Automatic Metaphor Interpretation Using Word Embeddings

Kfir Bar, Nachum Dershowitz, Lena Dankin
School of Computer Science, Tel Aviv University

Abstract

We suggest a model for metaphor interpretation using word embeddings trained over a relatively large corpus. Our system handles nominal metaphors, like *time is money*. It generates a ranked list of potential interpretations of given metaphors. Candidate meanings are drawn from collocations of the topic (*time*) and vehicle (*money*) components, automatically extracted from a dependency-parsed corpus. We explore adding candidates derived from word association norms (common human responses to cues). Our ranking procedure considers similarity between candidate interpretations and metaphor components, measured in a semantic vector space. Lastly, a clustering algorithm removes semantically related duplicates, thereby allowing other candidate interpretations to attain higher rank. We evaluate using a set of annotated metaphors.

1 Introduction

Metaphor is pervasive in language and thought [2]. A specific metaphor often has an ambiguous interpretation. For example, when we say *memory is a river*, both *fluid* and *long* might be considered acceptable interpretations [6]. The intended meaning of a metaphor may be related more to the topic, the vehicle, or to both. For example, when one says that *Joe is a chicken*, the meaning is usually described as being *afraid*, which is more closely related to the vehicle *chicken* than to the topic *Joe*.

We describe a system designed for interpreting nominal metaphors given without context. Similarly to previous works, we exploit a large corpus of text documents for semantically describing words and properties using a mathematical device. We use a word-embedding representation for calculating similarity between a candidate interpretation and the topic and the vehicle, so as to rank candidates based on a semantic score. As a final step, we automatically cluster results and keep only the best interpretations out of each cluster.

2 Metaphor Processing

Given a metaphor, we begin by generating a list of interpretation candidates. We do this by finding collocations of the topic and vehicle individually, and consider each one of them as a potential candidate. For each candidate c, we calculate a topic semantic score, which is the cosine similarity between c and the k most significant collocations of the topic (k is a parameter) and aggregate it into a single score. We use average for aggregation. Similarly, we calculate a vehicle semantic score.

In the next step, we calculate two pointwise mutual information (PMI) values, between c and the topic and vehicle respectively. We add the frequency of c as another score and combine all the five score functions in a log-linear structure, with weights assigned to each. The weights are adjusted automatically.

To remove semantically related interpretations from the list, we cluster the results and keep only the highest ranked candidates in each cluster. The remaining candidates are ranked according to their final score and the best n candidates (n, too, is a parameter) are returned as interpretations.

Similar to other relevant works, an *interpretation* is taken to be a single word that conveys the main concept of the metaphor. For example, among the interpretations of the metaphor *city is a jungle* one can find *crazy* and *crowded.* As is common works, we consider all adjectives as potential interpretations. Additionally, gerunds are good candidates. In [8], they only consider abstract words as potential interpretations. The level of abstractness of a word was measured by Turney et al. [7] automatically for about 11,000 words. To avoid the limitation in using such a list, we did not go that route.

Our interpretation process begins with extracting collocations of the vehicle and the topic individually using a relatively large corpus. Specifically, we use DepCC,[1] a dependency-parsed "web-scale corpus" based on CommonCrawl.[2] There are 365 million documents in the corpus, comprising about 252B tokens. Among other preprocessing steps, every sentence was given with word dependencies discovered by MaltParser [4]. We only use a fraction of the corpus containing some 1.7B tokens. Here, we considered a "collocation" to be words that are found to be dependent in either the topic or the vehicle, and assigned with a relevant part-of-speech tag: adjective or verb+*ing.* The main assumption is that many potential modifiers of a given noun will appear somewhere in the corpus as a dependent in the dependency graph.

For example, the dependency-based collocations for *school* are: *high*, *elementary*, *old*, *grad*, *middle*, *med*, *private*, *attending*, *graduating*, *secondary*, *leaving*, and *primary.*

To eliminate noisy results that might transpire given that the corpus was generated from the open web, we preserve only candidates that have an entry in WordNet [1].

In parallel with our objective data-driven collocation extraction process, we experimented with word associations as an alternative, more subjective, process

[1]`https://www.inf.uni-hamburg.de/en/inst/ab/lt/resources/data/depcc.html`
[2]`http://commoncrawl.org`

for generating interpretation candidates. Word-association norms are repositories of pairs of words and their association frequency in a given population. The first word is a cue or trigger given to participants, and the second is the reported associated word that first came to a subject's mind. For example, *bank* is paired with *money*, because the cue *bank* often elicits the response *money*. Those pairs form various semantic-relation types; some might not be deemed symmetric. Word association norms have been used in psychological and medical research, as well as a device for measuring creativity.

We use the University of South Florida Free (USF) Association Norms [3][3] for generating alternative candidates. This repository contains 5,019 cue words that were given to 6,000 participants beginning in 1973. The way we work with this repository is by adding all the associated words of the topic and vehicle individually. In this case, we allow words of all POS classes to be considered as candidates. For example, the associations for cue *school* are: *work*, *college*, *book*, *bus*, *learn*, *study*, *student*, *homework*, *teacher*, *class*, *education*, *USF* (!), *hard*, *boring*, *child*, *house*, *day*, *elementary*, *friend*, *grade*, *time*, *yard*. We evaluate our system's performance with and without the associations.

For each candidate we calculate a couple of semantic scores, one for the topic and one for the vehicle. We use word embeddings to transform every word into a continuous vector that captures the meaning of the word, as evidenced in the underlying corpus. We used pre-trained GloVe [5] vectors; specifically, we use the ones that were trained over a 6B token corpus, comprising 400K vectors, each of 300 dimensions. In what follows, we denote the vector of a word v by w_v.

We believe that the most significant collocations of the topic/vehicle tend to reliably represent the way the topic/vehicle, respectively, can be described in different contexts. Therefore, the semantic score $sem(c,t)$ of a candidate c and the topic t is the average cosine similarity between w_c and the vectors of the k most significant collocations of t. Similarly, $sem(c,v)$ is the semantic score of a candidate c and the vehicle v. We experimented with different values for k. Results are reported in the next section.

For each candidate c, we calculate $npmi(c,t)$ and $npmi(c,v)$, the normalized Pointwise Mutual Information (PMI) values for the topic and vehicle, respectively. Normalized PMI is similar to PMI, except that it is normalized between -1 and 1. The PMI between a candidate c and a noun n is calculated over the dependency graph; that is, we calculate the chances of seeing c as a dependent of n in a dependency graph. We add $freq(c)$, the frequency of c as another score, calculated over the entire corpus.

To summarize, given a candidate c, the full list of scores is

$$\langle sem(c,t), sem(c,v), npmi(c,t), npmi(c,v), freq(c) \rangle$$

[3]`http://w3.usf.edu/FreeAssociation`

combined using a log-linear structure, with each score amplified by a weight:

$$FinalScore(c) = \sum_{k=1}^{5} \lambda_k \log score_k$$

We automatically adjust these weights over a development set of metaphors and interpretations to optimize for recall, as explained below. As a result, each candidate is ranked according to its final score.

Lastly, we cluster the list of candidates as a way to deduplicate it. We run clustering using word vectors for finding groups of words that have a strong semantic association of any kind, keeping only the best candidates in each cluster.

We use Density-Based Spatial Clustering of Applications with Noise (DBSCAN) for clustering. This method groups together vectors that are bundled in the space by forcing a minimum number of neighbors. Vectors that do not have the requisite number of neighbors, or in other words occur in low-density areas, are reported as noise and are not placed under any cluster. For us it means that they were not connected with other vectors, so they might have a unique meaning among the listed candidates. We treat such vectors as if they form singletons.

For example, among the interpretation candidates for the metaphor *anger is fire* we find *red* and *black*. After clustering, *black* is removed. As another example, the following candidates for the metaphor *a desert is an oven* may be grouped together: *eating*, *healthy*, *delicious*, *fried*, *spicy*, *leftover*, *veggie*, *steamed*, *lentil*, *roasted*, *homemade*, *yummy*, *creamy*, *glazed*, *seasoning*, *crunchy*, *baking*. (These likely result from the frequent misspelling of “dessert” in the corpus used.)

There are two parameters that need to be configured for DBSCAN: (1) ε – the radius of the consideration area around every vector; and (2) μ – the minimum number of neighbors required in the consideration area. The distance measure should also be configured. We use the common Euclidean distance, which usually shows good performance in a relatively low-dimension space like ours.

3 Experimental Results

We evaluate our system with the dataset published by [6], containing 84 unique topic/vehicle pairs that were associated with interpretations by twenty different study participants. Each participant was asked to assign interpretation for different aspects of the pairs, such as treating a pair as a metaphor (e.g. *knowledge/power*, from the phrase *knowledge is power*) or as a simile (e.g. *knowledge/power*, from *knowledge is like power*). We focus on the interpretation of metaphors, both lexicalized and non-lexicalized.

As a preprocessing step, we lemmatize the interpretations, so as to allow our method's results and the true interpretations to match more smoothly. Additionally, we allow interpretations to match if they are considered as synonyms in WordNet. In this work we focus on nominal metaphors, and since our collocation as well as word-embedding models were trained to handle unigrams, we had to modify some of the metaphors that have multiword vehicles; for example *sermon is a sleeping pill*. Therefore, we modified such multiword vehicles into one word simply by eliminating the space characters, knowing it may cause performance reduction; for example, *sermon is a sleeping pill* was modified to *sermon is a sleepingpill*.

Each metaphor might be associated with more than one interpretation. As do other related works [8, 6], we only consider interpretations that were assigned by at least five participants; we call them *qualified interpretations*. This leaves us with only 76 qualified metaphors (i.e. metaphors with at least one qualified interpretation), with two qualified interpretations per metaphor on average.

To stay in line with related works [8], we report *Recall* @K, which is the average percentage of human-associated interpretations that are found in the top K results. For example, the following results were generated for the pair *skating/flying*: *incredible*, *high*, *free*, *great*, *fast*. Therefore, Recall@3 is 33%, while Recall@5 is 66%. We compare our results with [8], which was evaluated on the same dataset following a similar preprocessing step. Therefore, we report on Recall at their reported K's: 5, 10, 15, 25, and 50.

To measure the false positives reported by the system, we evaluate the results with two additional standard metrics: mean reciprocal rank (MRR) and mean average precision (MAP).

Our log-linear structure is composed of a set of weighted score functions. We adjust the scores using a tuning process over a development set, composed of about 50% of the metaphors. For each weight, we explore a range of possible scores, while we test all possible score combinations taking the brute force approach. For all scores except *freq*, we consider the range 0.1 .. 1; because of scale differences, for *freq* we consider the range 1 .. 10.

We take a similar brute force approach for tuning the DBSCAN clustering parameters, ε and μ. We also tune n, the number of top results taken from each cluster. For tuning, we use the same development set, evaluated over MRR, MAP and Recall @K values. Table 1 shows the ranges and best values of all the parameters we tune.

We see that both semantic scores get higher weights than the npmi scores, suggesting that the semantic distance as measured by cosine similarity between the vectors of the candidates and the collocations of the topic/vehicle, is effective. The DBSCAN parameters are less stable across different metric optimizations. One thing we learn is that when optimizing for larger values of @K, DBSCAN requires dense areas around clustered vectors, resulting in a lower number of clusters. Additionally, the system does not benefit from high values of the DBSCAN n parameter. It turns out that it is better to consider only one interpretation from each cluster.

We evaluate our system against the 76 "qualified" metaphors in the dataset. For each metaphor, our system generates the top 100 interpretation results, which are then compared with the metaphor's human-associated qualified interpretations. For the clustering parameters and scoring weights, we use the tuned

Table 1: System parameters tuned to maximize MRR, MAP and Recall@K. The second column shows the range of values considered.

Param	**Range**	**MRR**	**MAP**	**@5**	**@10**	**@15**	**@25**	**@50**
DBSCAN ε	1 .. 6	4	5	4	5	5	4	4
DBSCAN μ	1 .. 5	4	1	6	1	5	5	5
DBSCAN n	1 .. 12	1	1	1	1	1	1	1
$sem(c, t)$	0.1 .. 1	0.6	0.6	0.6	0.1	0.1	0.6	0.6
$sem(c, v)$	0.1 .. 1	1.1	1.1	1.1	0.6	0.6	1.1	1.1
$npmi(c, t)$	0.1 .. 1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
$npmi(c, v)$	0.1 .. 1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
$freq(c)$	1 .. 10	3	3	5	7	7	5	3

Table 2: Each row shows evaluation results when using optimal parameter values for the metric mentioned in the first column.

Optimization	**MRR**	**MAP**	**@5**	**@10**	**@15**	**@25**	**@50**
MRR	**0.312**	**0.170**	0.198	0.254	0.278	0.405	0.562
MAP	**0.312**	**0.170**	0.198	0.254	0.278	0.405	0.562
@5	0.302	0.166	**0.207**	0.258	0.270	0.430	0.548
@10	0.233	0.151	0.180	**0.273**	0.322	0.374	0.521
@15	0.245	0.160	0.151	0.262	**0.331**	0.392	0.513
@25	0.302	0.166	**0.207**	0.258	0.270	**0.430**	0.548
@50	**0.312**	**0.170**	0.198	0.254	0.278	0.405	**0.562**

values reported in the previous subsections. Since we tune for different evaluation metrics, here we individually use each set of values for generating the top 100 results and calculating MRR, MAP and recall at all the relevant K values. Table 2 summarizes the evaluation results at MRR, MAP and Recall@5, @10, @15, @25, and @50, for each set of parameter values. We observe that when optimizing the system for Recall@50 we at least get close to the best result for all other evaluation metrics. Therefore, in what follows we use the parameter values optimized for @50.

We compare our results with the ones reported by Meta4meaning [8], evaluating over the same set of metaphors and following similar preprocessing steps. Table 3 compares the results reported by both systems. While our system somewhat underperforms for the lower values of Recall @k, it is doing slightly better on @25 and @50. These results show that, while our system has a better overall coverage, correct interpretations are concentrated more in the lower part of the ranked list that we produce. With more work, we expect to be able to filter out many of the non-associated interpretations, thereby ranking the correct ones higher in the list.

To measure the effect of clustering on the results, we evaluate our system running with and without clustering. When running with clustering, we use the optimized set of parameters, as reported in Table 1. Table 4 compares

Table 3: Comparison with Meta4meaning.

System	**MRR**	**MAP**	**@5**	**@10**	**@15**	**@25**	**@50**
Meta4meaning	N/A	N/A	**0.221**	**0.303**	**0.339**	0.397	0.454
Ours	0.312	0.170	0.198	0.254	0.278	**0.405**	**0.562**

Table 4: Evaluation results, with and without clustering.

Method	**@5**	**@10**	**@15**	**@25**	**@50**
w/o clustering	0.198	0.254	0.278	0.351	0.534
w/ clustering	0.198	0.254	0.278	**0.405**	**0.562**

our system's results, with and without clustering. We learn that when using clustering, our system was able to eliminate noise in lower parts of the ranked list of candidates, thereby making room for alternative and correct interpretations that ranked lower without clustering.

Finally, we check how our system's performance is affected by adding word associations as an additional source for generating interpretation candidates. When we run our system using only dependency-based collocations as candidates, we obtain Recall@50 score of 0.55. This was improved to 0.56 when we add word associations as candidates.

4 Conclusions

We have described a system that interprets nominal metaphors, provided without a context. Given a metaphor, we generate a set of interpretation candidates and rank them according to how strongly they are associated with the topic, as well as with the vehicle. Candidates are generated using two techniques. First, we find collocations of the topic and vehicle, focusing on adjectives as well as gerunds, which were found to be dependent of the topic/vehicle in at least one sentence in a large corpus. We add to that list word associations of both. This addition has proven effective.

Our ranking procedure combines a number of scores assigned for each candidate, which are based on normalized PMI as well as cosine similarity between the representing GloVe vectors of the candidates and the topic/vehicle collocations. The scores are aggregated using a weighted log-linear structure. We tune the weights automatically, optimizing for various evaluation metrics: MRR, MAP and Recall@K for different K values. We found that with small K, the similarity between candidate and topic becomes more important than other score functions.

In a post-processing step, we cluster the results using DBSCAN and keep only the best candidates out of each cluster. The system benefits thereby.

Our system was evaluated against a set of metaphors that were assigned with properties by 20 human evaluators. We compare our results with Meta4meaning and obtained competitive results. Overall, the system could not generate even one correct interpretation (among its 50 best results) for 20 out of the 76 evaluated metaphors. Some of those metaphors did not have a correct interpretation anywhere in the list, even beyond the best 50; for example, *music is a medicine*.

References

[1] Christiane Fellbaum, editor. *WordNet: an electronic lexical database*. MIT Press, 1998.

[2] D. Gentner, B. F. Bowdle, P. Wolff, and C. Boronat. Metaphor is like analogy. In *The Analogical Mind: Perspectives from cognitive science*, chapter 6, pages 199–253. MIT Press, Cambridge, MA, 2001.

[3] Douglas L. Nelson, Cathy L. McEvoy, and Thomas A. Schreiber. The University of South Florida free association, rhyme, and word fragment norms. *Behavior Research Methods, Instruments, & Computers*, 36(3):402–407, 2004.

[4] Joakim Nivre and Johan Hall. Maltparser: A language-independent system for data-driven dependency parsing. In *In Proc. of the Fourth Workshop on Treebanks and Linguistic Theories*, pages 13–95, 2005.

[5] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[6] Carlos Roncero and Roberto G. de Almeida. Semantic properties, aptness, familiarity, conventionality, and interpretive diversity scores for 84 metaphors and similes. *Behavior Research Methods*, 47(3):800–812, 2015.

[7] Peter D. Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 680–690, Stroudsburg, PA, 2011. Association for Computational Linguistics.

[8] Ping Xiao, Khalid Alnajjar, Mark Granroth-Wilding, Kathleen Agres, and Hannu Toivonen. Meta4meaning: Automatic metaphor interpretation using corpus-derived word associations. In *Proceedings of the Seventh International Conference on Computational Creativity (ICCC)*, pages 230–237, Paris, France, June 2016.