Names are not just Sound and Smoke: Word Embeddings for Axiom Selection

Ulrich Furbach[1] and Teresa Krämer[1] and Claudia Schon[2] ⋆
{tbergk, uli, schon}@uni-koblenz.de

[1] Institute for Computer Science, University of Koblenz-Landau, Germany
[2] Institute for Web Science and Technologies, University of Koblenz-Landau, Germany

Abstract. First-order theorem proving with large knowledge bases makes it necessary to select those parts of the knowledge base, that are necessary to prove the theorem at hand. We extend syntactic axiom selection procedures like SInE to use semantics of symbol names. For this, not only occurrences of symbol names but also semantically similar names are taken into account. We use a similarity measure based on word embeddings. An evaluation of this similarity based SInE is given using problems from TPTP's CSR problem class and Adimen-SUMO. This evaluation is done with two very different systems, namely the Hyper tableau prover and the saturation based system E.

1 Introduction

Automated theorem proving attempts to find a proof for the (unsatisfiability) of a set of formulae. Most problems, even those from benchmark suites like TPTP [28], consist of hand-coded sets of logical formulae and hence used to be relatively small. This situation changed dramatically since automated theorem provers have been used in contexts with large background knowledge. This can be large ontologies like SUMO [19], CYC [11], Yago [27] or mathematical libraries, where the theorem prover has to face millions of formulae that are potentially necessary to find a proof of the problem under consideration. Once a proof is found, it typically turns out that only a very small part of the vast background knowledge was necessary to find the proof. This leads to the following challenge: Given a large background knowledge base and a problem, find a (preferably small) subset of the knowledge base with which a proof for the problem can be found.

One of the most used approaches for this task is the SInE selection method [9]. It's basic idea is to determine symbols relevant for a problem at hand and to select formulae to be included into the proof search based on this relevancy. This method is very well suited for many problem areas and it is used by many theorem provers nowadays. However, in order to apply this method it is mandatory that the symbols are used in a consistent manner throughout all formulae in the knowledge base and in the axiomatisation of the problem. And, of course, this is fully consistent with the way the logical formulae are usually processed by theorem provers. To prove the unsatisfiability of a set of formulae it does not matter which symbols are used, it is only important

⋆ Work supported by DFG grant CoRg – Cognitive Reasoning. Author names are given in alphabetical order.

that symbols, e.g., a predicate symbol p, is used consistently in the entire formulae set. If every occurrence of p is substituted by q the problem to prove the unsatisfiability of the formulae remains the same. This is why the SInE selection strategy only counts occurrences of symbol names and does not consider the meaning of symbol names.

In certain areas such as commonsense reasoning, however, symbol names in knowledge bases are anything but random. This is very obvious as soon as we have to apply reasoning within the context of natural language. Examples are the commonsense reasoning benchmarks from [13] where the task is to determine which one of the given alternatives is the most plausible one:

My body cast a shadow over the grass. What was the CAUSE of this?
1. The sun was rising.
2. The grass was cut.

In the approach followed in the project CoRg [24] these sentences are transfered into first-order logic as a starting point for reasoning. This is done automatically by KNEWS [3], which is based on Boxer, a system for translating natural language input into various logical output formats [5]. For the premise together with the first alternative from our example we get:

$$\begin{aligned}\exists A, B(\exists C, D, E, F(rover(D,B) \wedge \exists G(rtopic(G,A) \wedge arisingC(G))\\ \wedge rthat(B,C) \wedge rpatient(D,E) \wedge ragent(D,F)\\ \wedge vcast(D) \wedge nshadow(E) \wedge nbody(F) \wedge rof(F,E)\\ \wedge nperson(E)) \wedge nsun(A) \wedge ngrassC(B)).\end{aligned}$$

So far, this approach follows very much the traditional paradigm in computational linguistics, namely to translate a natural language sentence into a metalanguage that represents the meaning of the sentence. In our case, first-order logic is used as a metalanguage, which has the advantage that powerful automatic reasoning systems can be used for further processing of the meaning of the sentence. The predicates in the resulting first-order logic formula are derived either directly from the words appearing in the English sentence or by portraying the connections. Examples for the former are $nbody$ or $vcast$ with a prefix indicating the word type (n for noun, v for verb), while the latter can be exemplified by $ragent(D,F)$ (r for role), which states that F (the noun body) has the role of the agent of D (the verb cast).

It is easy to see that the COPA task shown above can not be solved by simply applying a reasoning system to the formula generated from it. To reason that a shadow can be cast by an object and the sun needs more background knowledge. We use the ontology SUMO [19] as a knowledge base representing this background knowledge. The task of axiom selection is to consider those parts of SUMO for the reasoning process that contain the symbols of the formula or related ones. There are two problems: Firstly, a symbol might not be used in the ontology, but a synonym or a semantically similar one. E.g., $vcast$ does not occur in the ontology (and neither does $cast$), but the symbol $project$, which has a similar meaning occurs in the ontology and can be used instead of $vcast$. This can be easily solved by the use of WordNet [17], which immediately gives us $project$ as a synonym of $cast$. This is even supported by KNEWS, which determines

WordNet synsets for the symbol names in the generated formulae. The use of WordNet is further discussed in Sec. 4.3.

The second problem is much more subtle. There is no guarantee that the symbols of the background knowledge in SUMO fit semantically to the symbols in the generated predicate logic formulae — even if synonyms are taken into account.

This is why we also want to consider symbols in the background knowledge that are semantically similar to the symbols in the formulae. This offers the possibility to select more relevant axioms for the reasoning process from the background knowledge. For such a similarity measure there is an obvious choice, namely the distributional semantics of natural language, which is applied in many statistical natural language processing systems. J. R. Firth, one of the founders of this approach, put it like this:

> You shall know a word by the company it keeps. [8]

For such a semantics, a large corpus of natural language text is evaluated to find co-occurences of words. Words that occur more frequently together are more similar compared to words with less frequent co-occurences. Similar to this co-occurrence information are word embeddings, which we use in our approach. More specifically, we use the word embedding ConceptNet Numberbatch [26] to give us a real-valued vector for a symbol, which can be used to compare symbols with respect to similarity. In our example Numberbatch gives us a similarity value for $cast$ and $project$ of 0.19337063, which appears to be pretty low — the reason certainly is that $project$ is not only a verb but also a noun with a semantics very different from $cast$, and this is mirrored by the low value. On the other hand, if we compare sun and $shadow$ the value is 0.25696868 and for $grass$ and $shadow$ we get 0.09716037, which clearly indicates that the first alternative from the example above is more plausible. There are numerous systems which use this kind of distributional semantics together with machine learning methods for solving commonsense reasoning tasks very successfully [12,33,32].

We use both kinds of semantics — translation into logic as a metalanguage and distributional semantics — together. This offers the possibility of getting insights into the reasoning process and applying statistical methods at the same time. For this, we introduce a selection technique that considers the meaning of symbol names. The main contributions are:

- A selection technique that takes the meaning of symbol names into account by using word embeddings.
- A discussion of the need of mapping symbol names from the problem description and the background knowledge base to the vocabulary of the word embedding.
- An evaluation of the methods presented, which demonstrates that these methods can be helpful for reasoning.

This paper is organized as follows: after discussing related work in Sec. 2, in Sec. 3 we review the SInE selection method from [9]. Sec. 4 introduces our approach for combining word embeddings with the SInE method and in Sec. 5 this method is evaluated. Finally we discuss future work.

2 Related Work

There are a number of papers on axiom selection in large theories. Most of them are of purely syntactic nature, like [14], [23] or [9]. In [30] there is a semantic approach for axiom selection. It is based on the computation of models for subsets of the available axioms and by consecutively extending these sets. This approach is model based and the meaning of symbol names is not taken into account. A very similar aproach is desribed in [34], where it is used to search for conjectures which might be interesting.

In [10] the authors extend SInE by a method which is well known from market basket analysis, namely frequent item set mining. Instead of co-occurences given by word embeddings in our case, frequent item set mining is working directly on axiom sets in order to find symbols which frequently occur together. This information is then used for extending SInE. Frequent item mining is evaluated with prover E and with SPASS with the MPTP2078 benchmarks and with both provers there was no increase of performance with the extended SInE. Moreover, the authors state that in many cases the selection itself was too costly and often resulted in a timeout.

In [22] similarities of symbol names are used to define an extended unification method. The similarity is learned by neural networks which allow calculation of proof success with respect to a vector representation of symbol names. This method is used to improve the similarity measure and it is shown in this paper that it is well suited for inductive logic programming.

The observation that names are meaningful is explictly formulated and evaluated in [7]. This approach is based on knowledge graphs as they are used in the semantic web area. The authors test and evaluate their hypothesis, namely that the names of IRIs (Internationalized Resource Identifiers) carry a kind of social semantics, and come to the conclusion that semantics encoded in the names of IRIs significantly coincides with the formal meaning of the denoted resources. Hence the authors prove, that names in RDF graphs encode semantics. The authors use this insight to motivate the development of semantic web tools that make use of the meaning of names.

3 Selection Techniques for Large Knowledge Bases

The reasoning task we are considering in this paper consists of a very large set of axioms called the knowledge base (KB), a small set of further axioms $F_1, \ldots, F_n$ called assumptions and a query Q. The task is to show that KB together with $F_1, \ldots, F_n$ implies Q. In classical logic, this can be reduced to showing that $F_1 \wedge \ldots \wedge F_n \rightarrow Q$ is entailed by KB. As suggested in [9], we will denote $F_1 \wedge \ldots \wedge F_n \rightarrow Q$ as *goal*.

Given a query and a real-world KB, automated theorem provers (ATP) nowadays are able to compute proofs quite efficiently. However, once the underlying KB has reached a certain size, it is not feasible to consider the entire KB when trying to find a proof. Analysis of proofs on large KBs has shown that in the vast majority of cases only a very small part of the KB has been used for the proof. This observation is the basis of a strategy for reasoning in large KB: starting from the goal an attempt is made to determine a subset of the KB that contains sufficient knowledge to construct a proof.

In [9] the relevance-based selection strategy SInE is introduced. This selection is based on a trigger relation that determines which symbols trigger an axiom. Assume for a given axiom A and a symbol s occurring in A, a relation $triggers(s, A)$.

Definition 1 (Trigger-based selection [9]). *Let KB be a knowledge base, A be an axiom in KB and s be a predicate or function symbol occurring in KB. Let furthermore g be a goal to be proven from KB.*

1. *If s is a symbol occurring in the goal g, then s is 0-step triggered.*
2. *If s is k-step triggered and s triggers A ($triggers(s, A)$), then A is $k + 1$-step triggered.*
3. *If A is k-step triggered and s occurs in A, then s is k-step triggered, too.*

An axiom or a symbol is called triggered if it is k-step triggered for some $k \geq 0$.

In order to obtain a selection strategy, it is necessary to define which symbols trigger an axiom. A naive choice for this relation would be to determine that an axiom is triggered by all symbols occurring in it. Usually there are symbols like *subClass* or *hasPart* in KBs that are very common. Therefore, this naive trigger relation would result in the selection of almost all axioms. The SInE selection defines the triggers relation such that only the least common symbol in an axiom is allowed to trigger this axiom. This ensures that common symbols do not lead to the selection of all axioms.

Definition 2 (Trigger relation for the SInE selection [9]). *Let KB be a knowledge base and s be a symbol. Let furthermore $occ(s)$ denote the number of axioms in which s occurs in KB. Then the triggers relation is defined as follows:*

$$triggers(s, A) \text{ iff for all symbols } s' \text{ occurring in } A \text{ we have } occ(s) \leq occ(s') \quad (1)$$

Def. 1 together with Def. 2 define a trigger-based selection strategy. If a trigger-based selection selects all k-step relevant axioms, we call this selection SInE with *recursion depth* k in the following.

The SInE selection is an incomplete selection technique. This means that it can happen that given a goal, SInE selects a set of axioms with which no proof for the goal can be found (see Ex. 1 in Sec. 4.3 for an example). One reason for the incompleteness is SInE's fragility w.r.t. the number of occurrences of a symbol: even if two symbols occurring in an axiom occur almost equally often in the KB, only the one with the least number of occurrences is allowed to trigger the axiom. In order to soften this effect, the tolerance parameter, a real number $t \geq 1$, was introduced. To take this parameter into account, Equ. (1) can be changed such that

$$triggers(s, A) \text{ iff for all symbols } s' \text{ occurring in } A \text{ we have } occ(s) \leq t \cdot occ(s') \quad (2)$$

This allows not only the symbol with the least number of occurrences to trigger an axiom but also symbols with t times more occurrences.

The SInE selection strategy is successfully used by many provers. One property of SInE is that it completely ignores the names of the symbols in the KB. The behaviour of the selection does not change if a symbol called $beer$ would be renamed to p. In many areas where the names of symbols are not very meaningful, this is a legitimate approach. In commonsense KBs, the symbols usually have meaningful names and we suspect that this meaning can be very helpful for the selection process. In the following, we introduce a selection technique taking into account the symbol name's meaning which is implemented as a SInE extension.

4 Integration of Distributional Semantics into Axiom Selection

An example of a KB where symbols carry meanings, is SUMO [19,20]. To simplify its use, there is even a mapping offered that relates symbols occurring in SUMO to WordNet [17] synsets. SUMO is no exception; many other KBs in the commonsense reasoning area, like Cyc [11], also offer such mappings to WordNet or include the WordNet class hierarchy like Yago [27]. This illustrates the importance of symbol names in this area.

In current selection techniques, however, the meaning of the symbol names used in a KB is completely ignored. Intuitively, a symbol like *beer* in the goal should not only lead to the selection of axioms triggered by the symbol *beer*, but also to the selection of axioms triggered by symbols with a similar meaning, like *pilsner* for example. This is why we aim at including the meaning of the symbol names in the selection process. We semantically guide the selection process by integrating word-embeddings into the selection process by replacing the above trigger relation with one that takes a word embedding into account. Using this selection technique together with a theorem prover leads to a hybrid approach using both reasoning and statistical methods.

4.1 Distributional Semantics

The area of distributional semantics researches theories for quantifying semantic similarities between words based on their distributional properties in large text corpora. More specifically, distributional semantics is based on the distributional hypothesis [18], which states that linguistic elements with a similar distribution have a similar meaning. A currently very popular technique in the field of distributional semantics is word embeddings [16], which map words or phrases from a given vocabulary to vectors of real numbers. The rough idea behind the construction of word embeddings, as performed by methods like Word2vec [15], is to first determine word co-occurrences on a large text corpus and store them in a matrix. If the vocabulary V of the text corpus consists of $|V|$ words, this step leads to a $|V|$ times $|V|$ matrix. In the next step, Word2vec performs a dimensionality reduction that leads to a smaller matrix. Each column of the resulting matrix corresponds to the vector representation of a word. A nice property of these word embeddings is that relative similarities of the vectors correlate with semantic similarity. This means that the vector representations of $beer$ and $pilsner$ are closer together than the representations of $beer$ and $stone$. More precisely, the similarity of two words w_1 and w_2 from the vocabulary can be calculated by as the cosine similarity of their vector representations x_{w_1} and x_{w_2}:

$$\frac{x_{w_1} \cdot x_{w_2}}{||x_{w_1}||\ ||x_{w_2}||} \tag{3}$$

Where $x_{w_1} \cdot x_{w_2}$ denotes the dot product and $||x_{w_1}||$ the magnitude of vector x_{w_1}. The value of this cosine similarity lies between -1 for completely opposite vectors to 1 for the same vectors.

Surprisingly, simple algebraic operations on the vector representations can be used to answer questions about analogies. For example the question: Which word w has the same relationship to *king* as *woman* to *man*? can be answered by calculating $x_y = x_{woman} - x_{man} + x_{king}$ where x_w denotes the vector representation of word w normalized to unit form. The vector x'_y with the greatest cosine similarity to x_y yields the expected answer $y = queen$.

The word embedding toolkits Word2vec [15] and GloVe [21] are able to learn vector representations for given text corpora. We exploit the possibility of using word embeddings to provide the k most similar words for a given word, and integrate this information into the selection of axioms. In the following, we assume a given word embedding. The word embedding we use in our experiments is ConceptNet Numberbatch [26], which was derived using ConceptNet, Word2vec, GloVe, and OpenSubtitles 2016 [31].

Definition 3 (Set of k vectors most similar to x_v). *Let V be a vocabulary and f a word embedding, i.e. $f : V \rightarrow \mathbb{R}^n$, $k \in \mathbb{N}$, $|V| > k$ and $x_v \in \mathbb{R}^n$ a vector. Then $simvecs_f(x_v, k)$, the set of k vectors in $f(V)$ most similar to vector x_v, is defined as*

$$simvecs_f(x_v, k) = \begin{cases} \{x_1, \dots, x_k\} & \textit{if } x_v \in f(V) \\ \emptyset & \textit{else} \end{cases}$$

such that

- $\{x_1, \dots x_k\} \subseteq f(V)$,
- $|\{x_1, \dots, x_k\}| = k$ *and*
- *there is no* $x_j \in f(V)$ *with* $x_j \notin \{x_1, \dots, x_k\}$ *and* $\frac{x_j \cdot x_v}{||x_j||\, ||x_v||} > \frac{x_i \cdot x_v}{||x_i||\, ||x_v||}$ *for some* $x_i \in \{x_1, \dots, x_k\}$.

Furthermore, $simwords_f(w, k)$, the set of words similar to word w, is defined as

$$simwords_f(w, k) = \begin{cases} \{w' \in V \mid f(w') \in simvecs_f(f(w), k)\} & \textit{if } w \in V \\ \emptyset & \textit{else} \end{cases}$$

Next, we show how to integrate a word embedding into the selection process.

4.2 Using Similarities for Axiom Selection

Using the example of the frequently used selection SInE, we now show how word embeddings can be integrated into the selection process. For this the trigger relation must be exchanged. The trigger-based selection presented in Def. 1 can be used unchanged. In the following definition we assume the set of predicate and function symbols coincide with the set of words in the vocabulary of the used word embedding. We do this only to increase the readability of the definition. Since this assumption does not apply in practice, the next section presents details on how to map the symbols from the KB to the vocabulary of the embedding and vice versa.

Definition 4 (Word embedding enhanced trigger relation). *Let KB be a knowledge base with Σ the set of function and predicate symbols occurring in KB, A an axiom in KB, $s \in \Sigma$ be a symbol and $k \in \mathbb{N}$. Let furthermore $f : V \to \mathbb{R}^n$ be a word embedding with vocabulary $V = \Sigma$. Then the word embedding enhanced set of symbols triggering axiom A is defined as follows:*

$$simtriggers_f(A, k) = \bigcup_{s \in \{s' \mid triggers(s', A)\}} (\{s\} \cup (simwords_f(s, k)))$$

Intuitively, not only the rarest symbol s is allowed to trigger an axiom but also all symbols which are among the k most similar symbols of s according to word embedding f. This can result in an axiom like $subClass(beer, beverage)$ to be triggered by $pilsner$ which does not occur in the axiom but is clearly related to content of the axiom.

Setting $triggers(s, A)$ iff $s \in simtriggers_f(A, k)$ in Def. 1 for some embedding f and some $k \in \mathbb{N}$, results in a selection technique that takes word embedding f into account.

Def. 4 assumes that the symbols occurring in the KB coincide with the vocabulary of the word embedding. Next we describe problems occurring if this assumption is not true and we show how to solve these problems.

4.3 Relating Symbol Names to the Vocabulary of a Word Embedding

Usually, a KB does not come in combination with a suitable word embedding. There are a variety of word embeddings that can be downloaded and used directly. For the integration of a word embedding into axiom selection it is important that the word embedding used matches the KB from which the selection is to be made. In other words, the embedding should be calculated on a text that matches the KB thematically. Ideally, for example, a word embedding learned on the text of Wikipedia would be used for selection on Yago, since Yago contains the knowledge of Wikipedia. Even if there is a word embedding which is close to the content of the KB, we cannot assume all predicate and function symbols of the KB to have vector representations in the embedding. A disadvantage of word embeddings is that they usually do not distinguish between the different meanings of a word. This means, for example, that the meaning of the verb *project* together with the meaning of the noun *project* is represented by the same vector. Therefore, WordNet mappings for word embeddings are usually not offered. However, when relating symbol names with elements in the embedding's vocabulary, an existing WordNet mapping for the KB can be used. Such a WordNet mapping of a KB is a mapping of the symbol names occurring in the KB to WordNet synsets.[3] A WordNet synset represents a *set of synonyms*, meaning a set of words with a similar meaning. Fig. 1 shows some of the noun senses in WordNet for the word *grass*.

One way to relate a symbol name s of the KB to elements in the word embedding's vocabulary is to check for each symbol name, if the symbol name occurs as a word in the vocabulary. We create a relation rel from this information such that $rel(s, s)$ for all

[3] WordNet mappings usually also contain information about *subclass* or *instance of* relations. Since these relations are not relevant for this paper, they are omitted.

- S: (n) grass (narrow-leaved green herbage: grown as lawns; used as pasture for grazing animals; cut and dried as hay)
- S: (n) supergrass, grass (a police informer who implicates many people)
- S: (n) pot, grass, green goddess, dope, weed, gage, sess, sens, smoke, skunk, locoweed, Mary Jane (street names for marijuana)

Fig. 1: Some of the noun senses of the word *grass* in WordNet. Each line represents one synset of the word *grass*. The text given in parentheses represents the sense. The synset shown in the third line contains the synonyms *supergrass* and *grass* and has the sense *a police informer who implicates many people*.

symbols names s that are contained in the embedding's vocabulary. If the symbols have meaningful names, this can produce a relation covering many symbol names.

To achieve a higher coverage of the relation between symbol names and elements in the word embedding's vocabulary, the WordNet mapping of the KB can be used as follows: the symbol name s is first mapped to a WordNet synset using the WordNet mapping and then for all synonyms l belonging to this synset it is checked if l is in the word embedding's vocabulary. Meaning that for all synonyms l occurring in the vocabulary of the embedding $rel(s, l)$ is added to the relation.

In the following definition, we omit all further information in WordNet and consider a synset to be a set of synonyms. Therefore, a WordNet mapping is assumed to be a mapping $w : \Sigma \rightarrow 2^{Synsets}$ with $Synsets$ the set of WordNet synsets and each synset $S = \{l_1, \dots l_n\}$ a set of synonyms.

Definition 5 (Bridging relation between symbol names and a word embedding's vocabulary). *Let KB be a knowledge base with Σ the set of function and predicate symbols occurring in KB and $s \in \Sigma$ be a symbol. Let furthermore $f : V \rightarrow \mathbb{R}^n$ be a word embedding with vocabulary V. Let Synsets be the set of synsets in WordNet and $w : \Sigma \rightarrow 2^{Synsets}$ be a WordNet mapping of KB. Then the bridging relation $rel \subseteq \Sigma \times V$ is defined as $\{(s, s) \mid s \in \Sigma \cap V\} \cup \{(s, l) \mid S \in w(s)$ and $l \in S$ and $l \in V\}$.*

Note that the bridging relation rel is not total. So there may be symbol names not taking part in the relation and there may be words in the vocabulary not taking part in the relation. Furthermore, rel is not a function. So there may be distinct words w_1, w_2 in the vocabulary with $rel(s, w_1)$ and $rel(s, w_2)$ for some symbol s.

Definition 6 (Word embedding enhanced trigger relation using a bridging relation). *Let KB be a knowledge base with Σ the set of function and predicate symbols occurring in KB, A an axiom in KB, $s \in \Sigma$ be a symbol and $k \in \mathbb{N}$. Let furthermore $f : V \rightarrow \mathbb{R}^n$ be a word embedding with vocabulary V and rel a bridging relation between symbols in Σ and words in V. Then the word embedding enhanced set of symbols triggering axiom A using rel is defined as follows:*

$$
\begin{aligned}
simtriggers_f(A, k) = \bigcup_{s \in \{s' \mid triggers(s', A)\}} &(\{s\} \cup \{s'' \mid rel(s, w) \text{ and } w' \in simwords_f(w, k) \\
&\text{and } rel(s'', w')\})
\end{aligned}
$$

We call the resulting selection *Similarity SInE*. Note that for an empty bridging relation rel Def. 6 is identical to Def. 4. In general, $simtriggers_f(A, k)$ does not necessarily contain $k+1$ symbols. In general, it contains many less than $k+1$ symbols, since not all words contained in $simwords_f(w, k)$ can be mapped by rel to a symbol in Σ. In extreme cases, if no additional similar symbols can be added to the triggers relation due to an empty bridging relation, the resulting selection behaves exactly like SInE. Conversely, it could also happen that more than $k+1$ symbols end up in $simtriggers_f(A, k)$, since a symbol s can be associated with more than one word from V via rel, and for all these words the k most similar words are looked up in the word embedding. In Sec. 5, we present the experiences we have made in this respect in our experiments.

Example 1. We consider the following set of axioms, which is an extended version of the example given in [9]:

$$\forall X, Y, Z((subClass(X, Y) \land subClass(Y, Z)) \to subClass(X, Z)) \tag{4}$$

$$subClass(stone, liquid) \to \bot \tag{5}$$
$$subClass(petrol, liquid) \tag{6}$$
$$subClass(beverage, liquid) \tag{7}$$
$$subClass(beer, beverage) \tag{8}$$
$$subClass(guiness, beer) \tag{9}$$
$$subClass(pilsner, beer) \tag{10}$$
$$subClass(coolant, liquid) \tag{11}$$
$$subClass(lager, beer) \tag{12}$$
$$subClass(ale, beer) \tag{13}$$

The second column in Fig. 2 shows for each symbol the number of axioms it occurs in. This information is used to determine the set of axioms triggered by each symbol which is shown in the third column. The fourth column presents for each given symbol the set of similar symbols found by using rel to map the symbol name to the word embedding ConceptNet Numberbatch's vocabulary V, determining the $k = 100$ most similar words and using rel to map these words back (if possible) to symbols in the axiom set. For the example, we assume rel to be defined as $rel = \{(s, s) \mid s \in V \cap \Sigma\}$. Of course ConceptNet Numberbatch contains many other similar words that do not occur in the axioms. For example the 100 words most similar to $beverage$ in ConceptNet Numberbatch's vocabulary contain *tea_like_drink*, *beverages*, *red_bull*, *beer_run* and *in_drink* all of which can not be mapped back to the symbols in our example. The last column of Fig. 2 presents for each symbol the axioms that are similarity triggered by this symbol.

We now consider the following conjecture:

$$?subClass(beer, liquid) \tag{14}$$

With regular SInE symbols $subClass$, $beer$ and $liquid$ are 0-step triggered. $subClass$ triggers axiom (4) whereas symbol $beer$ and $liquid$ do not trigger further axioms (see third column Fig. 2). With the one selected axiom a proof cannot be found.

With Similarity SInE symbols $subClass$, $beer$ and $liquid$ are 0-step triggered. Symbol $subClass$ triggers axiom (4), $beer$ triggers axioms (7), (8),(10), (12) and (13) causing the symbols $beverage$, $liquid$, $pilsner$, $lager$ and ale to be 1-step triggered. The symbols $liquid$, $beverage$, $pilsner$, $lager$ and ale do not trigger any further axioms. Using the selected axioms, a proof can be found.

Symbol	Occurrences	$\mathbf{triggers}(\mathbf{s}, \mathbf{A})$	Similar symbols	$\mathbf{simtriggers_f}(\mathbf{s}, \mathbf{A})$
$subClass$	10	(4)		(4)
$liquid$	4			
$beer$	5		$ale, lager, pilsner, beverage$	(7), (8),(10), (12), (13)
$beverage$	2	(7), (8)	$beer$	(7), (8)
$petrol$	1	(6)		(6)
$stone$	1	(5)		(5)
$guiness$	1	(9)		(9)
$pilsner$	1	(10)	$lager, ale, beer$	(10), (12),(13)
$coolant$	1	(11)		(11)
$lager$	1	(12)	$pilsner, ale, beer$	(10), (12),(13)
ale	1	(13)	$lager, beer, pilsner$	(10), (12),(13)

Fig. 2: SInE's and Similarity SInE's trigger relation.

Using a tolerance of 2 for regular SInE causes SInE to select axioms (4), (7) and (8), which are sufficient to find a proof. The extent to which tolerance must be increased, however, depends strongly on the axioms present in the KB. For example, if we add the information about two more beers and two more liquids to the KB, the tolerance must be set to 3 for SInE to select all necessary axioms. If we then add three more beers and three more liquids, SInE selects all axioms necessary for the proof only at tolerance 4.5. In our example, the tolerance to be used depends on the relationship between the number of the occurrence of the $beverage$ and $liquid$ symbols as well as the relationship between the number of occurrences of the $beverage$ and $beer$ symbols. If the symbols $beer$ and $liquid$ occur much more frequently than the symbol $beverage$, a high tolerance is required to be able to select all necessary axioms with SInE. In contrast, Similarity SInE selection is less dependent on relationships between frequencies. Regardless of how many more beers and liquids are added, Similarity SInE always selects axioms (4), (7), (8),(10), (12) and (13).

5 Experiments

In order to evaluate Similarity SInE, we need KBs in which the symbols have meaningful names. In addition, the KBs used must be large, otherwise the use of selection techniques does not make sense. The LTB division of CASC [29] contains very large problems. Unfortunately, most of the problems in this division do not have meaningful symbol names. The CSR SUMO problems (CSR075 - CSR109 and CSR118) are a positive exception here, and therefore considered for the experiments. In addition to the CSR SUMO problems, we consider Adimen-SUMO for the experiments. It was obtained by translating a large part of SUMO into first-order logic [2]. The reason for choosing Adimen-SUMO over SUMO is the fact that the current version of Adimen-SUMO (v2.6), comes with a set of 8010 automatically generated white-box truth-tests [1]. These problems are supposed to be entailed by Adimen-SUMO and are therefore fit for the evaluation of Similarity SInE. Just like SUMO, Adimen-SUMO uses meaningful symbol names and provides a mapping to WordNet (which corresponds to

SUMO's WordNet mapping). Since there is no word embedding that fits exactly to CSR SUMO or Adimen-SUMO, the word embedding ConceptNet Numberbatch was used for the experiments, which contains a broad general knowledge.

In the experiments, we use SInE and Similarity SInE with different parameters as selection techniques. We adapted E's SInE implementation, which can be used as a stand-alone program, such that it takes similar symbols into account. The resulting program is used to perform the Similarity SInE selection [4]. E's implementation of SInE is used in the experiments to perform the SInE selection. Since the selection of the formulae is completely independent of the ATP used afterwards, we were able to use two different ATPs, Hyper [4] and E [25], after the selection step. Hyper is a tableau prover that is very well suited for tasks within cognitive reasoning because of its compact proof structure, efficient equality handling and confluence property. E is one of the best high performance saturation-based first-order systems.

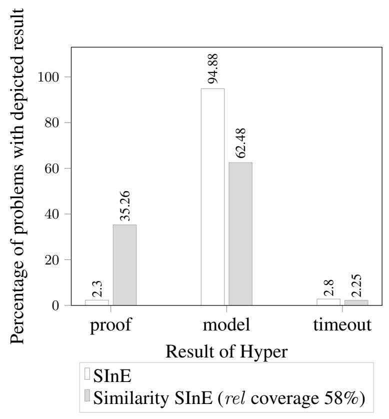
5.1 Relating Symbol Names to a Word Embedding's Vocabulary

For the experiments, we use the ConceptNet Numberbatch word embedding. Adimen-SUMO contains 3,917 symbols. Starting with an empty bridging relation 58% namely 2,279 of these symbols can be mapped to words in the vocabulary by throwing away prefixes and reformatting compound symbol names. E.g., symbol *c__FamilyBusiness* was mapped to word *family_business*, which occurs in ConceptNet Numberbatch's vocabulary, and led to adding *(c__FamilyBusiness*, *family_business)* to the bridging relation rel. For those 2,279 symbols for which this brute-force relating of symbols to words in ConceptNet Numberbatch's vocabulary worked, the $k = 100$ most similar words were determined. Of these words, on average, 2.4 (standard deviation 2.6) could be mapped back to symbol names of Adimen-SUMO. In total, 5,521 similar symbols were found and used in the triggers relation. The value of k has been determined by small normative experiments. Further values of k will be considered in future work.

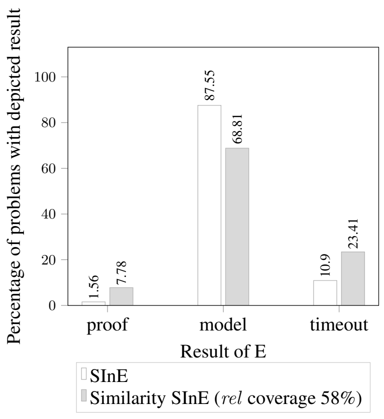
The coverage with this brute-force relating of symbol names to the word embedding's vocabulary can be improved to 63% using a bridging relation derived from the SUMO WordNet mapping.

Not all the CSR SUMO problems include the same axiom sets. This is why it is not possible to present one single number for the coverage of the bridging relation rel. Depending on the included axiom sets, the CSR SUMO problems use 3,452 to 34,239 different symbols. Using the brute-force method to create the bridging relation rel, the coverage was between 14% and 21%. Using SUMO's WordNet mapping increases coverage to 20% to 31%. Note that even using SUMO's WordNet mapping, the coverage of the bridging relation for CSR SUMO is far below the coverage of the bridging relation for Adimen-SUMO. In order to evaluate if the coverage of the bridging relation plays an important role for the selection and the subsequent ATP run, we performed experiments with both the bridging relation created by brute-force and the improved bridging relation (using the WordNet mapping).

[4] Implementation (git hash 'eeee0fc0b46c688ec25e08806d39ec8cea93cbc0') available at `https://gitlab.uni-koblenz.de/corg/similaritysine`.



(a) Result of Hyper on the 6,701 problems, it was not able to solve without selection.



(b) Result of E on 3,405 problems, it was not able to solve without selection.

Fig. 3: Percentage of the considered Adimen-SUMO problems where Hyper or E found a proof, a model or ran into a timeout broken down by the selection technique used. For all selections, tolerance one and recursion depth 1-6 was used.

In our implementation, we take advantage of the fact that WordNet gives a list of synonyms ordered by relevance. Therefore, for a given symbol name s we only add (s, l) to rel for the most relevant word l occurring in the list of synonyms.

5.2 Experimental Results

All tests were carried out on a computer featuring an Intel(R) Xeon(R) CPU E5-2699 v3 @ 2.30GHz (only two cores were used) with 8GB RAM. For the provers used in the experiments, we used a timeout of 15 seconds (cpu time). The timeout is based on the experimental results from [1], where for the majority of the Adimen-SUMO problem's proofs were found in up to 10 seconds.

Hyper and E on Adimen-SUMO For the Adimen-SUMO problems we first determined the set of problems for which selection makes sense for the used provers. For this, Hyper and E were run for all 8,010 Adimen-SUMO problems without using a selection technique. This resulted in 1,309 problems solved by Hyper and 4,605 problems solved by E without selection. For the sets of problems unsolved without selection (6,701 for Hyper and 3,405 for E) SInE and Similarity SInE (with 58% coverage of rel) were used to select formulae that were then fed into the respective prover. Fig. 3a shows the results for Hyper, Fig. 3b for E. Hyper was able to find a proof for 2.3% of the problems using the formulae selected by SInE and for 35.26% of the problems using the Similarity SInE selected formulae. E was able to find a proof for 10.9% of the problems using the SInE selected formulae and for 23.41% using the formulae selected with Similarity SInE. This 12.51% increase shows that both tableau-based and saturation-based provers benefit from the Similarity SInE selection technique.

Experiments with Different Tolerance Values To investigate the effect of the tolerance parameter on selection, we performed experiments with tolerance values between 1.0 and 6.0 with SInE, Similarity SInE (*rel* coverage 58%) and Similarity SInE (*rel* coverage 63%) on the CSR SUMO problems and on 1,000 randomly selected Adimen-SUMO problems. Both problem sets consist only of problems Hyper is not able to solve without selection. A comparison of Fig. 3a and the values for tolerance 1.0 in Fig. 4b shows that the 1,000 randomly selected Adimen-SUMO problems are representative for the entire set of problems. Furthermore Fig. 3 shows that Similarity SInE performs better than SInE for the given setup no matter the prover. Therefore the formulae selected by the different selection techniques are checked for satisfiability only with Hyper.

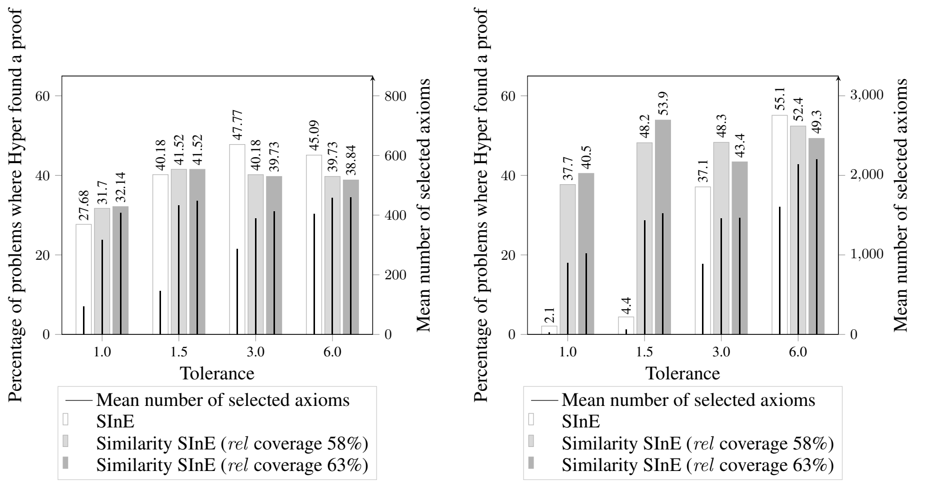
Fig. 4 shows that on the formulae selected by Similarity SInE with low tolerance values, significantly more proofs were found than on the formulae selected by SInE with the same tolerance. This is due to the fact that Similarity SInE selects more, but the additionally selected formulae are targeted towards the goal due to the use of the similar symbols. SInE can catch up by using higher tolerance values. Starting from a tolerance value of 3.0 for CSR problems and 6.0 for Adimen-SUMO problems, more proofs can be found on the formulae selected with SInE than on the formulae selected with Similarity SInE. For these high tolerance values, too many additional formulae are selected with Similarity SInE. Fig. 4b shows that a higher coverage of the bridging relation *rel* further improves the results for low tolerance values. On the CSR SUMO problems this effect is less clear, which we attribute to the fact that the CSR SUMO problems are generally more difficult to solve than the Adimen-SUMO problems. The fact that the improved coverage of the bridging relation for high tolerance values worsens the proportion of solved problems can be explained by the additionally selected formulae. For high tolerance values Similarity SInE with bridging relation with 58% coverage already selects too much. Similarity SInE with bridging relation with a coverage of 63% selects even more formulae for these high tolerance values, which exacerbates this problem.

Since the trigger relation of Similarity SInE extends SInE's trigger relation, Similarity SInE always selects a superset of the axioms selected by SInE. Fig. 4 presents the number of selected axioms for both SInE and Similarity SInE. Even with a low tolerance, Similarity SInE already selects significantly more axioms than SInE. Nevertheless, there are not many more timeouts with the Similarity SInE selected axioms than with the SInE selected axioms (see Fig. 3).

5.3 Discussion

The experiments on the Adimen-SUMO problems reveal that the coverage of the bridging relation of symbol names to the used word embedding's vocabulary is crucial. Thus it is worthwhile to put work into a good coverage. Furthermore, the experiments have shown that Similarity SInE is superior to SInE selection at low tolerance values. Although SInE provides similar or better results with higher tolerance values, it must be taken into account that suitable tolerance values are not known in advance and are heavily dependent on the fact distributions in the KB.

To the best of our knowledge, the TPTP [28] does not include problems where background knowledge, like word embeddings or WordNet mappings, can be used to find a proof. Since background knowledge plays an important role in practice, we propose



(a) Hyper on 224 CSR SUMO problems, it was not able to solve without selection.

(b) Hyper on 1,000 randomly selected Adimen-SUMO problems

Fig. 4: Percentage of the considered problems where Hyper found a proof using the selected formulae broken down by the selection technique and tolerance values used. For all selections recursion depth 1-6 was used.

an extension of the TPTP by a problem class (domain BGK), where it is necessary to consult background knowledge to solve the tasks. This background knowledge could be specified WordNet mappings, word embeddings or even text corpora which may be used by provers, when solving problems. The SUMO problems of the CSR domain together with the SUMO WordNet mapping as well as problems from Yago [27], Yago-SUMO [6] and Cyc [11] would be a good starting point for this new domain.

6 Conclusion and Future Work

In many areas the symbol names in KBs are not chosen arbitrarily, but have a meaning. Although this meaning of symbol names is an interesting source of information, it has been ignored by previous axiom selection techniques. In this paper, we introduced a selection technique that uses word embeddings to consider the similarities of symbol names and thus includes their meaning in the selection process. Since the presented method is based on the SInE selection technique, it is called Similarity SInE. In future work, we will investigate different values for the number k of similar symbol names considered by Similarity SInE. Furthermore, we want to determine to what extent Similarity SInE is suitable as a selection technique for commonsense reasoning benchmarks such as the Choice of Plausible Alternatives Challenge [13]. More specifically, we want to analyze whether for a given problem, Similarity SInE is able to extract thematically appropriate modules from a background KB.

References

1. J. Álvez, M. Hermo, P. Lucio, and G. Rigau. Automatic white-box testing of first-order logic ontologies. *CoRR*, abs/1705.10219, 2017.
2. J. Álvez, P. Lucio, and G. Rigau. Adimen-sumo: Reengineering an ontology for first-order reasoning. *Int. J. Semantic Web Inf. Syst.*, 8:80–116, 2012.
3. V. Basile, E. Cabrio, and C. Schon. KNEWS: Using Logical and Lexical Semantics to Extract Knowledge from Natural Language. In *Proceedings of the European Conference on Artificial Intelligence (ECAI) 2016 conference*, 2016.
4. M. Bender, B. Pelzer, and C. Schon. System description: E-krhyper 1.4. In M. P. Bonacina, editor, *Automated Deduction – CADE-24*, pages 126–134, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
5. J. R. Curran, S. Clark, and J. Bos. Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 33–36, Prague, Czech Republic, 2007.
6. G. de Melo, F. M. Suchanek, and A. Pease. Integrating YAGO into the suggested upper merged ontology. In *20th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2008), November 3-5, 2008, Dayton, Ohio, USA, Volume 1*, pages 190–193. IEEE Computer Society, 2008.
7. S. de Rooij, W. Beek, P. Bloem, F. van Harmelen, and S. Schlobach. Are names meaningful? quantifying social meaning on the semantic web. In P. T. Groth, E. Simperl, A. J. G. Gray, M. Sabou, M. Krötzsch, F. Lécué, F. Flöck, and Y. Gil, editors, *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, volume 9981 of *Lecture Notes in Computer Science*, pages 184–199, 2016.
8. J. R. Firth. *Papers in Linguistics 1934 - 1951: Rep.* Oxford University Press, 1991.
9. K. Hoder and A. Voronkov. Sine qua non for large theory reasoning. In N. Bjørner and V. Sofronie-Stokkermans, editors, *Automated Deduction – CADE-23*, volume 6803 of *Lecture Notes in Computer Science*, pages 299–314. Springer Berlin Heidelberg, 2011.
10. E. Kuksa and T. Mossakowski. Prover-independent axiom selection for automated theorem proving in Ontohub. In P. Fontaine, S. Schulz, and J. Urban, editors, *Proceedings of the 5th Workshop on Practical Aspects of Automated Reasoning co-located with International Joint Conference on Automated Reasoning (IJCAR 2016), Coimbra, Portugal, July 2nd, 2016.*, volume 1635 of *CEUR Workshop Proceedings*, pages 56–68. CEUR-WS.org, 2016.
11. D. B. Lenat. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38, 1995.
12. Z. Luo, Y. Sha, K. Q. Zhu, S. Hwang, and Z. Wang. Commonsense causal reasoning between short texts. In C. Baral, J. P. Delgrande, and F. Wolter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016.*, pages 421–431. AAAI Press, 2016.
13. N. Maslan, M. Roemmele, and A. S. Gordon. One hundred challenge problems for logical formalizations of commonsense psychology. In *Twelfth International Symposium on Logical Formalizations of Commonsense Reasoning, Stanford, CA*, 2015.
14. J. Meng and L. C. Paulson. Lightweight relevance filtering for machine-generated resolution problems. *J. Applied Logic*, 7(1):41–57, 2009.
15. T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013.
16. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th*

Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., pages 3111–3119, 2013.
17. G. A. Miller. WordNet: a lexical database for english. *Commun. ACM*, 38(11):39–41, 1995.
18. G. A. Miller and W. G. Charles. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28, 1991.
19. I. Niles and A. Pease. Towards a standard upper ontology. In *Proceedings of the international conference on Formal Ontology in Information Systems-Volume 2001*, pages 2–9. ACM, 2001.
20. A. Pease. *Ontology: A Practical Guide*. Articulate Software Press, Angwin, CA, 2011.
21. J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In A. Moschitti, B. Pang, and W. Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.
22. T. Rocktäschel and S. Riedel. End-to-end differentiable proving. In *NIPS*, pages 3791–3803, 2017.
23. A. Roederer, Y. Puzis, and G. Sutcliffe. Divvy: An ATP meta-system based on axiom relevance ordering. In *CADE*, volume 5663 of *Lecture Notes in Computer Science*, pages 157–162. Springer, 2009.
24. C. Schon, S. Siebert, and F. Stolzenburg. The CoRg project – cognitive reasoning. *KI*, 33(3), 2019. To appear.
25. S. Schulz. System Description: E 1.8. In K. McMillan, A. Middeldorp, and A. Voronkov, editors, *Proc. of the 19th LPAR, Stellenbosch*, volume 8312 of *LNCS*. Springer, 2013.
26. R. Speer, J. Chin, and C. Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In S. P. Singh and S. Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 4444–4451. AAAI Press, 2017.
27. F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A large ontology from Wikipedia and WordNet. *Web Semant.*, 6(3):203–217, Sept. 2008.
28. G. Sutcliffe. The TPTP problem library and associated infrastructure: From CNF to TH0, TPTP v6.4.0. *Journal of Automated Reasoning*, pages 1–20, 2 2017.
29. G. Sutcliffe. The 9th IJCAR automated theorem proving system competition CASC-J9. *AI Communications*, 31(6):495–507, 1 2018.
30. G. Sutcliffe and Y. Puzis. SRASS - A semantic relevance axiom selection system. In *CADE*, volume 4603 of *Lecture Notes in Computer Science*, pages 295–310. Springer, 2007.
31. J. Tiedemann. Parallel data, tools and interfaces in opus. In N. C. C. Chair), K. Choukri, T. Declerck, M. U. Dogan, B. Maegaard, J. Mariani, J. Odijk, and S. Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA).
32. L. Wang, M. Sun, W. Zhao, K. Shen, and J. Liu. Yuanfudao at semeval-2018 task 11: Three-way attention and relational knowledge for commonsense machine comprehension. In M. Apidianaki, S. M. Mohammad, J. May, E. Shutova, S. Bethard, and M. Carpuat, editors, *Proceedings of The 12th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT, New Orleans, Louisiana, June 5-6, 2018*, pages 758–762. Association for Computational Linguistics, 2018.
33. B. Williams, H. Lieberman, and P. H. Winston. Understanding stories with large-scale common sense. In A. S. Gordon, R. Miller, and G. Turán, editors, *Proceedings of the Thirteenth International Symposium on Commonsense Reasoning, COMMONSENSE 2017, London, UK, November 6-8, 2017.*, volume 2052 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

34. J. Zhang. System description: MCS: model-based conjecture searching. In H. Ganzinger, editor, *Automated Deduction - CADE-16, 16th International Conference on Automated Deduction, Trento, Italy, July 7-10, 1999, Proceedings*, volume 1632 of *Lecture Notes in Computer Science*, pages 393–397. Springer, 1999.