Looking for Hyponyms in Vector Space

Marek Rei
SwiftKey
95 Southwark Bridge Rd
London, UK
marek@swiftkey.net

Ted Briscoe
Computer Laboratory
University of Cambridge
Cambridge, UK
ted.briscoe@cl.cam.ac.uk

Abstract

The task of detecting and generating hyponyms is at the core of semantic understanding of language, and has numerous practical applications. We investigate how neural network embeddings perform on this task, compared to dependency-based vector space models, and evaluate a range of similarity measures on hyponym generation. A new asymmetric similarity measure and a combination approach are described, both of which significantly improve precision. We release three new datasets of lexical vector representations trained on the BNC and our evaluation dataset for hyponym generation.

1 Introduction

Hyponymy is a relation between two word senses, indicating that the meaning of one word is also contained in the other. It can be thought of as a *type-of* relation; for example car, $ship$ and $train$ are all hyponyms of $vehicle$. We denote a hyponymy relation between words a and b as $(a \rightarrow b)$, showing that a is a hyponym of b, and b is a hypernym of a. Hyponymy relations are closely related to the concept of entailment, and this notation is consistent with indicating the direction of inference – if a is true, b must be true as well.

Automatic detection and generation of hyponyms has many practical applications in nearly all natural language processing tasks. Information retrieval, information extraction and question answering can be improved by performing appropriate query expansions. For example, a user searching for *arthritis treatment* is most likely also interested in results containing the hyponyms of *treatment*, such as *arthritis therapy*, *arthritis medication*, and *arthritis rehabilitation*. Summarisation systems can increase coherence and reduce repetition by correctly handling hyponymous words in the input text. Entailment and inference systems can improve sentence-level entailment resolution by detecting the presence and direction of word-level hyponymy relations. Distributionally similar words have been used for smoothing language models and word co-occurrence probabilities (Dagan et al., 1999; Weeds and Weir, 2005), and hyponyms can be more suitable for this application.

We distinguish between three different tasks related to hyponyms. Given a directional word pair, the goal of **hyponym detection** is to determine whether one word is a hyponym of the other (Zhitomirsky-Geffet and Dagan, 2009; Kotlerman et al., 2010; Baroni and Lenci, 2011). In contrast, **hyponym acquisition** is the task of extracting all possible hyponym relations from a given text (Hearst, 1992; Caraballo, 1999; Pantel and Ravichandran, 2004; Snow et al., 2005). Such systems often make use of heuristic rules and patterns for extracting relations from surface text, and populate a database with hyponymous word pairs. Finally, the task of **hyponym generation** is to return a list of all possible hyponyms, given only a single word as input. This is most relevant to practical applications, as many systems require a set of appropriate substitutes for a specific term. Automated ontology creation (Biemann, 2005) is a related field that also makes use of distributional similarity measures. However, it is mostly focused on building prototype-based ontologies through clustering (Ushioda, 1996; Bisson et al., 2000; Wagner, 2000; Paaß et al., 2004; Cimiano and Staab, 2005), and is not directly applicable to hyponym generation.

While most work has been done on hyponym detection (and the related task of lexical substitution), barely any evaluation has been done for hyponym generation. We have found that systems for hyponym detection often perform poorly on hyponym generation, as the latter requires returning results from a much less restricted candidate set,

Proceedings of the Eighteenth Conference on Computational Language Learning, pages 68–77,
Baltimore, Maryland USA, June 26-27 2014. ©2014 Association for Computational Linguistics

and therefore a task-specific evaluation is required.

In this paper we focus on hyponym generation and approach it by scoring a very large candidate set of potential hyponyms. Distributional similarity methods are especially interesting for this task, as they can be easily applied to different domains, genres and languages without requiring annotated training data or manual pattern construction. We perform a systematic comparison of different vector space models and similarity measures, in order to better understand the properties of a successful method for hyponym generation.

The main contributions of this paper are:

1. Systematic evaluation of different vector space models and similarity measures on the task of hyponym generation.
2. Proposal of new properties for modelling the directional hyponymy relation.
3. Release of three lexical vector datasets, trained using neural network, window-based, and dependency-based features.

2 Vector space models

In order to use similarity measures for hyponym detection, every word needs to be mapped to a point in vector space. The method of choosing appropriate features for these vectors is crucial to achieving the optimal performance. We compare five different approaches:

Window: As a simple baseline, we created vectors by counting word co-occurrences in a fixed context window. Every word that occurs within a window of three words before or after is counted as a feature for the target word. Pointwise mutual information is then used for weighting.

CW: Collobert and Weston (2008) constructed a neural network language model that is trained to predict the next word in the sequence, and simultaneously learns vector representations for each word. The vectors for context words are concatenated and used as input for the neural network, which uses a sample of possible outputs for gradient calculation to speed up the training process. Turian et al. (2010) recreated their experiments and made the vectors available online.[1]

HLBL: Mnih and Hinton (2007) created word representations using the hierarchical log-bilinear model – a neural network that takes the concatenated vectors of context words as input, and is trained to predict the vector representation of the next word, which is then transformed into a probability distribution over possible words. To speed up training and testing, they use a hierarchical data structure for filtering down the list of candidates. Both CW and HLBL vectors were trained using 37M words from RCV1.

Word2vec: We created word representations using the word2vec[2] toolkit. The tool is based on a feedforward neural network language model, with modifications to make representation learning more efficient (Mikolov et al., 2013a). We make use of the skip-gram model, which takes each word in a sequence as an input to a log-linear classifier with a continuous projection layer, and predicts words within a certain range before and after the input word. The window size was set to 5 and vectors were trained with both 100 and 500 dimensions.

Dependencies: Finally, we created vector representations for words by using dependency relations from a parser as features. Every incoming and outgoing dependency relation is counted as a feature, together with the connected term. For example, given the dependency relation *(play, dobj, guitar)*, the tuple *(>dobj, guitar)* is extracted as a feature for *play*, and *(<dobj, play)* as a feature for *guitar*. We use only features that occur more than once in the dataset, and weight them using pointwise mutual information to construct feature vectors for every term. Features with negative weights were retained, as they proved to be beneficial for some similarity measures.

The window-based, dependency-based and word2vec vector sets were all trained on 112M words from the British National Corpus, with preprocessing steps for lowercasing and lemmatising. Any numbers were grouped and substituted by more generic tokens. For constructing the dependency-based vector representations, we used the parsed version of the BNC created by Andersen et al. (2008) with the RASP toolkit (Briscoe et al., 2006). When saved as plain text, the 500-dimensional word2vec vectors and dependency-based vectors are comparable in size (602MB and 549MB), whereas the window-based vectors are twice as large (1,004MB). We make these vector

[1] http://metaoptimize.com/projects/wordreprs/

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

sets publically available for download.[3]

Recently, Mikolov et al. (2013b) published interesting results about linguistic regularities in vector space models. They proposed that the relationship between two words can be characterised by their **vector offset**, for example, we could find the vector for word “queen” by performing the operation “king - man + woman” on corresponding vectors. They also applied this approach to hyponym relations such as ($shirt \rightarrow clothing$) and ($bowl \rightarrow dish$). We evaluate how well this method applies to hyponym generation with each of the vector space models mentioned above. Using the training data, we learn a vector for the hyponymy relation by averaging over all the offset vectors for hyponym-hypernym pairs. This vector is then added to the hypernym during query time, and the result is compared to hyponym candidates using cosine similarity. For sparse high-dimensional vector space models it was not feasible to use the full offset vector during experiments, therefore we retain only the top 1,000 highest-weighted features.

3 Similarity measures

We compare the performance of a range of similarity measures, both directional and symmetrical, on the task of hyponym generation.

Cosine similarity is defined as the angle between two feature vectors and has become a standard measure of similarity between weighted vectors in information retrieval (IR).

Lin similarity, created by Lin (1998), uses the ratio of shared feature weights compared to all feature weights. It measures the weighted proportion of features that are shared by both words.

DiceGen2 is one possible method for generalising the Dice measure to real-valued weights (Curran, 2003; Grefenstette, 1994). The dot product of the weight vectors is normalised by the total sum of all weights. The same formula can also be considered as a possible generalisation for the Jaccard measure.

WeedsPrec and **WeedsRec** were proposed by Weeds et al. (2004) who suggested using precision and recall as directional measures of word similarity. In this framework, the features are treated similarly to retrieved documents in information retrieval – the vector of the broader term b is used as the gold standard, and the vector of the narrower term a is in the role of retrieval results. Precision is then calculated by comparing the intersection (*items correctly returned*) to the values of the narrower term only (*all items returned*). In contrast, WeedsRec quantifies how well the features of the broader term are covered by the narrower term.

Balprec is a measure created by Szpektor and Dagan (2008). They proposed combining WeedsPrec together with the Lin measure by taking their geometric average. This aims to balance the WeedsPrec score, as the Lin measure will penalise cases where one vector contains very few features.

ClarkeDE, proposed by Clarke (2009), is an asymmetric *degree of entailment* measure, based on the concept of distributional generality (Weeds et al., 2004). It quantifies the weighted coverage of the features of the narrower term a by the features of the broader term b.

BalAPInc, a measure described by Kotlerman et al. (2010), combines the APInc score with Lin similarity by taking their geometric average. The APInc measure finds the proportion of shared features relative to the features for the narrower term, but this can lead to unreliable results when the number of features is very small. The motivation behind combining these measures is that the symmetric Lin measure will decrease the final score for such word pairs, thereby balancing the results.

4 Properties of a directional measure

Finding similar words in a vector space, given a symmetric similarity measure, is a relatively straightforward task. However finding hyponyms is arguably more difficult, as the relation is asymmetric, and looking at the distance or angle between the two words may not be enough.

Kotlerman et al. (2010) investigate the related problem of detecting directional lexical entailment, and they propose three desirable properties that a directional distributional similarity measure should capture:

1. The relevance of the shared features to the narrower term.

2. The relevance of the shared features to the broader term.

3. That relevance is less reliable if the number of features of either the narrower or the broader term is small.

[3] http://www.marekrei.com/projects/vectorsets/

Given a term pair ($a \to b$) we refer to a as the narrower term and b as the broader term. The features of a that are also found in b (have non-zero weights for both a and b) are referred to as *shared features*.

They show that existing measures which correspond to these criteria perform better and construct the BalAPInc measure based on the principles. However, it is interesting to note that these properties do not explicitly specify any directional aspects of the measure, and symmetric similarity scores can also fulfil the requirements.

Based on investigating hyponym distributions in our training data, we suggest two additions to this list of desired properties, one of which specifically targets the asymmetric properties of the desired similarity measures:

4. The shared features are more important to the directional score calculation, compared to non-shared features.
5. Highly weighted features of the broader term are more important to the score calculation, compared to features of the narrower term.

Most existing directional similarity scores measure how many features of the narrower term are present for the broader term. If a entails b, then it is assumed that the possible contexts of a are a subset of contexts for b, but b occurs in a wider range of contexts compared to a. This intuition is used by directional measures such as $ClarkeDE$, $WeedsPrec$ and $BalAPInc$. In contrast, we found that many features of the narrower term are often highly specific to that term and do not generalise even to hypernyms. Since these features have a very high weight for the narrower term, their absence with the broader term will have a big negative impact on the similarity score.

We hypothesise that many terms have certain individual features that are common to them but not to other related words. Since most weighting schemes reward high relative co-occurrence, these features are also likely to receive high weights. Therefore, we suggest that features which are not found for both terms should have a decreased impact on the score calculation, as many of them are not expected to be shared between hyponyms and hypernyms. However, removing them completely is also not advisable, as they allow the measure to estimate the overall relative importance of the shared features to the specific term.

We also propose that among the shared features, those ranked higher for the broader term are more important to the directional measure. In the hyponymy relation ($a \to b$), the term b is more general and covers a wider range of semantic concepts. This also means it is more likely to be used in contexts that apply to different hyponyms of b. For example, some of the high-ranking features for *food* are *blandly-flavoured*, *high-calorie* and *uneaten*. These are properties that co-occur often with the term $food$, but can also be applied to most hyponyms of *food*. Therefore, we hypothesise that the presence of these features for the narrower term is a good indication of a hyponymy relation. This is somewhat in contrast to most previous work, where the weights of the narrower term have been used as the main guideline for similarity calculation.

5 Weighted cosine

We now aim to construct a similarity measure that follows all five of the properties mentioned above. Cosine similarity is one of the symmetric similarity measures which corresponds to the first three desired properties, and our experiments showed that it performs remarkably well at the task of hyponym generation. Therefore, we decided to modify cosine similarity to also reflect the final two properties and produce a more appropriate asymmetric score.

The standard feature vectors for each word contain weights indicating how important this feature is to the word. We specify additional weights that measure how important the feature is to that specific directional relation between the two terms. Weighted cosine similarity, shown in Table 1, can then be used to calculate a modified similarity score. F_a denotes the set of weighted features for word a, $w_a(f)$ is the weight of feature f for word a, and $z(f)$ is the additional weight for feature f, given the directional word pair (a, b).

Based on the new desired properties we want to downweight the importance of features that are not present for both terms. For this, we choose the simple solution of scaling them with a small constant $C \in [0, 1]$. Next, we also want to assign higher $z(f)$ values to the shared features that have high weights for the broader term b. We use the relative rank of feature f in F_b, $r_b(f)$, as the indicator of its importance and scale this value to the range from C to 1. This results in the importance

$$WeightedCosine(F_a, F_b) = \frac{\sum_{f \in F_a \cap F_b} (z(f) \times w_a(f)) \times (z(f) \times w_b(f))}{\sqrt{\sum_{f \in F_a} (z(f) \times w_a(f))^2} \times \sqrt{\sum_{f \in F_b} (z(f) \times w_b(f))^2}}$$

$$z(f) = \begin{cases} (1 - \frac{r_b(f)}{|F_b|+1}) \times (1 - C) + C & \text{if } f \in F_a \cap F_b \\ C & \text{otherwise} \end{cases}$$

Table 1: Weighted cosine similarity measure

function decreasing linearly as the rank number increases, but the weights for the shared features always remain higher compared to the non-shared features. Tied feature values are handled by assigning them the average rank value. Adding 1 to the denominator of the relative rank calculation avoids exceptions with empty vectors, and also ensures that the value will always be strictly greater than C. While the basic function is still the symmetric cosine, the $z(f)$ values will be different depending on the order of the arguments.

The parameter C controls the relative importance of the 'unimportant' features to the directional relation. Setting it to 0 will ignore these features completely, while setting it to 1 will result in the traditional cosine measure. Experiments on the development data showed that the exact value of this parameter is not very important, as long as it is not too close to the extreme values of 0 or 1. We use the value $C = 0.5$ for reporting our results, meaning that the non-shared features are half as important, compared to the shared features.

6 Dataset

As WordNet (Miller, 1995) contains numerous manually annotated hyponymy relations, we can use it to construct suitable datasets for evaluating hyponym generation. While WordNet terms are annotated with only the closest hyponyms, we are considering all indirect/inherited hyponyms to be relevant – for example, given relations ($genomics \rightarrow genetics$) and ($genetics \rightarrow biology$), then $genomics$ is also regarded as a hyponym of $biology$. WordNet relations are defined between synsets, but we refrain from the task of word sense disambiguation and count word a as a valid hyponym for word b if it is valid for any sense of b.

Synonymy can be thought of as a symmetric *is-a* relation, and most real-world applications would require synonyms to also be returned, together with hyponyms. Therefore, in our dataset we consider synonyms as hyponyms in both directions. We also performed experiments without synonyms and found that this had limited effect on the results – while the accuracy of all similarity measures slightly decreased (due to fewer numbers of correct answers), the relative ranking remained the same. As shown in the next section, the number of synonyms is typically small compared to the number of all inherited hyponyms.

To construct the dataset, we first found all single-word nouns in WordNet that are contained at least 10 times in the British National Corpus (BNC). Next, we retained only words that have at least 10 hyponyms, such that they occur 10 or more times in the BNC. This selection process aims to discard WordNet hypernyms that are very rare in practical use, and would not have enough examples for learning informative vector representations. The final dataset contains the remaining terms, together with all of their hyponyms, including the rare/unseen hyponyms. As expected, some general terms, such as *group* or *location*, have a large number of inherited hyponyms. On average, each hypernym in the dataset has 233 hyponyms, but the distribution is roughly exponential, and the median is only 36.

In order to better facilitate future experiments with supervised methods, such as described by Baroni et al. (2012), we randomly separated the data into training (1230 hypernyms), validation (922), and test (922) sets, and we make these datasets publically available online.[4]

7 Experiments

We evaluate how well different vector space models and similarity measures perform on the task of hyponym generation. Given a single word as input, the system needs to return a ranked list of words with correct hyponyms at the top. As the list of candidates for scoring we use all words in the BNC that occur at least 10 times (a total of 86,496 words). All the experiments are performed using tokenised and lemmatised words.

As the main evaluation measure, we report

[4] http://www.marekrei.com/projects/hypgen/

	Cosine			Cosine+offset		
	MAP	**P@1**	**P@5**	**MAP**	**P@1**	**P@5**
Window	2.18	19.76	12.20	2.19	19.76	12.25
CW-100	0.66	3.80	3.21	0.59	3.91	2.89
HLBL-100	1.01	10.31	6.04	1.01	10.31	6.06
Word2vec-100	1.78	15.96	10.12	1.50	12.38	8.71
Word2vec-500	2.06	19.76	11.92	1.77	17.05	10.71
Dependencies	**2.73**	**25.41**	**14.90**	**2.73**	**25.52**	**14.92**

Table 2: Experiments using different vector space models for hyponym generation on the test set. We report results using regular cosine similarity and the vector offset method described in Section 2.

Mean Average Precision (MAP), which averages precision values at various recall points in the returned list. It combines both precision and recall, as well as the quality of the ranking, into a single measure, and is therefore well-suited for comparing different methods. The reported MAP values are very low – this is due to many rare WordNet hyponyms not occurring in the candidate set, for which all systems are automatically penalised. However, this allows us to evaluate recall, making the results comparable between different systems and background datasets. We also report precision at top-1 and top-5 returned hyponyms.

As a baseline we report the results of a traditional hyponym acquisition system. For this, we implemented the pattern-based matching process described by Hearst (1992), and also used by Snow et al. (2005). These patterns look for explicit examples of hyponym relations mentioned in the text, for example:

$$X\ such\ as\ \{Y_1,\ Y_2,\ ...\ ,\ (and|or)\}\ Y_n$$

where X will be extracted as the hypernym, and Y_1 to Y_n as hyponyms. We ran the patterns over the BNC and extracted 21,704 hyponym pairs, which were then ranked according to the number of times they were found.

7.1 Evaluation of vector spaces

Table 2 contains experiments with different vector space models. We report here results using cosine, as it is an established measure and a competitive baseline. For our task, the HLBL vectors perform better than CW vectors, even though they were trained on the same data. Both of them are outperformed by word2vec-100 vectors, which have the same dimensionality but are trained on much more text. Increasing the dimensionality with word2vec-500 gives a further improvement. Interestingly, the simple window-based vectors perform just as well as the ones trained with neural networks. However, the advantage of word2vec-500 is that the representations are more compact and require only about half the space. Finally, the dependency-based vectors outperform all other vector types, giving 2.73% MAP and 25.41% precision at the top-ranked result. While the other models are built by using neighbouring words as context, this model looks at dependency relations, thereby taking both semantic and syntactic roles into account. The results indicate that word2vec and window-based models are more suitable when the general topic of words needs to be captured, whereas dependency-based vectors are preferred when the task requires both topical and functional similarity between words. Our experiments also included the evaluation of other similarity measures on different vector space models, and we we found these results to be representative.

Contrary to previous work, the vector offset method, described in Section 2, did not provide substantial improvements on the hyponym generation task. For the neural network-based vectors this approach generally decreased performance, compared to using direct cosine similarity. There are some marginal improvements for window and dependency-based models. Unfortunately, the original work did not include baseline performance using cosine similarity, without applying vector modifications. It is possible that this method does not generalise to all word relations equally well. As part of future work, it is worth exploring if a hypernym-specific strategy of selecting training examples could improve the performance.

	Validation			Test		
	MAP	**P@1**	**P@5**	**MAP**	**P@1**	**P@5**
Pattern-based	0.53	7.06	4.58	0.51	8.14	4.45
Cosine	2.48	21.06	12.96	2.73	25.41	14.90
Lin	1.87	16.50	10.75	2.01	21.17	12.23
DiceGen2	2.27	18.57	12.62	2.44	21.82	14.55
WeedsPrec	0.13	0.00	0.09	0.12	0.11	0.04
WeedsRec	0.72	0.33	2.45	0.69	0.54	2.41
BalPrec	1.78	15.31	10.55	1.88	17.48	11.34
ClarkeDE	0.23	0.00	0.02	0.24	0.00	0.09
BalAPInc	1.64	14.22	9.12	1.68	15.85	9.66
WeightedCosine	2.59	21.39	13.59	2.85	25.84	15.46
Combined	**3.27**	**23.02**	**16.09**	**3.51**	**27.69**	**18.02**

Table 3: Evaluation of different vector similarity measures on the validation and test set of hyponym generation. We report Mean Average Precision (MAP), precision at rank 1 (P@1), and precision at rank 5 (P@5).

7.2 Evaluation of similarity measures

Table 3 contains experiments with different similarity measures, using the dependency-based model, and Table 4 contains sample output from the best system. The results show that the pattern-based baseline does rather poorly on this task. MAP is low due to the system having very limited recall, but higher precision at top ranks would have been expected. Analysis showed that this system was unable to find any hyponyms for more than half (513/922) of the hypernyms in the validation set, leading to such poor recall that it also affects Precision@1. While the pattern-based system did extract a relatively large number of hyponyms from the corpus (21,704 pairs), these are largely concentrated on a small number of hypernyms (e.g., *area*, *company*, *material*, *country*) that are more likely to be mentioned in matching contexts.

Cosine, DiceGen2 and Lin – all symmetric similarity measures – perform relatively well on this task, whereas established directional measures perform unexpectedly poorly. This can perhaps be explained by considering the distribution of hyponyms. Given a word, the most likely candidates for a high cosine similarity are synonyms, antonyms, hypernyms and hyponyms of that word – these are words that are likely to be used in similar topics, contexts, and syntactic roles. By definition, there are an equal number of hyponym and hypernym relations in WordNet, but this ratio changes rapidly as we remove lower-frequency words. Figure 1 shows the number of relations extracted from WordNet, as we restrict the minimum frequency of the main word. It can be seen that the number of hyponyms increases much faster compared to the other three relations. This also applies to real-world data – when averaging over word instances found in the BNC, hyponyms cover 85% of these relations. Therefore, the high performance of cosine can be explained by distributionally similar words having a relatively high likelihood of being hyponyms.

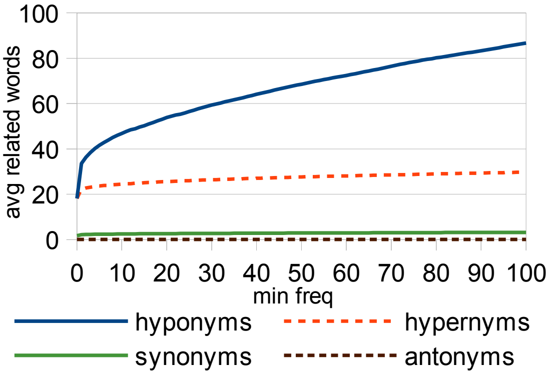

Figure 1: Average number of different relations per word in WordNet, as we restrict the minimum word frequency.

One possible reason for the poor performance of directional measures is that most of them quantify how well the features of the narrower term are included in the broader term. In contrast, we found that for hyponym generation it is more important to measure how well the features of the broader term are included in the narrower term. This

scientist	**researcher**, **biologist**, **psychologist**, **economist**, observer, **physicist**, **sociologist**
sport	**football**, **golf**, club, **tennis**, **athletics**, **rugby**, **cricket**, game, recreation, entertainment
treatment	**therapy**, **medication**, patient, procedure, surgery, remedy, regimen, medicine

Table 4: Examples of top results using the combined system. WordNet hyponyms are marked in bold.

is supported by WeedsRec outperforming WeedsPrec, although the opposite was intended by their design.

Another explanation for the low performance is that these directional measures are often developed in an artificial context. For example, Kotlerman et al. (2010) evaluated lexical entailment detection on a dataset where the symmetric Lin similarity measure was used to select word pairs for manual annotation. This creates a different task, as correct terms that do not have a high symmetric similarity will be excluded from evaluation. The BalAPInc measure performed best in that setting, but does not do as well for hyponym generation, where candidates are filtered only based on minimum frequency.

The weighted cosine measure, proposed in Section 5, outperformed all other similarity measures on both hyponym generation datasets. The improvement over cosine is relatively small; however, it is consistent and the improvement in MAP is statistically significant on both datasets ($p < 0.05$), using the Approximate Randomisation Test (Noreen, 1989; Cohen, 1995) with 10^6 iterations. This further supports the properties of a directional similarity measure described in Section 4.

Finally, we created a new system by combining together two separate approaches: the weighted cosine measure using the dependency-based vector space, and the normal cosine similarity using word2vec-500 vectors. We found that the former is good at modelling the grammatical roles and directional containment, whereas the latter can provide useful information about the topic and semantics of the word. Turney (2012) also demonstrated the importance of both topical (domain) and functional vector space models when working with semantic relations. We combined these approaches by calculating both scores for each word pair and taking their geometric average, or 0 if it could not be calculated. This final system gives considerable improvements across all evaluation metrics, and is significantly ($p < 0.05$) better compared to cosine or weighted cosine methods individually. Table 4 contains some example output from this system.

8 Conclusion

Hyponym generation has a wide range of possible applications in NLP, such as query expansion, entailment detection, and language model smoothing. Pattern-based hyponym acquisition can be used to find relevant hyponyms, but these approaches rely on both words being mentioned together in a specific context, leading to very low recall. Vector similarity methods are interesting for this task, as they can be easily applied to different domains and languages without any supervised learning or manual pattern construction. We created a dataset for evaluating hyponym generation systems and experimented with a range of vector space models and similarity measures.

Our results show that choosing an appropriate vector space model is equally important to using a suitable similarity measure. We achieved the highest performance using dependency-based vector representations, which outperformed neural network and window-based models. Symmetric similarity measures, especially cosine similarity, performed surprisingly well on this task. This can be attributed to an unbalanced distribution of hyponyms, compared to other high-similarity words. The choice of vector space can be highly dependent on the specific task, and we have made available our vector datasets created from the same source using three different methods.

We proposed two new properties for detecting hyponyms, and used them to construct a new directional similarity measure. This weighted cosine measure significantly outperformed all others, showing that a theoretically-motivated directional measure is still the most accurate method for modelling hyponymy relations. Finally, we combined together two different methods, achieving further substantial improvements on all evaluation metrics.

References

Øistein E. Andersen, Julien Nioche, Edward J. Briscoe, and John Carroll. 2008. The BNC parsed with RASP4UIMA. In *Proceedings of the Sixth Interna-*

tional Language Resources and Evaluation Conference (LREC08), Marrakech, Morocco.

Marco Baroni and Alessandro Lenci. 2011. How we BLESSed distributional semantic evaluation. In *Proceedings of the GEMS 2011 Workshop on GEometrical Models of Natural Language Semantics*, Edinburgh.

Marco Baroni, Raffaella Bernardi, Ngoc-Quynh Do, and Chung-chieh Shan. 2012. Entailment above the word level in distributional semantics. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 23–32.

Chris Biemann. 2005. Ontology learning from text: A survey of methods. *LDV Forum*, 20(2002):75–93.

Gilles Bisson, Claire Nédellec, and Dolores Cañamero. 2000. Designing clustering methods for ontology building-The Mo'K workbench. In *ECAI Ontology Learning Workshop*.

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, number July, pages 77–80, Sydney, Australia. Association for Computational Linguistics.

Sharon A. Caraballo. 1999. Automatic construction of a hypernym-labeled noun hierarchy from text. *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 120–126.

Philipp Cimiano and Steffen Staab. 2005. Learning concept hierarchies from text with a guided hierarchical clustering algorithm. In *ICML-Workshop on Learning and Extending Lexical Ontologies by using Machine Learning Methods*.

Daoud Clarke. 2009. Context-theoretic semantics for natural language: an overview. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, number March, pages 112–119. Association for Computational Linguistics.

Paul R Cohen. 1995. *Empirical Methods for Artificial Intelligence*. The MIT Press, Cambridge, MA.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning*.

James R. Curran. 2003. *From distributional to semantic similarity*. Ph.D. thesis, University of Edinburgh.

Ido Dagan, Lillian Lee, and Fernando C. N. Pereira. 1999. Similarity-based models of word cooccurrence probabilities. *Machine Learning*, 31:1–31.

Gregory Grefenstette. 1994. *Explorations in Automatic Thesaurus Discovery*. Kluwer Academic Publishers, Norwell, MA, USA.

Marti A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th conference on Computational linguistics (COLING '92)*, number July, page 539, Morristown, NJ, USA. Association for Computational Linguistics.

Lili Kotlerman, Ido Dagan, Idan Szpektor, and Maayan Zhitomirsky-Geffet. 2010. Directional distributional similarity for lexical inference. *Natural Language Engineering*, 16(04):359–389.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pages 768–774. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *ICLR Workshop*, pages 1–12.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013b. Linguistic Regularities in Continuous Space Word Representations. (June):746–751.

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

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. *Proceedings of the 24th international conference on Machine learning - ICML '07*, pages 641–648.

Eric W. Noreen. 1989. *Computer Intensive Methods for Testing Hypotheses: An Introduction*. Wiley, New York.

Gerhard Paaß, Jörg Kindermann, and Edda Leopold. 2004. Learning prototype ontologies by hierachical latent semantic analysis.

Patrick Pantel and Deepak Ravichandran. 2004. Automatically labeling semantic classes. In *Proceedings of HLT/NAACL*.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems*.

Idan Szpektor and Ido Dagan. 2008. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING '08)*, pages 849–856, Morristown, NJ, USA. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Y Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.*

Peter D. Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research*, 44:533–585.

Akira Ushioda. 1996. Hierarchical clustering of words and application to NLP tasks. In *Fourth Workshop on Very Large Corpora*, pages 28–41.

Andreas Wagner. 2000. Enriching a lexical semantic net with selectional preferences by means of statistical corpus analysis. In *ECAI Workshop on Ontology Learning*.

Julie Weeds and David Weir. 2005. Co-occurrence retrieval: A flexible framework for lexical distributional similarity. *Computational Linguistics*.

Julie Weeds, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. *Proceedings of the 20th international conference on Computational Linguistics - COLING '04*.

Maayan Zhitomirsky-Geffet and Ido Dagan. 2009. Bootstrapping Distributional Feature Vector Quality. *Computational Linguistics*, 35(3):435–461, September.