10166 (2017).
[6] Z. He, X. Xu, Z.J. Huang, S. Deng, Fp-outlier: frequent pattern based outlier detection, Comput. Sci. Inf. Syst. 2 (1) (2005) 103–118.
[7] J. Kuchař, V. Svátek, Spotlighting anomalies using frequent patterns, in: Proceedings of the KDD 2017 Workshop on Anomaly Detection in Finance. Halifax: PMLR, 2018.
[8] L. Galárraga, C. Teflioudi, K. Hose, F.M. Suchanek, Fast rule mining in ontological knowledge bases with AMIE+, VLDB J. 24 (6) (2015) 707–730.
[9] T. Kliegr, J. Kuchař, S. Vojíř, V. Zeman, Easyminer-short history of research and current development, in: ITAT 2017, 2017, pp. 235–239.
[10] B. Goethals, S. Moens, J. Vreeken, MIME: a framework for interactive visual pattern mining, in: Proceedings of the 17th Int. Conference on Knowledge Discovery and Data Mining, ACM, 2011, pp. 757–760.
[11] M. Hahsler, S. Chelluboina, K. Hornik, C. Buchta, The arules R-package ecosystem: analyzing interesting patterns from large transaction data sets., J. Mach. Learn. Res. 12 (2011) 2021–2025.
[12] T. Kliegr, J. Kuchař, D. Sottara, S. Vojíř, Learning Business Rules with Association Rule Classifiers, Springer, Rules on the Web. From Theory to Applications. RuleML 2014. Lecture Notes in Computer Science, vol 8620 Springer, Cham, pp. 236–250.
[13] C. Borgelt, Efficient implementations of Apriori and Eclat, in: FIMI03: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations, in: CEUR Workshop Proc., 2003.
[14] B. GOETHALS, M.J. ZAKI, FIMI03: workshop on frequent itemset mining implementations, in: Third IEEE ICDM Workshop On Frequent Itemset Mining Implementations, 2003, pp. 1–13.
[15] J. Rauch, M. Šimůnek, Apriori and GUHA – comparing two approaches to data mining with association rules, Intell. Data Anal. 21 (4) (2017) 981–1013.
[16] J. Alcala-Fdez, R. Alcala, F. Herrera, A fuzzy association rule-based classification model for high-dimensional problems with genetic rule selection and lateral tuning, IEEE Trans. Fuz. Syst. 19 (5) (2011) 857–872.
[17] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[18] T. Kliegr, J. Kuchař, Benchmark of Rule-Based Classifiers in the News Recommendation Task, Springer, pp. 130–141.
[19] W. Grossmann, S. Rinderle-Ma, Fundamentals of Business Intelligence, Springer Berlin Heidelberg, Berlin, Heidelberg, 2015.
[20] U. Yun, H. Ryang, G. Lee, H. Fujita, An efficient algorithm for mining high utility patterns from incremental databases with one database scan, Knowl. Based Syst. 124 (Supplement C) (2017) 188–206, doi:10.1016/j.knosys.2017.03.016.
[21] U. Yun, D. Kim, E. Yoon, H. Fujita, Damped window based high average utility pattern mining over data streams, Knowl. Based Syst. 144 (2018) 188–205, doi:10.1016/j.knosys.2017.12.029.

Appendix G: InBeat: JavaScript recommender system supporting sensor input and linked data

J4 Jaroslav Kuchař, Tomáš Kliegr, InBeat: JavaScript recommender system supporting sensor input and linked data, Knowledge-Based Systems, Volume 135, 2017, Pages 40-43, ISSN 0950-7051, https://doi.org/10.1016/j.knosys.2017.07.026. Keywords: Recommender system; Semantic web; Association rules; Sensors

Knowledge-Based Systems 135 (2017) 40–43

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Original Software Publication

InBeat: JavaScript recommender system supporting sensor input and linked data

Jaroslav Kuchař[a,c,*], Tomáš Kliegr[b,c]

[a] Web Intelligence Research Group, Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, 160 00, Prague, Czech Republic
[b] Multimedia and Vision Research Group, Queen Mary University of London, 327 Mile End Road, London E1 4NS, United Kingdom
[c] Department of Information and Knowledge Engineering, Faculty of Informatics and Statistics, University of Economics Prague, nám. W Churchilla 4, Czech Republic

ARTICLE INFO

Article history:
Received 8 December 2016
Revised 18 July 2017
Accepted 19 July 2017
Available online 20 July 2017

Keywords:
Recommender system
Semantic web
Association rules
Sensors

ABSTRACT

Interest Beat (inbeat.eu) is an open source recommender framework that fulfills some of the demands raised by emerging applications that infer ratings from sensor input or use linked open data cloud for feature expansion. As a recommender algorithm, InBeat uses association rules, which allow to explain why a specific recommendation was made. Due to modular architecture, other algorithms can be easily plugged in. InBeat has a pure JavaScript version, which allows to confine processing to a client-side device. There is a performance optimized server-side bundle, which succesfully participated in two recent recommender competitions involving large volumes of streaming data. InBeat works on a number of platforms and is also available for Docker.

1. Introduction

InBeat is an open recommender system framework that supports sensor input and linked data while addressing the privacy and scalability requirements. Sensor, as understood in Inbeat, is a source of signal indicating the level of user's preference for a given item. Unlike traditional, discrete, recommender input, such as user putting an item to shopping cart, sensor input can provide a near continuous stream of data (such as user's face expression). Another characteristic of sensor input is that the individual data point has limited information value, a meaningful measure of user engagement is obtained by aggregating the input over a specific time period.

While InBeat has all main capabilities of a recommender system, its main strength is the ability to combine sensor input into a single preference (*interest*) value within a given scope (for example video shot). If item features are linked data identifiers, InBeat can propagate weights from the original features into the ontological classes, generating new features.

Since InBeat is almost purely implemented in JavaScript, it can run on client side allowing for near complete privacy preservation: user data do not leave the user's device. Its reference recommender algorithm is an association rule learner. Since rules are easily interpretable, this allows to effectively explain the recommendation to the user. User can edit the model by simply deleting a specific rule, conceivably also by altering it. The default rule-based classifier can be replaced by any machine learning or collaborative filtering library if interpretability is not a required. However, InBeat by itself does not contain a collaborative filtering algorithm, as its design imperative was on addressing functionality requirements unmet by existing open systems. Due to its modular architecture, algorithms from other frameworks can be plugged-in.

2. Problems and background

There are multiple open source recommender systems and frameworks, a good recent overview of their functionality with respect to the standard collaborative filtering criteria is provided in Lee et al. [1]. Given the limited space, we provide here only partial comparison, focusing on the new functionality requirements outlined in the introduction. The following main stream open source systems are included: MyMedia Lite (MML), LensKit (LK), Recommenderlab (RL), easyrec (er), PredictionIO (PIO), racoon (ra), HapiGER (HG), GraphLab (GL), Xelopes (Xe), Apache Mahout (Ma),

[*] Corresponding author at : Web Intelligence Research Group, Faculty of Information Technology, Czech Technical University in Prague, Thákurova 9, 160 00, Prague, Czech Republic.

E-mail addresses: jaroslav.kuchar@fit.cvut.cz (J. Kuchař), t.kliegr@qmul.ac.uk (T. Kliegr).

http://dx.doi.org/10.1016/j.knosys.2017.07.026

Table 1
Comparison between InBeat and mainstream open recommender systems.

	InBeat	MML	LK	Rl	er	PIO	ra	HG	GL	Xe	Ma	Rs	Rdb
Sensor	+++	o	?	?	++	+	++	+++	?	?	?	++	o
Linked Data	+	o	o	o	o	o	o	o	o	o	o	o	o
Explains	++	o	o	o	o	?	o	o	o	o	o	o	o
Privacy	+++	o	o/+	o	o	o/+	o	o	o	?	o	o/+	o
Scalability	+++	o/+	o/+	o	+	+++	++	++	+	o/+	+++	o	o/+

RankSys (Rs), RecDB (Rdb).[1] We did not include experimental approaches described in the literature that do not meet common requirements on open software (free license, sufficient documentation).

It should be noted that the data presented in Table 1 are only indicative, a thorough comparison would require a dedicated survey [2]. However, even considering for a margin of error in individual criteria, the overview suggests that there is currently no other open system available with similar feature set.

Sensor support (Streaming implicit feedback): **o** accepts rank only, + view time only, ++ multiple types of implicit preference, +++ supports aggregation mechanism for combining arbitrary set of preference clues with different weights.

Linked data support: **o** no support, + taxonomy support, ++ dynamic retrieval of features from LOD cloud [see 3].

Explains recommendation: **o** no ability, + ability to explain recommendation, ++ user can edit the model

Privacy preserving: **o** uses collaborative filtering with no privacy preservation option, +/++ has some degree of privacy preservation such as k-identity, ++ user data sent to server, but not used for other recommendations (that is no collaborative filtering), +++ full client-side option (no user data sent to server).

Scalability: **o** cannot be used on-line, + web service wrapper available, ++ natively implemented as a web service, +++ scalable implementation (employs technologies such as load balancing, parallel execution, map reduce)

3. Software framework

3.1. Software architecture

InBeat is composed of three main modules: GAIN (General Analytics INterceptor), Preference Learning and Recommender System. All of these modules are designed to be independent on each other in order to support flexible and expose separate RESTfull interfaces.

GAIN module captures descriptions of the content interacted with as well as clues indicating or expressing user interest in the individual content items and provides aggregated outputs on multiple granularities.

Preference Learning module builds a recommendation model for each user. The current version implements only association rule learning (either using own JavaScript implementation or relevant R packages[2]), but this can be substituted by any standard learner.

Recommender System module executes the model created in the Preference learning module providing a list of candidate recommendations associated with a confidence value. The current implementation uses *one rule classification* following the CBA algorithm [4], allowing for easy explanation.

The framework contains a simple graphical administration interface, which for all modules allows to (i) set configuration parameters, (ii) test their functionality, (iii) provide basic visualizations.

3.2. Software functionalities

InBeat retrieves interactions in JSON structure containing following essential input information: {*user, interaction, object*} , where *user* is the one who interacts, *object* is an entity which can be interacted with and *interaction* is an interest clue for the object. Each object can be described by multiple *attributes*. If a corresponding taxonomy or taxonomies (OWL format [5]) are provided, GAIN propagates weights up the taxonomy, boosting classes that the objects interacted with have in common. GAIN supports nested objects, for example, one chapter of a video can contain multiple shots, with which the user can interact separately. As one of the outputs, GAIN can aggregate interactions on multiple objects, for example on shots within a video chapter, into one fixed-length vector.

Similarly, multiple interactions, interpreted as interest clues, can be captured for each object. These are aggregated into a single interest value. By default, this is performed by a predefined set of rules that increase or decrease interest value based on the type of the interaction, for example, user looking on the screen increases interest, muting sound decreases interest. This process can be replaced by a supervised classifier.

4. Implementation and empirical results

InBeat is implemented in JavaScript, including its default association rule learner *apriori-js* and a rule engine for recommendation. This allows the system to be embedded in a JavaScript enabled end-user device, such as a set-top-box or an Internet browser.

The default InBeat setup targets server deployment within *Node.js* (http://nodejs.org/), a platform for building fast and scalable applications. The main entry point is load balancer *nginx*(http://nginx.org/), which distributes the workloads across multiple instances of API applications. As storage we selected *MongoDB* (http://www.mongodb.org/). Its key advantage is a schema-less design, which allows to use a custom set of attributes without the need to update the schema. In order to provide a higher performance alternative to *apriori-js*, InBeat provides a wrapper for the R arules package [6] and its own lightweight rule engine (a scorer).

InBeat was largely developed within the scope of Television Linked to the Web (LinkedTV) project [7]. The *GAIN* module was evaluated using sensor data from Microsoft Kinect as input [8]. In Kliegr and Kuchař [9] we show that a linked data layer can be added to existing recommender problems, if the available text (for example video subtitles) is run through a semantic entity classifier, such as DBpedia Spotlight [10]. The association rule mining algorithms were benchmarked with other common learners on a recommender dataset in Kliegr and Kuchař [11].

In terms of practical use cases and scalability, InBeat was evaluated in two on-line news challenges on the Plista platform, which provides recommendations for high-traffic news portals. In addi-

[1] (MML, mymedialite.net), (LK, lenskit.org), (RL, cran.r-project.org/package=recommenderlab), (er, easyrec.org), (PIO, prediction.io), (ra, github.com/guymorita/recommendationRaccoon), (HG, hapiger.com), (GL, dato.com), (Xe, www.prudsys.de), (Ma, mahout.apache.org), (Rs, github.com/RankSys/RankSys , (Rdb, github.com/DataSystemsLab/recdb-postgresql)).

[2] https://CRAN.R-project.org/package=rCBA.

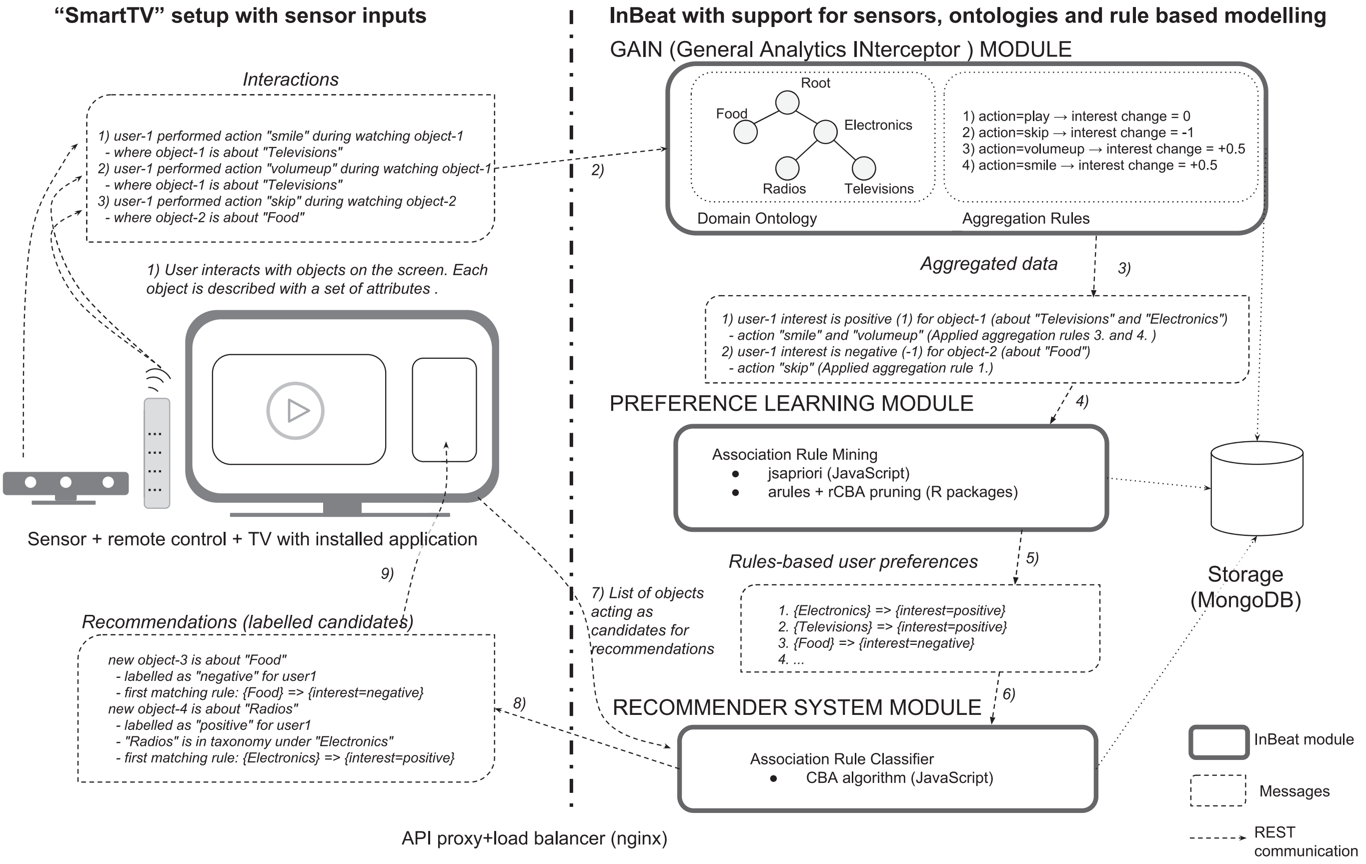

"SmartTV" setup with sensor inputs
InBeat with support for sensors, ontologies and rule based modelling
GAIN (General Analytics INterceptor) MODULE
Interactions
1) user-1 performed action "smile" during watching object-1
- where object-1 is about "Televisions"
2) user-1 performed action "volumeup" during watching object-1
- where object-1 is about "Televisions"
3) user-1 performed action "skip" during watching object-2
- where object-2 is about "Food"
2)
Root
Food
Electronics
Radios
Televisions
Domain Ontology
1) action=play → interest change = 0
2) action=skip → interest change = -1
3) action=volumeup → interest change = +0.5
4) action=smile → interest change = +0.5
Aggregation Rules
Aggregated data
3)
1) User interacts with objects on the screen. Each object is described with a set of attributes .
1) user-1 interest is positive (1) for object-1 (about "Televisions" and "Electronics")
- action "smile" and "volumeup" (Applied aggregation rules 3. and 4.)
2) user-1 interest is negative (-1) for object-2 (about "Food")
- action "skip" (Applied aggregation rule 1.)
PREFERENCE LEARNING MODULE
4)
Association Rule Mining
• jsapriori (JavaScript)
• arules + rCBA pruning (R packages)
Sensor + remote control + TV with installed application
9)
Rules-based user preferences
5)
Storage (MongoDB)
7) List of objects acting as candidates for recommendations
Recommendations (labelled candidates)
1. {Electronics} => {interest=positive}
2. {Televisions} => {interest=positive}
3. {Food} => {interest=negative}
4. ...
new object-3 is about "Food"
- labelled as "negative" for user1
- first matching rule: {Food} => {interest=negative}
new object-4 is about "Radios"
- labelled as "positive" for user1
- "Radios" is in taxonomy under "Electronics"
- first matching rule: {Electronics} => {interest=positive}
8)
RECOMMENDER SYSTEM MODULE
6)
Association Rule Classifier
• CBA algorithm (JavaScript)
InBeat module
Messages
REST communication
API proxy+load balancer (nginx)

Fig. 1. This figure explains the SmartTV use case powered by InBeat. Information on specific format of the REST calls is available in the developer documentation at http://inbeat.eu/gain/docs/rest. This figure is accompanied by a screencast, which is also available at inbeat.eu.

tion to coping with high volume of data, response to recommendation request had to be provided under 100 ms. The main criterion was the total number of successful recommendations. InBeat obtained a runner-up award in the 2013 edition of the challenge,[3] handling over 20 million recommendation requests over the three week evaluation period. In the 2014 edition[4], InBeat scored third [12].

5. Illustrative examples

The on-line documentation includes three illustrative scenarios. *(1) News recommender system* describes a use case for recommendation of articles on a news web site. InBeat collects interactions about consumptions of articles by users along with contextual features (e.g. daytime or location). Identified patterns in form of rules are then used as recommendations for other visitors. *(2) "SmartTV" deployment* presents a complex scenario, covering advanced topics such as sensor input (e.g. Microsoft Kinect positioned below the television), semantic description of content with entities and taxonomies, application of hand-coded rules for aggregation of implicit feedback into interest, building of rule-based user models and recommending new multimedia content to see based on the preference model. The scenario is illustrated in Fig. 1 and in a screencast that is included in supplementary material. *(3) External recommender system* is focused on a connection with other toolboxes. Since InBeat is composed of independent modules, they can be individually replaced with another tools. This scenario describes replacing of Preference Learning and Recommender System modules with MyMedia Lite collaborative filtering recommender.

The three diverse scenarios introduced above present the main novel approaches and advantages of the InBeat framework.

6. Conclusions

This paper presented InBeat, an open source recommender that allows to build a client side recommender handling data from various sources and sensors. InBeat can also provide a scalable server-side solution when deployed in node.js. The source code is available from https://github.com/KIZI/InBeat. This repository also contains documentation, tutorials and installation scripts (local development, server and cloud environment including docker).

Required metadata

Current executable software version

Table 2
Software metadata.

Nr.	(executable) Software metadata description	Please fill in this column
S1	Current software version	v1.0
S2	Permanent link to executables of this version	https://github.com/KIZI/InBeat/releases/tag/v1.0
S3	Legal Software License	BSD-3-Clause License
S4	Computing platform/Operating System	Linux, OS X, Microsoft Windows

(*continued on next page*)

[3] http://recsys.acm.org/recsys13/nrs/.
[4] http://www.clef-newsreel.org/previous-campaigns/newsreel-2014/.

Table 2 (*continued*)

Nr.	(executable) Software metadata description	Please fill in this column
S5	Installation requirements & dependencies	Node.js (≥ 4.6), MongoDB (≥3.0), R, Java 8 and nginx. Also available for Docker.
S6	link to user manual	https://github.com/KIZI/InBeat/blob/master/doc/main.md
S7	Support email for questions	jaroslav.kuchar@fit.cvut.cz

Current code version

Table 3
Code metadata.

Nr.	Code metadata description	Please fill in this column
C1	Current code version	v1.0
C2	Permanent link to repository for this code version	https://github.com/KIZI/InBeat/releases/tag/v1.0
C3	Legal Code License	BSD-3-Clause License
C4	Code versioning system used	git
C5	Software code languages, tools, and services used	JavaScript, R
C6	Compilation requirements, operating environments & dependencies	Node.js (≥ 4.6), MongoDB (≥ 3.0) and optionally R, Java 8 and nginx
C7	Link to developer documentation/manual	https://github.com/KIZI/InBeat/blob/master/doc/main.md
C8	Support email for questions	jaroslav.kuchar@fit.cvut.cz

Acknowledgements

The development of InBeat was supported via the EU LinkedTV project (FP7-287911). The writing of this article was partly supported by Czech Technical University in Prague by grant SGS14/104/OHK3/1T/18 (Jaroslav Kuchař), and by the Faculty of Informatics and Statistics, University of Economics in Prague through "long term institutional support for research activities" (Tomáš Kliegr).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.knosys.2017.07.026.

References

[1] J. Lee, M. Sun, G. Lebanon, PREA: personalized recommendation algorithms toolkit, J. Mach. Learn. Res. 13 (1) (2012) 2699–2703.

[2] A. Said, D. Tikk, P. Cremonesi, Benchmarking, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 275–300.

[3] H. Paulheim, J. Fürnkranz, Unsupervised generation of data mining features from linked open data, in: Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics, ACM, New York, NY, USA, 2012.

[4] B. Liu, W. Hsu, Y. Ma, Integrating classification and association rule mining, in: R. Agrawal, P. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining (KDD-98), 1998, pp. 80–86.

[5] Owl web ontology language reference, 2004, W3C Recommendation.

[6] M. Hahsler, B. Grün, K. Hornik, Arules - a computational environment for mining association rules and frequent item sets, J. Stat. Softw. 14 (15) (2005) 1–25.

[7] J. Kuchař, T. Kliegr, GAIN: web service for user tracking and preference learning - a smart TV use case, in: Proceedings of the 7th ACM Conference on Recommender Systems (RecSys '13), ACM, 2013, pp. 467–468.

[8] J. Leroy, F. Rocca, M. Mancas, R.B. Madhkour, F. Grisard, T. Kliegr, J. Kuchař, J. Vit, I. Pirner, P. Zimmermann, KINterestTV - towards non-invasive measure of user interest while watching TV, in: Innovative and Creative Developments in Multimodal Interaction Systems, Springer, 2013, pp. 179–199.

[9] T. Kliegr, J. Kuchař, Orwellian eye: video recommendation with Microsoft Kinect, in: Proceedings of the Conference on Prestigious Applications of Intelligent Systems (PAIS'14) collocated with European Conference on Artificial Intelligence (ECAI'14), IOS Press, 2014, pp. 1227–1228.

[10] P.N. Mendes, M. Jakob, A. Garcia-Silva, C. Bizer, DBpedia spotlight: shedding light on the web of documents, in: Proceedings of the 7th International Conference on Semantic Systems (I-Semantics'11), 2011.

[11] T. Kliegr, J. Kuchař, Benchmark of rule-based classifiers in the news recommendation task, in: Proceedings of the 6th International Conference of the CLEF Initiative, CLEF'15, Springer, 2015. To appear.

[12] B. Kille, T. Brodt, T. Heintz, F. Hopfgartner, A. Lommatzsch, J. Seiler, Overview of NEWSREEL 2014: summary of the news recommendation evaluation lab, Working Notes for CLEF 2014 Conference, No. 1180, 2014, pp. 790–801. in series CEUR Workshop Proceedings Aachen.