PBML

The Prague Bulletin of Mathematical Linguistics
NUMBER 105 APRIL 2016 111–142

A Minimally Supervised Approach for Synonym Extraction with Word Embeddings

Artuur Leeuwenberg[a], Mihaela Vela[b], Jon Dehdari[bc], Josef van Genabith[bc]

[a] KU Leuven – University of Leuven, Belgium
[b] Saarland University, Germany
[c] DFKI, German Research Center for Artificial Intelligence, Germany

Abstract

In this paper we present a novel approach to minimally supervised synonym extraction. The approach is based on the word embeddings and aims at presenting a method for synonym extraction that is extensible to various languages.

We report experiments with word vectors trained by using both the continuous bag-of-words model (CBoW) and the skip-gram model (SG) investigating the effects of different settings with respect to the contextual window size, the number of dimensions and the type of word vectors. We analyze the word categories that are (cosine) similar in the vector space, showing that cosine similarity on its own is a bad indicator to determine if two words are synonymous. In this context, we propose a new measure, relative cosine similarity, for calculating similarity relative to other cosine-similar words in the corpus. We show that calculating similarity relative to other words boosts the precision of the extraction. We also experiment with combining similarity scores from differently-trained vectors and explore the advantages of using a part-of-speech tagger as a way of introducing some light supervision, thus aiding extraction.

We perform both intrinsic and extrinsic evaluation on our final system: intrinsic evaluation is carried out manually by two human evaluators and we use the output of our system in a machine translation task for extrinsic evaluation, showing that the extracted synonyms improve the evaluation metric.

© 2016 PBML. Distributed under CC BY-NC-ND. Corresp. author: tuur.leeuwenberg@cs.kuleuven.be
Cite as: Artuur Leeuwenberg, Mihaela Vela, Jon Dehdari, Josef van Genabith. A Minimally Supervised Approach for Synonym Extraction with Word Embeddings. The Prague Bulletin of Mathematical Linguistics No. 105, 2016, pp. 111–142. doi: 10.1515/pralin-2016-0006.

1. Introduction

The research presented here explores different methods to extract synonyms from text. We try to do this using as little supervision as possible, with the goal that the method can be applied to multiple languages.

1.1. Motivation

The initial motivation for our research comes from machine translation (MT) evaluation. MT output to be evaluated is referred to as a *hypothesis translation*. A *reference translation* is a translation produced by a proficient human translator. To evaluate an MT system, hypothesis translations are compared with reference translations. This comparison is often done automatically.

While simple automatic evaluation approaches (Snover et al., 2006; Papineni et al., 2002; Doddington, 2002) are based on exact (sub-)string matches between hypotheses and references, more recent evaluation methods are using machine learning approaches (Stanojević and Sima'an, 2014; Gupta et al., 2015b; Vela and Tan, 2015; Vela and Lapshinova-Koltunski, 2015) to determine the quality of machine translation. More sophisticated approaches such as Meteor (Denkowski and Lavie, 2014; Banerjee and Lavie, 2005), Asiya (Gonzà lez et al., 2014), and VERTa (Comelles and Atserias, 2014), incorporate lexical, syntactic and semantic information into their scores, attempting to capture synonyms and paraphrases, to better account for hypotheses and references that differ in form but are similar in meaning.

Meteor computes an alignment between the hypothesis and reference to determine to what extent they convey the same meaning. Alignments are defined by what parts of the two sentences can match. Finding possible matches is done by means of four modules (1) exact matching, (2) stemmed matching, (3) synonym matching, and (4) paraphrase matching. Exact matching uses string identity between tokens, stemmed matching between stemmed tokens. Paraphrase matching employs a paraphrase database to match phrases which may not be string identical. The synonym module does the same for words and uses a synonym database resource. For example, the best alignment for the hypothesis sentence 1 and the reference sentence 2 is shown in Figure 1.

(1) *Hypothesis:*
The practiced reviewer chose to go through it consistently.

(2) *Reference:*
The expert reviewers chose to go through it in a coherent manner.

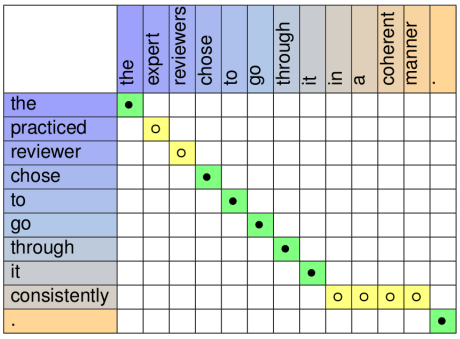

Figure 1. Meteor 1.5 alignment of hypothesis sentence 1, and reference sentence 2

In Figure 1, exact matches are indicated by black dots. The stemming module matched "reviewer" with "reviewers". The paraphrase module matched "consistently" with "in a coherent manner", and the synonym module matched "practiced" with "expert".

Three of these matching modules use language-dependent resources. Paraphrases and synonyms come from a pre-constructed lexical database, and stemming happens with a pre-trained stemmer. For this reason, not all modules are available for all languages. Currently in Meteor 1.5, the synonym module is only available for English. The module uses synonyms from the lexical database WordNet (Miller, 1995). Manual construction of lexical resources such as WordNet is time consuming and expensive, and needs to be done for each different language.

By contrast, large text resources are available for many languages. In our research we investigate whether, and if so to what extent, it is possible to automatically extract synonym resources from raw text using unsupervised or minimally supervised methods based on the *distributional hypothesis*: words that occur in the same contexts tend to have similar meanings (Harris, 1954). In particular we use word embeddings, i.e. dense distributional word vectors (Mikolov et al., 2013a), to compute similarity between words. We develop a new similarity metric, relative cosine similarity, and show that this metric improves the extraction of synonyms from raw text. We evaluate our method using both intrinsic and extrinsic evaluation: we use human evaluation to judge the quality of synonyms extracted and employ the extracted synonyms in the synonymy module of Meteor.

1.2. Word and Synonym

In most recent works on synonym extraction the synonyms from WordNet are used for evaluation. In WordNet, synonyms are described as "words that denote the same concept and are interchangeable in many contexts". In the current work, our notion of words is merely a string of characters. Since there is *homography*, i.e. one word can have different lemmas, with different meanings and origins, we modifiy this notion of synonyms slightly. We think of *synonyms* as words that denote the same concept and are interchangeable in many contexts, with regard to one of their senses.

1.3. Outline

In Section 2, we will proceed to describe the distributional word vectors we used in our experiments, and the related work in synonym extraction. In Section 3 we describe different experiments in which we explore synonym extraction using the continuous bag-of-words model and the skip-gram model. Section 4 describes and evaluates a few methods that introduce some supervision, such as using a part-of-speech tagger. In Section 5 we do an evaluation of a system that combines different proposed findings, for English and German. We evaluate manually, and additionally by using the extracted synonyms for the task of machine translation evaluation. Section 6 concludes the article by giving a summary of the findings and possibilities for future work.

2. Related Work

2.1. Distributional Word Vectors

Distributional word vectors, or *word embeddings*, are word representations that can be constructed from raw text, or a collection of documents, based on their context. The representation of each word will be a vector of numbers, usually real numbers. In some cases linguistic information, such as word dependency information, or morphological information, is also used during the construction process (Levy and Goldberg, 2014; Luong et al., 2013). These word vector representations can then be used to calculate, for example, word similarity and have a wide application domain.

In the last few years many new methods have been proposed to construct distributional word vectors based purely on raw text (Mikolov et al., 2013a; Pennington et al., 2014, *inter alia)*. Some methods also use the document structure that can be present in the data (Huang et al., 2012; Liu et al., 2015a,b).

In this work, we experiment mostly with word vectors trained using the *continuous bag-of-words model* (CBoW), and the *skip-gram model* (SG) developed by Mikolov et al. (2013a). It has been shown that these vectors, especially the skip-gram model, can also encode relations between words in a consistent way (Mikolov et al., 2013b). This means that they not only encode word similarity, but also similarity between pairs of words. For example, the offset between the vectors for "queen" and "king" lies very close to the offset between "woman" and "man", i.e. $v(\mathrm{queen}) - v(\mathrm{king}) \approx v(\mathrm{woman}) - v(\mathrm{man})$.

This property has been exploited to extract hypernyms from raw text by Fu et al. (2014) and Tan et al. (2015). The work of Fu et al. (2014) automatically learned, in a supervised way, a piece-wise linear projection that can map a word to its hypernym in the word vector space, for Chinese. To do this they clustered the vector offsets $(v_1 - v_2)$, and then found a projection for each cluster. Using this method they could successfully find hypernym pairs. Tan et al. (2015) searched for hypernym pairs in English. They also projected a word to its hypernym in the word vector space. However, instead of automatically learning this projection by using a thesaurus, they concatenated the words "is", and "a" into an "is_a" token in the corpus, and used this as projection. So, $v(w) + v(\mathrm{is\_a})$ would lie very close to the vector for the hypernym of word w.

Both the CBoW and the SG model can be seen as a simplified feedforward neural network, that is constructed from a word and its context. The architecture of the network is shown in Figure 2. CBoW word representations are optimized for predicting the word from its context, the surrounding words. SG word representations are optimized for predicting the context from the word, i.e. given the word, predicting its surrounding words.

In Figure 2, the word is represented as $w(t)$; the contextual window, here of size 2 (two words to the left, and two to the right), is represented as $w(t-2), w(t-1), w(t+1)$, and $w(t+2)$. The final word vector is built from the weights of the projection layer. During training, the window iterates over the text, and updates the weights of the network. Two training methods were described by Mikolov et al. (2013a), namely *hierarchical softmax*, and *negative sampling*. In (hierarchical) softmax, the weights are updated based on the maximization of log-likelihood. In negative sampling, the weights get updated based on whether or not the target word is drawn from the training set, or from a random distribution. The implementation in word2vec[1] has been shown to be quite fast for training state-of-the-art word vectors.

[1] https://code.google.com/p/word2vec/

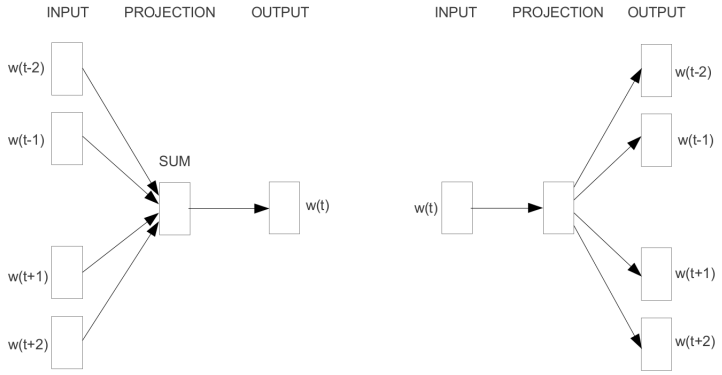

Figure 2. Continuous bag-of-words architecture on the left, and skip-gram on the right.

Depending on the application, it can be beneficial to modify pre-trained word vectors towards specific properties. Faruqui et al. (2015) refined a vector space using relational information, such synonymy and hypernymy, from a lexical database. For the task of antonym detection, Ono et al. (2015) transformed a pre-trained vector space by minimizing the similarity between synonyms and maximizing the similarity between antonyms. Since we would like to use as little supervision as possible, we did not resort to these particular methods.

2.2. Synonym Extraction

Many methods that have been developed for synonym extraction use three main ideas. Firstly, the distributional hypothesis (Van der Plas and Tiedemann, 2006; Agirre et al., 2009; Gupta et al., 2015a; Saveski and Trajkovski, 2010; Pak et al., 2015; Plas and Bouma, 2005). Secondly, the assumption that words that translate to the same word have the same, or a very similar, meaning (Van der Plas and Tiedemann, 2006; Gupta et al., 2015a; Saveski and Trajkovski, 2010; Lin et al., 2003) . And third, the use of linguistic patterns that are typical, or atypical for synonyms to occur in (Lin et al., 2003; Yu et al., 2002).

Van der Plas and Tiedemann (2006) used both distributional word similarity, and translational context for synonym extraction in Dutch. They used a large monolingual corpus to construct a measure for distributional similarity, which was based on grammatical relations. Furthermore, they used different

parallel corpora, and automatic alignment, for the construction of a translational context. A contexual similarity measure is constructed to rank the best synonym candidates. The authors remark that when only using distributional similarity there were some word categories that show up frequently but are not synonyms, but rather antonyms, (co)hyponyms, or hypernyms. When using the translational context, these error categories were less frequent, and more synonyms were found. In 2010, an adaptation of the method achieved 31.71% precision at the best candidate (P@1) for high frequency words (most frequent $\frac{1}{3}$ of the vocabulary), 16.22% for low frequency words (least frequent $\frac{1}{3}$), and 29.26% for remaining middle frequency words (van der Plas et al., 2010). Evaluation was done using a selection of 3000 words from Dutch EuroWordNet (Vossen, 1998).

It is very difficult to compare different methods of synonym extraction by only looking at their performance measures, as most papers use different ways to evaluate their approach. They use different word frequency ranges, language(s), textual resources, and gold standard synonyms. These can all have a large influence on the final evaluation.

The word categories mentioned by Van der Plas and Tiedemann (2006) seem to be a common problem when using purely distributional methods (Pak et al., 2015; Plas and Bouma, 2005; Lin et al., 2003). However, the advantage of using methods based on distributional properties is that the coverage is usually greater than that of manually constructed corpora, as Lin et al. (2003) also observed. They tackle the problem of discriminating synonyms from other strongly related words using linguistic patterns. They mention some English patterns in which synonyms hardly occur, like "from X to Y", and "either X or Y".

Rather than filtering by means of linguistic patterns, Yu et al. (2002) used particular patterns in which synonyms occur frequently. Their application domain was finding synonyms for gene and protein names. They found that in MEDLINE abstracts synonyms are often listed by a slash or comma symbol. This is probably a more domain dependent pattern. Some other patterns they found were "also called", or "known as", and "also known as".

In this work, we do not resort to a pattern based approach, as they are language and domain dependent.

3. Synonyms in Word Vector Space

In this Section we explain different experiments we carried out to analyze how synonyms behave in different word vector spaces. First, we analyze the effect of contextual window size, the number of dimensions, and the type of

word vectors on the precision of extraction, for English and German. Secondly, we look closely at the word categories that are (cosine) similar in the vector space. Then, we look at cosine similarity and introduce relative cosine similarity. Lastly, we examine the overlap of the most similar words in different vector spaces.

3.1. Data and Preprocessing

For English and German we use a 150 million word subset of the NewsCrawl corpus from the 2015 Workshop on Machine Translation[2]. As preprocessing for both languages, we apply lowercasing, tokenization, and digit conflation. In this work, we do not deal with multiword units. For example, for a separable verb in German or English (e.g. abholen / to pick up) can only be found as one word in infinitival or past perfect form (abgeholt/picked up).

We only consider the vocabulary of words that occur at least 10 times in the corpus to ensure that the vectors have a minimum quality. We randomly split the vocabulary into a training, development, and testing set with proportions 8:1:1 respectively. We used vocabularies S_{train}, and S_{dev} in the experiments to explore, and analyze the different methods described in the paper. After all initial experiments were done, we ran the experiments again using S_{test} instead of S_{dev} to evaluate our method. In Table 1, statistics about these vocabularies are given.

Language	Corpus	V	$V_{\geq 10}$	$S_{V_{\geq 10}}$	V_{train}	S_{train}	V_{dev}	S_{dev}	V_{test}	S_{test}
English	150M	650.535	136.821	21.098	109.454	16.882	13.681	2.116	13.683	2.100
German	150M	2.421.840	279.325	16.304	223.458	13.056	27.933	1.599	27.933	1.649

Table 1. Dataset Statistics: V indicates the size of the full corpus vocabulary, $V_{\geq 10}$ indicates the vocabulary size for words with counts greater than or equal to 10. S_x indicates the number of words for which at least one synonym is known, that also occurs in $V_{\geq 10}$.

For evaluation, we use the synonyms from WordNet 3.0 for English, and GermaNet 10.0 for German. In both WordNet and GermaNet words carry a corresponding part-of-speech. In WordNet these are nouns, verbs, adjectives, and adverbs. In GermaNet, synonyms are given for nouns, verbs, and adjectives. Because a given word's part of speech is unknown here, we consider the

[2]http://www.statmt.org/wmt15/translation-task.html

synonyms of each word to be those of all the parts of speech it can potentially have in WordNet or GermaNet.

3.2. Evaluation

We evaluate several experiments in terms of precision, recall and f-measure. *Precision* (P) is calculated as the proportion of correctly predicted synonym word pairs from all predictions. Because synonymy is symmetric, we consider the word pair (w_1, w_2) equivalent to (w_2, w_1) during evaluation. *Recall* (R) is calculated as the proportion of synonym pairs that were correctly predicted from all synonym pairs present in WordNet, or GermaNet. In the experiments we sometimes only search for synonyms of words from a subset of the vocabulary (S_{train} or S_{test}). In this case, recall is calculated only with regard to the synonym pairs from WordNet or GermaNet that involve a word from the mentioned subset. *F-measure* is given by:

$$F = 2 \cdot \frac{P \cdot R}{P + R}$$

3.3. Quantitative Analysis of Training Parameters

In this experiment, we trained CBoW, SG, and *Global Vectors* (GloVe) (Pennington et al., 2014) with different training parameters, and evaluated synonym precision for the $\{1^{st}, 2^{nd}, 4^{th}\}$-most-similar word(s), for vocabulary S_{train}. With similarity we refer to cosine similarity. The hyperparameters we varied are the contextual window size, and the number of dimensions of the vectors. The window size varied over {2, 4, 8, 16, 32}. The number of dimensions varied over {150, 300, 600, 1200}. The experiment is conducted for both English and German, and used 150M training tokens per language. We fixed the number of training iterations: 5 for CBoW and SG, and 25 for GloVe. For CBoW and SG training we used negative sampling with 5 negative samples [3].

The results for the CBoW and SG vectors, for both English and German, are shown in Tables 2, 3, 4, and 5. We excluded the results for the GloVe vectors, as they showed lower precision than SG and CBOW, and we did not use them in further experiments. The general trends of the GloVe vectors were that they had higher precision for larger window sizes. The vectors with highest precision of 0.067 for English were of dimension 300, with a window size of 32. For German, the highest precision was 0.055, and the vectors were of dimension 1200, with a window size of 32 as well.

[3] These are the default values given by the respective authors.

English CBoW																				
dim.	150					300					600					1200				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.077	0.076	0.072	0.066	0.058	0.084	0.083	0.079	0.072	0.068	0.086	**0.086***	0.081	0.074	0.068	0.083	0.083	0.082	0.073	0.067
P-2	0.058	0.056	0.055	0.051	0.046	0.062	0.061	0.059	0.055	0.052	0.063	0.063	0.060	0.056	0.052	0.061	0.061	0.060	0.055	0.050
P-4	0.039	0.039	0.038	0.036	0.032	0.042	0.042	0.041	0.039	0.036	0.043	0.043	0.042	0.039	0.036	0.042	0.042	0.041	0.039	0.036

*Table 2. Precision for different window sizes and number of dimensions, using the **CBoW** model, for **English**.*

English Skip-gram																				
dim.	150					300					600					1200				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.069	0.062	0.055	0.048	0.044	0.069	0.062	0.053	0.048	0.044	0.066	0.059	0.046	0.043	0.039	0.061	0.051	0.039	0.034	0.030
P-2	0.050	0.045	0.040	0.037	0.034	0.050	0.046	0.039	0.036	0.033	0.049	0.044	0.035	0.032	0.030	0.045	0.039	0.029	0.026	0.024
P-4	0.034	0.032	0.028	0.026	0.024	0.034	0.032	0.028	0.025	0.024	0.033	0.030	0.025	0.023	0.021	0.031	0.026	0.020	0.018	0.017

*Table 3. Precision for different window sizes and number of dimensions, using the **Skip-gram** model, for **English**.*

German CBoW																				
dim.	150					300					600					1200				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.073	0.082	0.082	0.083	0.080	0.076	0.084	0.086	0.086	0.082	0.076	0.087	**0.089***	0.088	0.080	0.076	0.083	0.086	0.085	0.081
P-2	0.052	0.057	0.057	0.058	0.056	0.054	0.060	0.062	0.061	0.059	0.054	0.060	0.062	0.062	0.059	0.053	0.059	0.062	0.060	0.058
P-4	0.034	0.036	0.038	0.038	0.037	0.036	0.039	0.041	0.040	0.039	0.035	0.039	0.041	0.041	0.040	0.035	0.039	0.041	0.040	0.039

*Table 4. Precision for different window sizes and number of dimensions, using the **CBoW** model, for **German**.*

German Skip-gram																				
dim.	150					300					600					1200				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.065	0.068	0.066	0.064	0.064	0.064	0.069	0.064	0.062	0.060	0.063	0.064	0.057	0.051	0.049	0.061	0.059	0.046	0.039	0.035
P-2	0.048	0.049	0.049	0.046	0.046	0.048	0.049	0.048	0.045	0.046	0.047	0.046	0.042	0.039	0.037	0.046	0.043	0.035	0.030	0.027
P-4	0.032	0.033	0.032	0.032	0.031	0.033	0.033	0.032	0.031	0.031	0.031	0.031	0.029	0.027	0.026	0.031	0.029	0.025	0.022	0.020

*Table 5. Precision for different window sizes and number of dimensions, using the **Skip-gram** model, for **German**.*

In general, it can be noticed from Tables 2, 3, 4, and 5 that the CBoW vectors give higher precision than SG for both German and English. A reason for this could be that CBoW vectors tend to be slightly more syntactical compared to SG vectors. It could be that the syntactical constraint on synonyms, as they are to appear in similar contexts, has enough influence for CBoW vectors to perform better.

It can also be noticed that for English, smaller contextual windows (2 and 4) generally give better precision, for both CBoW and SG vectors. For German, the optimal window size lies between 8 and 16 for CBoW, and around 4 for SG vectors. The difference in optimal window sizes between English and German could be due to the difference in types of synonyms that are available. WordNet contains synonyms for nouns, verbs, adjectives and adverbs, whereas GermaNet does not include synonyms for adverbs. It could be that adverbs require only a small contextual window to be predicted, compared to nouns, verbs, and adjectives. Another observation that can be made is that for both English and German the optimal window size for SG tends to be slightly lower than for CBoW vectors. Again, this can be due to training difficulty. A larger window can make the training of the SG model more difficult, as a bigger context is to be predicted from one word.

To get an impression of the performance if we would use the most-similar words as synonyms, we calculated precision, recall and f-measure on the test set $S_{\texttt{test}}$. For English, using the CBoW vectors of dimension 600 with window size 4, precision is 0.11, recall 0.03, and f-measure is 0.05. For German, using a CBoW model of dimension 600 with a window size of 8, precision is 0.08, recall is 0.05, and f-measure 0.06. For both languages these scores are very low. In the next section, we look at some frequent error categories, with the goal to get more insight into the reason behind these low scores.

3.4. Distributionally Similar Words

Only looking at precision, calculated on WordNet or GermaNet, allows us to compare different vector spaces with regard to finding synonyms. However, it might not reflect actual precision, due to lack of coverage of WordNet and GermaNet. Also, it gives only few cues for possible improvements.

For this reason, we also looked more in depth at the most similar words. For 150 randomly chosen English words from $S_{\texttt{train}}$ we looked at the most-similar word, as well as the 2nd-most-similar words, and categorized them. This was done manually. Categories were made based on what was found during the analysis. The word vectors used to create the most similar and 2nd-most-similar words were from the CBoW model of dimension 600, with

window size 2, from the previous experiment. The results from this analysis are shown in Table 6. The categories we found are the following:

- *WordNet-Synonyms*: Synonyms as given in WordNet.
- *Human-judged Synonyms*: Synonyms judged by a fluent, but non-native, English speaker.
- *Spelling Variants*: Abbreviations, differences between American and British spelling, and differences in hyphenations.
- *Related:* The two words are clearly semantically related, but not consistently enough to make a separate category.
- *Unrelated / Unknown*: The relation between the two words is unknown.
- *Names*: Names of individuals, groups, institutions, cities, countries or other topographical areas.
- *Co-Hyponyms*: The two words share a close hypernym.
- *Inflections / Derivations*: Inflections or derivations other than plural.
- *Plural*: The found word is the plural version of the given word.
- *Frequent collocations*: The two words occur frequently next to each other.
- *Hyponyms*: The found word is conceptually more specific.
- *Contrastive*: There is an opposition or large contrast between the meaning of the two words.
- *Hypernym*: The found word is conceptually more general.
- *Foreign*: A non-English word.

What can be noticed from Table 6 is that the number of human-judged synonyms is about twice as large as the number of synonyms given by WordNet, even though WordNet considers spelling variants also to be synonyms. This suggests that the actual precision may lie a corresponding amount higher. Where WordNet would give a precision of 0.12 for this set of words, the human annotation gives 0.25. A reason for this large difference can be that resources like WordNet are usually constructed by manually adding the synonyms for a given word. This requires the annotator to think of all the word senses of a word, and their synonyms. This can be a difficult task. Here, the two words are presented and the question is whether they are synonyms. It is probably easier to find the corresponding word senses of both words in this case.

The two biggest error categories are the related words, and unknowns. Since both categories are rather vaguely defined, and consisting of many subcategories we will not go into much more detail on these. There appears some overlap with the error types that were also found by Lin et al. (2003), Plas and Bouma (2005) and Pak et al. (2015), namely co-hyponyms, and hyponyms. However, contrastives and hypernyms are not as frequent in our experiment. Some other major error categories we found are different types of inflections

Category	1st-most-similar	2nd-most-similar	Example
WordNet-Synonyms	18	7	laundry / washing
Human-Synonyms	29	20	masking / obscuring
Spelling Variants	8	4	commander / cmdr
Related	27	33	head-on / three-vehicle
Unrelated/Unknown	13	20	gat / por
Names	15	15	consort / margherete
Co-hyponyms	15	13	sunday / saturday
Inflections/Derivations	12	10	figuring / figured
Plural	11	2	tension / tensions
Frequent Collocations	7	5	dragon / lantern
Hyponyms	5	12	swimsuit / bikini
Contrastive	3	7	rambunctious / well-behaved
Hypernym	2	4	laundry / chores
Foreign	2	4	inhumation / éventualité

Table 6. Counts per category for the most similar word and second most similar word, of 150 randomly chosen English words, in a CBoW model of dimension 600 with a window size of 2.

and derivations, and in particular plurals. This category is not a major problem for our application—machine translation evaluation—as the inflections might already have been matched by the stem module of Meteor. Another category that is fairly frequent involves names. The reason is probably that names might not have many single-word synonyms. The error category of frequent collocations can be explained by the fact that both words usually occur together, and are thus trained on a set of very similar contexts.

3.5. Relative Cosine Similarity

One idea we tested with the goal of improving precision was to only consider word pairs that have very high cosine similarity. In practice this would mean setting a threshold, and only consider those word pairs that have a cosine similarity higher than the threshold. Our expectation was that synonyms are most similar compared to the other word relations. We plotted precision, recall and f-measure on S_{train} against the cosine similarity threshold. This is shown in Figure 3.

What we found however, is that even increasing the cosine similarity threshold does not give an increase in precision. It does not even reach the precision we achieved from our baseline of taking the most-similar word. This indicates that cosine similarity on its own is not a good indicator for synonymy. Still, we get higher precision with choosing the most-similar word. We man-

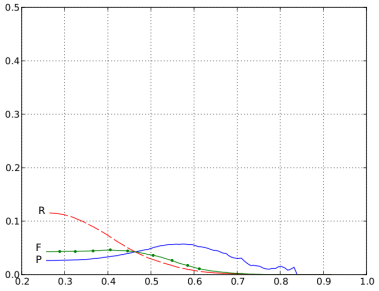

Figure 3. Precision, recall, and f-measure on S_{train} plotted against the cosine similarity threshold.

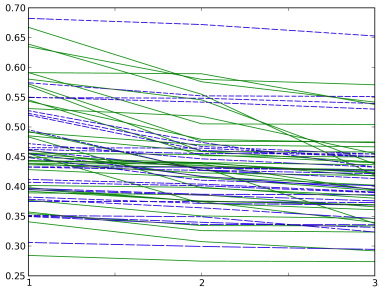

Figure 4. Cosine similarity against n-most similar position, for the 3-most-similar words, where the most-similar word is a synonym or a related word (dashed).

ually looked at the top 10 most-similar words of the 150 words from the previous section, and their cosine similarity. We noticed that when a synonym, inflection or contrastive occurs in the top 10, their cosine similarity is usually much higher than that of the other words in the top 10. That is, the difference in cosine-similarity between the most-similar word, and the second-most-similar word is very high for these categories. When we looked at this for other categories such as co-hyponyms, unknowns, and simply related words, this was not the case. This can be seen when we plot the cosine similarity of the 3-most-similar words for synonyms, and related words taken from the previous experiment.

This is plotted in Figure 4, from which two things can be noticed. Firstly, it is hardly possible to separate the start, at position 1, of the solid lines (synonyms) from the dashed lines (related words) by means of a horizontal cosine threshold. This corresponds to the observation we made earlier, that a cosine similarity threshold does not increase precision. Secondly, many solid lines tend to decrease, and many dashed lines stay relatively horizontal. This indicates that, in general, the difference in cosine similarity between synonyms and other similar words (from the top 10) is greater compared to, say, co-hyponyms. We also found this bigger difference for inflections and contrastives. This observation could be used to increase precision, as we can possibly filter out some co-hyponyms, related words, and unknowns.

To test this hypothesis, we developed a different measure to calculate similarity. We calculate similarity relative to the top n most similar words. We calculate *relative cosine similarity* between word w_i and w_j as in Equation 1.

$$\mathrm{rcs}_n(w_i, w_j) = \frac{\mathrm{cosine\_similarity}(w_i, w_j)}{\sum_{w_c \in \mathrm{TOP}_n} \mathrm{cosine\_similarity}(w_i, w_c)} \quad (1)$$

This will give words that have a high cosine similarity compared to other words in the top 10 most-similar words a high score. If all words in the top 10 most-similar words have almost an equal cosine similarity, they will get a lower score. When we do the same experiment again, changing the similarity threshold and plotting precision, recall and f-measure, using relative cosine similarity instead, we can see that precision goes up when we increase the rcs-threshold. This is shown in Figure 5. In Figure 6, it can also be noticed that when we look at the relative cosine similarity for the three most-similar words of words where the most similar word is synonym (solid), or simply a related word (dashed), part of the synonyms is now separable from the related words by a horizontal line, i.e. an rcs-threshold. This confirms our earlier hypothesis that synonyms have a bigger difference in cosine similarity with respect to other similar words.

We used WordNet synonyms here to calculate precision, recall and f-measure, and find the optimal rcs_{10}-threshold. However, what can be noticed is that the tilting point for the precision to go up lies at an rcs_{10}-threshold of 0.10. This is not a coincidence, as 0.10 is also the mean of the relative cosine similarities for 10 words. If a word has an rcs_{10} higher than 0.10, it is more similar than an arbitrary similar word. If synonyms are more similar compared to other similar word relations, we can find this tilting point at $\frac{1}{n}$, where n is the number of most-similar words we consider for calculating rcs_n.

Thus relative cosine similarity gives us the flexibility to increase precision, at the cost of recall, if needed. We can also identify the tilting point for precision to increase. For English and German this tilting point appears to lie at approximately the same threshold value. This will be shown in the next section, particularly in Figure 7.

3.6. Overlap of Similar Words in Different Vector Spaces

In this section, we explore whether we could use a combination of different vector spaces, trained using different training parameters to improve the synonym extraction. For this we analyze the most-cosine-similar words of the vocabulary $S_{\mathtt{train}}$ in different vector spaces. We considered pairs of vector

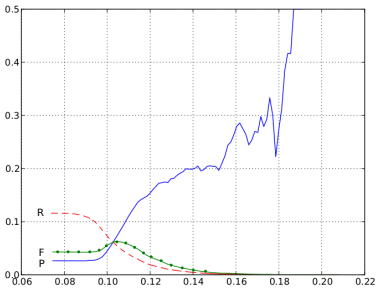

Figure 5. Precision, recall, and f-measure on $S_{\mathtt{train}}$ plotted against the relative cosine similarity threshold.

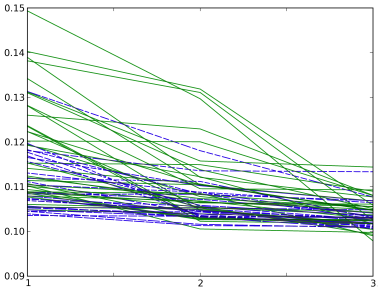

Figure 6. Relative cosine similarity against n-most similar position, for the 3-most-similar words, where the most-similar word is a synonym or a related word (dashed).

spaces with different training parameters. Then, we calculated the probability that an arbitrary word is most-cosine-similar in both vector spaces ($\mathrm{P}(\mathtt{both})$). We also calculated the probability that a synonym is most-cosine-similar in both vector spaces ($\mathrm{P}(\mathtt{both}|\mathtt{synonym})$). We altered the dimension, window size and model (CBoW vs. SG). We mostly considered CBoW vectors, as they gave highest precision in previous experiments. The results of this experiment are shown in Table 7. What can be seen in this table is that for all changes in

Constant	Varies	$\mathrm{P}(\mathtt{b})$	$\mathrm{P}(\mathtt{b}\|\mathtt{syn})$	$\mathrm{P}(\mathtt{b}\|\mathtt{syn}) - \mathrm{P}(\mathtt{b})$
CBoW win. 2	dim. 300 & 600	0.38	0.67	0.29
CBoW dim. 600	win. 2 & 4	0.31	0.60	0.30
CBoW dim. 600	win. 4 & 8	0.32	0.60	0.28
CBoW dim. 600	win. 2 & 8	0.24	0.52	0.28
dim. 300 win. 2	CBoW & SG	0.19	0.48	0.29

Table 7. Overlap between differently trained pairs of vector spaces, for arbitrary words, and synonyms. $\mathrm{P}(\mathtt{b})$ is the probability of a word pair being most-similar in both vector spaces, $\mathrm{P}(\mathtt{b}|\mathtt{syn})$ is conditioned on the word being synonym.

parameters $P(both|synonym)$ is considerably higher than $P(both)$. This indicates that it can be a good cue for synonymy if a word is most-cosine-similar in differently trained vector spaces. We can also see that the general overlap seems highest when only changing the number of dimensions, and lowest when changing the model, and fairly constant when doubling the window size. For all conditions, $P(both|synonym) - P(both)$ is fairly constant. This indicates that the cue for synonymy is almost equal for all pairs.

Because the numbers seem quite constant, it may be due to the inflections that overlap between both vector spaces. For this reason we repeated the experiment, but only considering word-pairs that have a Levenshtein distance greater than 3, to exclude the majority of the inflections. The results are shown in Table 8. Here we can see that the conclusion from Table 7 also holds for non-inflections. So, it is not just the inflections that overlap.

Constant	Varies	$P(b)$	$P(b|syn)$	$P(b|syn) - P(b)$
CBoW win. 2	dim. 300 & 600	0.31	0.61	0.30
CBoW dim. 600	win. 2 & 4	0.23	0.55	0.32
CBoW dim. 600	win. 4 & 8	0.24	0.56	0.32
CBoW dim. 600	win. 2 & 8	0.17	0.48	0.31
dim. 300 win. 2	CBoW & SG	0.12	0.42	0.30

Table 8. Overlap between differently trained pairs of vector spaces, for arbitrary words, and synonyms, when only considering word-pairs with a ***Levenshtein distance larger than 3.*** $P(b)$ *is the probability of a word pair being most-similar in both vector spaces,* $P(b|syn)$ *is conditioned on the word being synonym.*

To use this observations in our earlier synonym extraction method we calculate rcs_{10}^{m} in each vector space m for the 10 most-cosine-similar words on S_{train} in each space, and simply sum the rcs_{10} of the different models. The *summed relative cosine similarity* between word w_i and w_j is calculated in Equation 2, where $TOP_{10}^{m}(w_i)$ is the set containing the 10 closest cosine-similar words of w_i in vector space m.

$$rcs_{10}^{M} = \sum_{m}^{M} \begin{cases} rcs_{10}^{m}(w_i, w_j) & \text{if } w_j \in TOP_{10}^{m}(w_i) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

As in the previous section, we again plot precision, recall, and f-measure against the threshold, but now using the summed $\texttt{rcs}_{10}$ of a CBoW model, and a SG model. We did this for both German and English. For English, the CBoW model has 600 dimensions, and was trained with a window size of 4. The SG model has 150 dimensions, and a window size set to 2. For German, the CBoW model has 600 dimensions as well, and but a window size of 8. The results are shown in Figure 7. If we compare it to the results from Figure 5, we can

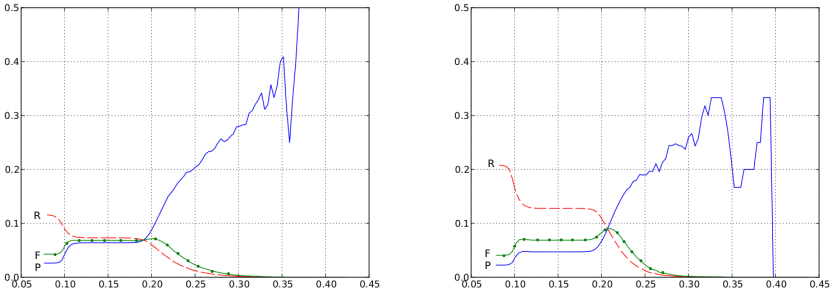

Figure 7. Precision, recall, and f-measure, on $S_{\texttt{train}}$ for English (left) and German (right), using the summed $\texttt{rcs}_{10}$ score for a CBoW and SG model.

see that for English, the general precision, recall, and f-measure lies higher using two vector spaces. Also, we can see that the tilting point now lies at around 0.2 instead of 0.1. It lies twice as high, as we sum $\texttt{rcs}_{10}$ of two spaces. Also, our expectation that for different languages this tilting point lies at the same threshold seems correct for German. The bump in both graphs around a threshold of 0.1 shows up because some words only occur in the top-10 most similar words in one of the two vector spaces.

When we choose the threshold that gives optimal f-measure on the $S_{\texttt{train}}$, and use it to extract synonyms for $S_{\texttt{test}}$, we find for English a WordNet precision of 0.12, a recall of 0.05, and an f-measure of 0.07. Compared to our baseline of only taking the most similar word, precision is 1% absolute higher, recall is 2% higher, and f-measure 1%. For German, we find a precision of 0.12, recall of 0.07, and f-measure of 0.09. Compared to the baseline, precision went up with 4% absolute, recall with 2%, and f-measure with 3%. From this, we conclude that combining differently trained models helps to extract syn-

onyms, both in precision, and recall. Also, combining the scores from the different vector spaces does not prevent us from finding the tilting point where precision rises.

4. Adding Parts-of-Speech

We now look at using a part-of-speech (POS) tagger to improve the synonym extraction in various ways.

4.1. Homography

The initial motivation to resort to POS-tagging is *homography*, i.e. one word (here, string of non-space characters) having several word-senses. In Figure 8, an example of homography of the words <phone> and <call> is given. The word senses and their respective parts of speech are shown in the leaves of the tree. The dotted link represents the synonym relation between the word-senses of <phone> and <call> for the action of making a telephone call.

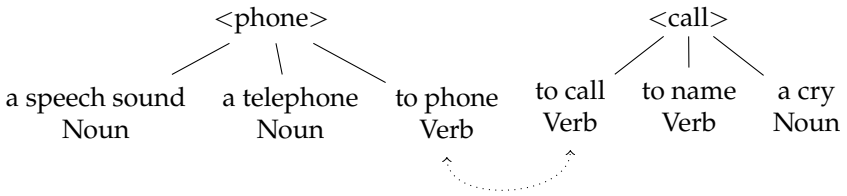

Figure 8. Schematic representation of the synonym relation between the corresponding word senses of the words <phone>, and <call>.

Homography can be a problem for finding synonyms when using one vector for each word, as the vector for <phone> is trained on all the different word-senses that occur in the corpus. In the case of <phone>, it is probably used more frequently as the noun telephone, or as a verb for the action of calling, compared to the noun meaning of a speech sound, in our news corpus. This can make it difficult to find synonyms with regard to this less frequent meaning.

To train vector representations for each word sense, ideally we would disambiguate each word in the corpus first, and then train the vectors on these disambiguated meanings. To our knowledge, there is not yet the possibility to do completely unsupervised word sense disambiguation. As can be seen

in the example in Figure 8, some of the word senses can be separated by their parts of speech. We experimented with this, since POS tagging is available for many languages, and there are also options for word clustering/unsupervised POS-tagging (Christodoulopoulos et al., 2010).

4.2. Simple Part-of-Speech Tagging

In order to separate some word senses we preprocessed both the English and German corpora from the previous chapter with the Stanford POS tagger (Toutanova et al., 2003), using the fastest tag-models. Afterwards, we conflated the POS tags to five categories: (1) nouns, (2) verbs, (3) adjectives, (4) adverbs, and (5) the rest (no tag). An example of what the text looks like after tagging and simplification is given in Sentence 1.

1. Every day_N , I walk_V my daily_Adj walk_N .

In the example we can see that walk_V is distinct from walk_N, which will give us two different vectors. We chose these four tags as they correspond to the POS tags provided in WordNet and GermaNet. In this way, we can have a straightforward way to evaluate on the vocabulary (e.g. S_{train}). For each word, we now evaluate with regard to the synonyms that have the same POS in WordNet or GermaNet.

Another advantage of having these simple POS tags is that we can filter bad synonyms from the 10-most cosine similar words. Synonyms are very similar also on a grammatical level, as they are interchangeable in many contexts, so they should be of the same part-of-speech.

Because the vocabulary has changed, and the average frequency of words is now lower—as some words are split—we again analyze what word vector training parameters work best. We train CBoW and Skip-gram vectors on the tagged corpus, varying the dimensions over {150, 300, 600}, and the contextual window size over {2, 4, 16, 32}. We calculate precision for the most-similar and second-most-similar word for all words in S_{train}. The results are shown in Tables 9, 10, 11, and 12.

CBoW (Tagged)															
dim.	150					300					600				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.079	0.080	0.073	0.067	0.060	0.084	**0.085***	0.080	0.074	0.066	0.084	0.084	0.081	0.073	0.069
P-2	0.058	0.056	0.053	0.049	0.045	0.061	0.061	0.059	0.055	0.050	0.061	0.062	0.059	0.055	0.053

*Table 9. Precision for different window sizes and number of dimensions, using the **CBoW** model, for POS-tagged **English**.*

Skip-gram (Tagged)															
dim.	150					300					600				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.068	0.065	0.057	0.049	0.045	0.069	0.066	0.057	0.052	0.046	0.067	0.062	0.052	0.046	0.041
P-2	0.050	0.047	0.041	0.038	0.036	0.050	0.047	0.042	0.038	0.035	0.050	0.045	0.038	0.034	0.031

*Table 10. Precision for different window sizes and number of dimensions, using the **Skip-gram** model, for POS-tagged **English**.*

CBoW (Tagged)															
dim.	150					300					600				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.086	0.092	0.094	0.092	0.090	0.092	0.100	0.100	0.099	0.094	0.090	0.102	**0.103***	0.101	0.101
P-2	0.060	0.065	0.066	0.065	0.063	0.065	0.069	0.072	0.070	0.069	0.064	0.070	0.072	0.071	0.071

*Table 11. Precision for different window sizes and number of dimensions, using the **CBoW** model, for POS-tagged **German**.*

Skip-gram (Tagged)															
dim.	150					300					600				
win.	2	4	8	16	32	2	4	8	16	32	2	4	8	16	32
P-1	0.084	0.085	0.086	0.082	0.080	0.085	0.085	0.083	0.077	0.077	0.082	0.079	0.072	0.066	0.065
P-2	0.059	0.061	0.061	0.059	0.058	0.061	0.063	0.059	0.057	0.056	0.058	0.059	0.053	0.049	0.047

*Table 12. Precision for different window sizes and number of dimensions, using the **Skip-gram** model, for POS-tagged **German**.*

If we look at Table 9 we can see that the highest precision is obtained using a CBoW model with a window size of 4, and 600 dimensions. If we compare this to the best results on the non-tagged corpus, from Table 2 in Section 3, the

optimal window size has stayed the same. Also CBoW vectors still perform better than Skip-gram vectors, and small windows work best for Skip-gram vectors. However, the best performing number of dimensions went from 600 to 300 when adding the POS-tag for English. A possible explanation can be that since the part-of-speech tags separate some of the word contexts, based on grammatical properties, the same information can be encoded with less dimensions.

For German, precision went up when adding the POS-tags. This can be seen if we compare the precision from Tables 4 and 5 with Tables 11 and 12. The best vectors are still CBoW vectors with 600 dimensions and a contextual window of 8. When we tried to find the reason why German has such a increase in precision compared to English, we found that it lies partially at the level of POS-tag simplification. As in the German part-of-speech tagset, the *Stuttgart-Tübingen tagset* (STTS), names are not considered as nouns. For this reason we did not conflate them to a noun tag, and they were excluded during evaluation. This was not the case for English. Names are one of the frequent error categories we found in Section 3.

This highlights another use of the POS tagger, which is that we can simply exclude categories for which we don't want to find synonyms, and maybe even filter bad synonym candidates from the 10-most-similar words. An example would be the frequent error category of plurals, but also other types of inflections, which can be filtered, as they are given a different POS tag (before tag conflation). These insights will be used in the final system, presented in Section 5.

To compare using the simplified POS tags with the previous approaches we also calculated precision, recall and f-measure on S_{test}. Compared to the baseline of looking only at the most-similar word, we found that recall in English increased from 3% to 4%, precision did not change (11%), and f-measure from 5% to 6%. Notably, German precision increased with 8% to 12%, recall from 5% to 7%, and f-measure from 6% to 9%.

From these experiments we conclude that POS tags can help to improve synonym extraction in three ways. Firstly, they can separate some of the word senses, however this effect is minor. Secondly, they can filter words that are not grammatically similar enough, such as plurals. And thirdly, they can exclude synonyms in categories for which there no, or very few, synonyms, such as names.

5. Final System and Evaluation

In this section we describe and evaluate the final systems for English and German that we constructed from the findings from the previous sections.

5.1. The System

For the final systems we used larger corpora than those used in the previous experiments. We used 500 million tokens from the same corpora as before, the English and German NewsCrawl 2014 corpora from the Workshop on Machine Translation in 2015. We POS tagged the corpora using the same parser and models as in Section 4. However, we do not simplify the POS tags, but instead use the fine-grained tags for nouns, verbs, adjectives or adverbs. We exclude the tags for names, as they have few to no synonyms.

It should be noted that in the German tagset there is only one tag for nouns, which covers both singular and plural nouns. This might result in more errors. For machine translation evaluation we do not expect this to have a large negative impact, as plurals would also have been matched by Meteor in the stemming module. However, it might result in a worse human evaluation.

For English we train CBoW vectors with 300 dimensions and a contextual window of 4. We also train Skip-gram vectors with 300 dimensions and a contextual window of 2. For German we train vectors with the same specifications, except for the German CBoW model we use a contextual window of size 8, and for Skip-gram a window of size 4. We chose these parameter settings as a compromise between the optimal parameters from our experiment in Chapter 4, and our expectations with respect to introducing fine-grained POS tags, which is that the optimal number of dimensions might decrease slightly.

We only consider words that occur at least 20 times in the corpus. The reasons for using a higher frequency threshold are (1) to obtain better quality word vectors, as we aim for high precision, and (2) to maintain a vocabulary size similar to the previous experiments, as we increased corpus size. The resulting tagged English vocabulary contains 115,632 word types, and the German vocabulary 311,664.

We then calculate the summed relative cosine similarity of both the CBoW and the Skip-gram vectors for the full vocabulary with regard to the top-10 most cosine-similar words. We select word pairs with a summed rcs_{10} similarity higher than 0.22. We choose 0.22 as it lies slightly above the expected tilting point of 0.2. For English, we obtain 16,068 word pairs. For German

we obtain 96,998 word pairs. It should be noted that the word pairs are also tagged, which can be useful depending on the application.

5.2. Manual Evaluation

To evaluate the precision of the obtained synonyms, we took a random sample of 200 word pairs for both languages. The word pairs were then annotated for synonymy. The annotation categories are synonyms, non-synonyms, or unknown. In the description the unknown category is indicated for when an annotator does not know any of the two words. The annotators could also indicate hesitation, but still had to give a preference for any of the three categories.

For English, annotation is done by two annotators. One annotator is a native English speaker and one a fluent non-native speaker. For German, annotation is also done by two annotators, one native German speaker, and one an intermediate non-native speaker. Annotators could access the internet to look up synonymy, or word meanings. We discriminate several situations:

SS: Both annotators annotate synonymy
NN: Both annotators annotate non-synonymy
SU: One annotator annotates synonymy, and the other unknown
NU: One annotator annotates non-synonymy, and the other unknown
SN: One annotator annotates synonymy, and the other non-synonymy
UU: Both annotators annotate unknown

We assume that if both annotators do not know the words, there is no synonymy. We can calculate a *lower bound of precision* ($P^{-}_{\mathtt{syn}}$), and an *upper bound of precision* ($P^{+}_{\mathtt{syn}}$). For the lower bound, we only consider word pairs of category SS as synonyms, and the rest as non-synonyms. For the upper bound, we consider word pairs of category SS and SU as synonyms, and the rest as non-synonyms.

We also calculate a lower and upper bound for non-synonymy ($P^{-}_{\neg\mathtt{syn}}$, and $P^{+}_{\neg\mathtt{syn}}$), and the percentage of disagreement on the categories of synonym and non-synonym ($P_{\mathtt{disagree}}$). This way we can get a better idea of how many clear errors there are, and how many errors are unclear.

The results for both English and German are shown in Table 13. What can be noticed is that for German, the precision is quite a bit lower than for English. However, the number of found word pairs is much higher. One reason can be that the threshold should be higher in order to get comparable precision. A second reason can be that for English the error categories, such as plurals, are separated by a POS tag, resulting in higher precision. In the German tagset these are not separated. We found that 10% of the German word pairs in this

Manual Evaluation	P^{-}_{syn}	P^{+}_{syn}	$P^{-}_{\neg syn}$	$P^{+}_{\neg syn}$	$P_{disagree}$	P_{UU}
English	0.55	0.59	0.15	0.21	0.16	0.05
German	0.30	0.35	0.42	0.49	0.15	0.03

Table 13. Manual evaluation of the final systems.

set are plurals. For English, there were no such cases. For our application, these errors should not be a major problem, as plurals would otherwise have been matched by the stemming module of Meteor.

The percentage of unknown words seems fairly small, and about the same for both languages. Also the disagreement on synonymy seems about the same for both languages, around 15%. The cause for disagreement could be the difference in the language level of the speakers. Another reason could be the subjectivity of the notion of synonymy.

5.3. Application in Machine Translation Evaluation

To see if the quality of the extracted synonyms is sufficient for the synonyms to be beneficial in an application we also used them in machine translation evaluation. We use them in the synonym module of the Meteor 1.5 evaluation metric.

We use the synonyms extracted by the system described in Section 5.1. So for German, the synonym resource will consist of the 96,998 word pairs, and for English we use 16,068 word pairs.

Meteor weighs the scores from each matching module. For English, we use the default weights (Denkowski and Lavie, 2014), as synonyms were already incorporated for English. For German, we use the default weights for all other modules, except we use the same weight for the synonym module as used for English (0.80).

To evaluate the metric, we test if the Meteor score correlates better with human judgments after adding our synonyms. We calculate the correlation using the data from the metrics task of the workshop on machine translation 2014[4] (WMT 2014) (Macháček and Bojar, 2014).

We use the news-test reference sentences from the language pair German-English, for English. This set consists of around 3000 segments, or sentences. For German, we use the reference sentences from the English-German language pair. This set consists of around 2700 segments, or sentences.

[4]http://www.statmt.org/wmt14/results.html

We calculate *segment-level Kendall's τ correlation* as calculated in the WMT 2014 for the following three Meteor conditions:

1. Using all four modules, with the default weights, and no synonym resource.
2. Using all four modules, using default weights, and with our synonyms.
3. Using all four modules, using default weights, using WordNet synonyms (only for English).

Kendall's τ is expected to predict the result of the pairwise comparison of two translation systems. In WMT-2014 this is calculated using human judgments on a ranking task of 5 systems per comparison. τ is calculated as in Equation 3, where *Concordant* is the set of human comparisons for which the Meteor score suggests the same order, and *Discordant* is the set of all human comparisons for which a given metric disagrees. When the Meteor score gives the same rankings as the human judgments, correlation will be high, and vice versa.

$$\tau = \frac{|\mathrm{Concordant}| - |\mathrm{Discordant}|}{|\mathrm{Concordant}| + |\mathrm{Discordant}|} \tag{3}$$

We calculated the Meteor scores for hypotheses from the 13 translation systems for the language pair German-English, and the 18 translation systems for English-German.

We also calculated the *system level correlation*, which indicates to what degree the evaluation metric orders the translation systems in the same order as the human judgments do, based on the total system score that the evaluation metric gives to each system. This is calculated as the *Pearson correlation*, as described by Macháček and Bojar (2014), and in Equation 4, where H is the vector of human scores of all systems translating in the given direction, M is the vector of the corresponding scores as predicted by the given metric, here Meteor. $\bar{H}$ and $\bar{M}$ are their means respectively.

$$r = \frac{\sum_{i=1}^{n}(H_i - \bar{H})(M_i - \bar{M})}{\sqrt{\sum_{i=1}^{n}(H_i - \bar{H})^2}\sqrt{\sum_{i=1}^{n}(M_i - \bar{M})^2}} \tag{4}$$

Both the segment-based correlations and the system-level correlations are shown in Table 14 for the same conditions as mentioned before. It can be seen that for both English and German using the extracted synonyms has a positive effect on both the segment correlation and the system correlation. It can also be noticed that using WordNet gives the highest correlation for English.

From this we conclude that currently our method, using only raw text and a POS tagger, does not outperform a large manually constructed synonym

German-**English**	τ	r
Condition 1	0.323	0.915
Condition 2	0.326	0.917
Condition 3	0.334	0.927

English-**German**	τ	r
Condition 1	0.238	0.263
Condition 2	0.243	0.277
Condition 3	-	-

Table 14. System level correlations (r), and segment level correlations (τ) for the Meteor 1.5 score without synonyms (condition 1), when adding the extracted synonyms (condition 2), and when using WordNet synonyms (condition 3).

database such as WordNet, but can be useful to extract synonyms when no such resource is available for the target language in Meteor[5].

What should be noted is that the extracted synonyms are not yet fully exploited, as Meteor ignores the POS tags that were given to the synonyms. If two words are synonymous with respect to their part of speech, but not synonymous if they are of different parts of speech, Meteor will align them in both situations. In the case when the words are of different POS, they will be falsely aligned by Meteor.

The improvement of the metric is greater for German than for English. This might seem odd at first, since the German synonyms had a lower precision in manual evaluation compared to the English synonyms. But still, they perform better in machine translation evaluation. This can be explained by what was already mentioned earlier, that a significant part of the German synonym errors are inflections, due to the difference in POS tagset. Also, the synonyms extracted for German are less ambiguous with respect to their part of speech. The German language frequently employs compounding (e.g. *Schwierigkeitsgrade*, 'degree of difficulty'), and grammatical case markers. This might result in less ambiguous words. The negative effect of Meteor not using parts of speech with synonyms could be smaller for German for this reason. Furthermore, the difference could also be explained by the difference in the number of synonyms (~16K for English, and ~97K for German).

6. Conclusions & Future Work

In this article we explored different methods to extract synonyms from text. The initial motivation was to use the extracted synonyms to improve machine translation evaluation. We tried to extract the synonyms using as little su-

[5]Our German results are an indirect example of this: even though a WordNet resource (GermaNet) exists, it is not available to Meteor due to licencing reasons.

pervision as possible, with the goal that the same method can be applied to multiple languages. We experimented with English and German.

Word vectors trained using the continuous bag-of-words model (CBoW), and the skip-gram model (SG) proposed by Mikolov et al. (2013a) were used in the experiments. We evaluated different hyperparameters for training these vectors for synonym extraction. In our experiments CBoW vectors gave higher precision and recall than SG vectors. The number of dimensions did not seem to play a very large role. For our experiments, dimensions of 300 and 600 seemed to give best results. The optimal contextual windows size was around 4 for English and 8 for German. We hypothesized that the difference in window size can be because of the difference in the distributions of word categories of the synonyms in WordNet and GermaNet.

For English, we manually looked at frequent error categories when using these vectors for this task. The largest well-defined error categories we found are inflections, co-hyponyms, and names.

We found that the cosine similarity on its own is a bad indicator to determine if two words are synonymous. We proposed *relative cosine similarity*, which calculates similarity relative to other cosine-similar words in the corpus. This is a better indicator, and can help improve precision. Also, the optimal thresholds for finding synonyms for English and German using this measure are almost the same. This gives hope for easy extension of this method to other languages, for which there is no synonym data. It would be very interesting to see to which other languages this method can generalize.

We also experimented with combining similarity scores from differently trained vectors, which seems to slightly increase both precision and recall. Furthermore, we explored the advantages of using a POS tagger as a way of introducing some light supervision. POS tags can help performance in different ways. Firstly, it can disambiguate some of the meanings of homographs. Secondly, it can help filter bad synonym candidates. And thirdly, it can prevent extraction of synonyms for word categories that have no, or very few synonyms, such as names. For future research, it would be interesting to examine the effect of using an unsupervised POS tagger (Christodoulopoulos et al., 2010).

We could also investigate the use of topical word embeddings (Liu et al., 2015a,b), or global context vectors (Huang et al., 2012). These techniques make different vectors for each word using topical information to disambiguate some of the different word senses.

We evaluated our final approach for both English and German. We did a manual evaluation with two annotators per language. We also applied the

extracted synonyms in machine translation evaluation. From the manual evaluation, the English synonyms had higher precision than the German ones. A likely reason for this is that the English POS tagset better separates the frequent error categories mentioned in Section 3.

When we evaluated the quality of the extracted synonyms in the task of machine translation evaluation (with the Meteor metric) for both English and German, the extracted synonyms increased the correlation of the metric with human judgments, resulting in an improved evaluation metric. While our method currently does not outperform a manually constructed synonym database such as WordNet, it can be useful to extract synonyms when no such resource is available for the target language, or domain. As the method uses tokenized raw text and optionally a POS tagger, it is applicable to a wide range of languages.

In the current research, we used a fixed frequency threshold, excluding infrequent words (a large part of the vocabulary). Setting a threshold also influences the word embedding training. For future research, it would be interesting to see the impact of the frequency threshold on our method.

Moreover, currently Meteor does not fully exploit the extracted synonyms, as it ignores their POS, which can cause false alignments. For future research on improving Meteor, it could be interesting to incorporate POS tags to prevent inappropriate generalization of synonyms.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

Bibliography

Agirre, Eneko, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Paşca, and Aitor 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*, pages 19–27, Boulder, CO, USA, June 2009. Association for Computational Linguistics.

Banerjee, Satanjeev and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, volume 29, pages 65–72, 2005.

Christodoulopoulos, Christos, Sharon Goldwater, and Mark Steedman. Two Decades of Unsupervised POS induction: How far have we come? In *Proceedings of the*

2010 Conference on Empirical Methods in Natural Language Processing, pages 575–584. Association for Computational Linguistics, 2010.

Comelles, Elisabet and Jordi Atserias. VERTa Participation in the WMT14 Metrics Task. In *Proceedings of the 9th Workshop on Statistical Machine Translation (WMT)*, pages 368–375, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

Denkowski, Michael and Alon Lavie. Meteor Universal: Language Specific Translation Evaluation for Any Target Language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*, 2014.

Doddington, George. Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. In *Proceedings of the 2nd International Conference on Human Language Technologies (HLT)*, pages 138–145, 2002.

Faruqui, Manaal, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. Retrofitting Word Vectors to Semantic Lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615, Denver, CO, USA, 2015. Association for Computational Linguistics.

Fu, Ruiji, Jiang Guo, Bing Qin, Wanxiang Che, Haifeng Wang, and Ting Liu. Learning semantic hierarchies via word embeddings. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics: Long Papers*, volume 1, 2014.

Gonzà lez, Meritxell, Alberto Barrón-Cedeño, and Lluís Màrquez. IPA and STOUT: Leveraging Linguistic and Source-based Features for Machine Translation Evaluation. In *Proceedings of the 9th Workshop on Statistical Machine Translation (WMT)*, pages 394–401, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

Gupta, Dishan, Jaime Carbonell, Anatole Gershman, Steve Klein, and David Miller. Unsupervised Phrasal Near-Synonym Generation from Text Corpora. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015a.

Gupta, Rohit, Constantin Orăsan, and Josef van Genabith. ReVal: A Simple and Effective Machine Translation Evaluation Metric Based on Recurrent Neural Networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Lisbon, Portugal, 2015b.

Harris, Zellig S. Distributional Structure. *Word*, 1954.

Huang, Eric H., Richard Socher, Christopher D. Manning, and Andrew Y. Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

Levy, Omer and Yoav Goldberg. Dependency-based word embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 2, pages 302–308, 2014.

Lin, Dekang, Shaojun Zhao, Lijuan Qin, and Ming Zhou. Identifying synonyms among distributionally similar words. In *IJCAI*, volume 3, pages 1492–1493, 2003.

Liu, Pengfei, Xipeng Qiu, and Xuanjing Huang. Learning Context-Sensitive Word Embeddings with Neural Tensor Skip-Gram Model. *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015a.

Liu, Yang, Zhiyuan Liu, Tat-Seng Chua, and Maosong Sun. Topical Word Embeddings. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015b.

Luong, Minh-Thang, Richard Socher, and Christopher D. Manning. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104, 2013.

Macháček, Matouš and Ondřej Bojar. Results of the WMT14 metrics shared task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 293–301, Baltimore, MD, USA, 2014.

Mikolov, Tomáš, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Mikolov, Tomáš, 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, 2013b.

Miller, George A. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

Ono, Masataka, Makoto Miwa, and Yutaka Sasaki. Word Embedding-based Antonym Detection using Thesauri and Distributional Information. In *Proc. of NAACL*, 2015.

Pak, Alexander Alexandrovich, Sergazy Sakenovich Narynov, Arman Serikuly Zharmagambetov, Sholpan Nazarovna Sagyndykova, Zhanat Elubaevna Kenzhebayeva, and Irbulat Turemuratovich. The Method of Synonyms Extraction from Unannotated Corpus. In *Digital Information, Networking, and Wireless Communications (DINWC), 2015 Third International Conference on*, pages 1–5. IEEE, 2015.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, 12, 2014.

Plas, Lonneke van der and Gosse Bouma. Syntactic contexts for finding semantically related words. *LOT Occasional Series*, 4:173–186, 2005.

Saveski, Martin and Igor Trajkovski. Automatic construction of wordnets by using machine translation and language modeling. In *13th Multiconference Information Society, Ljubljana, Slovenia*, 2010.

Snover, Matthew, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, 2006.

Stanojević, Miloš and Khalil Sima'an. BEER: BEtter Evaluation as Ranking. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 414–419, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

Tan, Liling, Rohit Gupta, and Josef van Genabith. USAAR-WLV: Hypernym Generation with Deep Neural Nets. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015.

Toutanova, Kristina, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 173–180. Association for Computational Linguistics, 2003.

Van der Plas, Lonneke and Jörg Tiedemann. Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 866–873. Association for Computational Linguistics, 2006.

van der Plas, Lonneke, Joerg Tiedemann, and Jean—Luc Manguin. Automatic acquisition of synonyms for French using parallel corpora. *Distributed Agent-based Retrieval Tools*, page 99, 2010.

Vela, Mihaela and Ekaterina Lapshinova-Koltunski. Register-Based Machine Translation Evaluation with Text Classification Techniques. In *Proceedings of the 15th Machine Translation Summit*, pages 215–228, Miami, Florida, November 2015. Association for Machine Translations in the Americas.

Vela, Mihaela and Liling Tan. Predicting Machine Translation Adequacy with Document Embeddings. In *Proceedings of the 10th Workshop on Statistical Machine Translation (WMT)*, pages 402–410, Lisbon, Portugal, September 2015. ACL.

Vossen, Piek. *A multilingual database with lexical semantic networks*. Kluwer Academic Publishers, Dordrecht, 1998.

Yu, Hong, Vasileios Hatzivassiloglou, Carol Friedman, Andrey Rzhetsky, and W John Wilbur. Automatic extraction of gene and protein synonyms from MEDLINE and journal articles. In *Proceedings of the AMIA Symposium*, page 919. American Medical Informatics Association, 2002.

Address for correspondence:
Artuur Leeuwenberg
tuur.leeuwenberg@cs.kuleuven.be
Katholieke Universiteit Leuven, Department of Computer Science
Celestijnenlaan 200A, 3000 Leuven, Belgium