MIL: Automatic Metaphor Identification by Statistical Learning

Yosef Ben Shlomo and Mark Last

Department of Information Systems Engineering
Ben-Gurion University of the Negev
Beer-Sheva 84105, Israel
E-mail: bshyossi@gmail.com, mlast@bgu.ac.il

Abstract. Metaphor identification in text is an open problem in natural language processing. In this paper, we present a new, supervised learning approach called MIL (Metaphor Identification by Learning), for identifying three major types of metaphoric expressions without using any knowledge resources or handcrafted rules. We derive a set of statistical features from a corpus representing a given domain (e.g., news articles published by Reuters). We also use an annotated set of sentences, which contain candidate expressions labelled as 'metaphoric' or 'literal' by native English speakers. Then we induce a metaphor identification model for each expression type by applying a classification algorithm to the set of annotated expressions. The proposed approach is evaluated on a set of annotated sentences extracted from a corpus of Reuters articles. We show a significant improvement vs. a state-of-the-art learning-based algorithm and comparable results to a recently presented rule-based approach.

Keywords: Metaphor Identification, Natural Language Processing, Supervised Learning.

1 Introduction

A metaphor is defined in previous works (such as Krishnakumaran & Zhu, 2007) as the use of terms related to one concept in order to describe a term, which is related to a different concept. For example, in the metaphoric expression “fertile imagination”, the word “imagination”, which is related to the concept / domain “cognition”, is described by the word “fertile”, which is usually related to the concept / domain, “land / soil”. Metaphor identification can be useful in numerous applications that require understanding of the natural language such as machine translation, information extraction, and automatic text summarization. For example, the word “жесткая” in Russian may have a metaphorical meaning of “tough” when applied to the word “политика” (policy) or a literal meaning of “hard” when applied to the word “кровать” (bed).

Metaphoric expressions have a variety of syntactic structures. In this paper, we focus on three major types of syntactic structures discussed in the work of (Krishnakumaran and Zhu, 2007). In type 1 expression, a subject noun is related to an object noun by a form of the copula verb “to be” (e.g., “God is a father”). In type 2 expression, the subject

In: P. Cellier, T. Charnois, A. Hotho, S. Matwin, M.-F. Moens, Y. Toussaint (Eds.): Proceedings of DMNLP, Workshop at ECML/PKDD, Nancy, France, 2014.

noun is associated with a metaphorically used verb and an object noun (e.g., "The actor painted his relationship with…"). Type 3 expression is an adjective-noun phrase (e.g., "sweet child"). An expression of one of the three structures above may or may not be a metaphor. Therefore, in this paper we treat the problem of metaphor identification as a binary classification problem of labelling a given expression as metaphoric or literal. The first step in our method is choosing a corpus representative of a given domain and building an annotated data set of labelled expressions from that corpus. The second step is feature extraction from the domain corpus, which is the main subject of our work. Then we induce a classification model for each expression type, using a set of annotated sentences, and evaluate its accuracy on a hold-out set.

This paper is organized as follows. Section 2 covers the state-of-the-art approaches to automated metaphor detection. Section 3 presents MIL (Metaphor Identification by Learning), a novel algorithm for metaphor detection. The proposed algorithm is empirically evaluated in Section 4. In Section 5, we conclude the paper with some insights and directions for future research.

2 Related Work

The work (Birke & Sarkar, 2006) focused on classifying the uses of verbs in a sentence as either literal or non-literal (type 2 expressions). They adopted the work of (Karov & Edelman, 1998), who worked on word sense disambiguation of words within their contexts (i.e., sentences).

Krishnakumaran & Zhu (2007) suggested three algorithms for distinguishing between live and dead metaphors. A dead metaphor is a metaphor that is already assimilated and familiar in the spoken language (e.g., "fell in love") and a live metaphor is a less familiar metaphor, which is not yet assimilated. The methodology of Krishnakumaran & Zhu 2007's work is based on conditional probabilities combined with WordNet (Fellbaum, 1999), which represents the language ontology.

The authors of (Neuman, et al., 2013) present three rule-based algorithms for metaphor identification in three expression types (1, 2, and 3). They suggest identifying metaphors by negating literalness. They define a set of rules for detecting whether a particular expression is literal, and if it is not literal, it is assumed to be a metaphor. Following the work of Turney, et al. (2011), they define as literal an expression comprised of words (e.g., verb and noun), which have in common at least one concrete category. Examples of concrete categories include *physical objects*, like "table", or *body parts*, like "hair". The ruleset defined by (Neuman, et al., 2013) builds upon multiple knowledge resources (such as Wiktionary and WordNet).

The work of (Shutova, et al. 2010) focuses on *semi-supervised learning* for identification of type 2 metaphors. The metaphor identification process is based on the principle of clustering by association, i.e., the clustering of words by their associative language neighborhood using the verb's subject as an anchor. A domain-independent approach to type 2 metaphor identification is presented in (Shutova, et al., 2013). The authors report a high precision of 0.79, but no information about the system recall and F-measure is provided.

Turney, et al. (2011) provide a solution for the type 3 metaphor classification problem called the *Concrete-Abstract* algorithm. In contrast to previous works that treated this issue as a sub-problem of the Word Sense Disambiguation problem, Turney, et al. (2011) suggest identifying metaphors by considering the abstractness level of words in a given expression. They found that in a wide range of type 3 metaphoric expressions, nouns tend to be abstractive while adjectives tend to be concrete. Their methodology builds upon a unique algorithm for ranking the abstractness level of a given word by comparing it to 20 abstract words and 20 concrete words that are used as paradigms of abstractness and concreteness. The main limitation of this approach is that it uses a single feature (abstractness level) that covers a limited range of expressions. The classification model used by Turney, et al. (2011) is logistic regression.

The authors of (Hovy, et al., 2013) use SVMs with tree kernels for supervised metaphor classification based on a vector representation of the semantic aspects of each word and different tree representations of each sentence. They report the best F1 measure of 0.75 without specifying the distribution of metaphor types in their dataset. Very similar results (F-measure = 0.76 for English) are reported for type 2 and type 3 expressions by (Tsvetkov, et al., 2014) who use three categories of features: 1) abstractness and imageability, (2) word supersenses (extracted from WordNet), and (3) unsupervised vector-space word representations. Using translation dictionaries, the authors apply a trained English model to three other languages (Spanish, Russian, and Farsi). The English dataset used by (Tsvetkov, et al., 2014) included 3,737 annotated sentences from the *Wall Street Journal* domain.

3 MIL (Metaphor Identification by Learning)

3.1 Overview

The first step in our proposed methodology is choosing a domain corpus, which represents a particular domain (area) of text documents (e.g., Reuter's news articles). The next step is the extraction of candidate expressions that satisfy one of the three syntactic structures discussed above (types 1, 2, and 3). This can be done using existing natural language processing tools. We also use the domain corpus to construct a word-context matrix to be used in the feature extraction phase. Then we manually annotate a selected subset of candidate expressions, since in a large corpus, it is not feasible to annotate them all. Each selected expression is labeled as 'literal' or metaphoric' by native language speakers.

Then we perform feature extraction, the largest and most important task in our work. The goal of feature extraction is to compute statistical features that may differentiate between metaphor and literal expressions. After the feature extraction is complete, a feature selection process must be carried out for choosing the features that are most relevant to the classification task.

The next step is inducing a classification model by running a supervised learning algorithm (e.g., C4.5). The model is built separately for each syntactic type. The resulting classification model can be used for classification of new expressions as literal

or metaphoric. The model performance can be evaluated by such measures as precision and recall using cross-validation.

3.2 Domain Corpus Pre-processing

The basic actions on a domain corpus are parsing the corpus into sentences, tokenization, stemming the tokens by the Porter Stemmer algorithm (Porter, 1980), removing stopwords, and then calculating the frequency of each token. Then we build a co-occurrence matrix for words with frequency of 100 and higher (like in Turney et al., 2011). The co-occurrence matrix is used for calculating the abstractness level of a given word. The idea behind this matrix is that the co-occurring words are more likely to share an identical concept. We denote the co-occurrence matrix by *F*. Then we calculate the Positive Pointwise Mutual Information (PPMI) of *F*. The purpose of PPMI is to prevent the bias that can be caused by highly frequent words. We calculate PPMI as described in (Turney and Pantel, 2010) and get a new matrix denoted as *X*.

The *X* matrix is smoothed with truncated Singular Value Decomposition (SVD) (Deerwester, et al., 1990), which decomposes *X* into a multiplication of three matrices $X \approx U_k S_k V_k^T$. The matrix U_k represents the left singular vectors, the matrix S_k is a diagonal matrix of the singular values and the matrix V_k^T represents the right singular vectors. The idea behind using SVD is that a conceptual relation between two words can be indicated by a third word called a "latent factor" (e.g., the relation between *soccer* and *basketball* can be established by the word *sport* even if that word does not occur in text).

SVD calculation has two parameters. The first one is *k*, which represents the number of latent factors. We manually calibrated *k* starting from the value of 1000 as used in the work of (Turney, et al., 2011) and reduced it by 100 at each iteration in order to reduce the running time (more latent factors means longer running time), without decreasing the quality of results. We stopped when there was a substantial decrease in the results accuracy. In this manner, we have set the best value of *k* to 300. The other parameter is *p*, which adjusts the weights of the latent factors as in (Caron, 2001). We adopted its value from the work of (Turney, et al., 2011) and set it to 0.5. The second matrix we use is the multiplication of $U_k S_k^p$. We use this matrix for computing the *semantic similarity* of two words as a cosine similarity of two matrix rows (Turney and Pantel, 2010). We annotate $U_k S_k V_k^T$ as *matA* and $U_k S_k^p$ as *matB*.

Finally, we represent each of the corpus documents by calculating the average vector of the frequent words the document contains. The idea behind that is that in our view, each document represents some concept of its own and it can contribute to the identification of the concept transition.

3.3 Feature Extraction

The proposed feature set contains four types of features: features, which apply to every single word in a candidate expression (two features for type 1 and type 3 expressions, three features for type 2), features, which apply to every word pair in a candidate expression (one feature for type 1 and 3, three features for type 2), features, which apply

to the sentence containing the candidate expression, and features, which apply to the candidate expression itself.

The first set of statistical-based features builds upon the idea of concrete-abstract mapping (Turney et al., 2011). They present a measure of the abstractness level, which can be calculated for each word in a candidate expression. We are using this measure to define the following features:

- *Abstract Scale*. The abstractness level of a word according to the algorithm presented in (Turney et al., 2011) is calculated as:

$$\sum_{k=1}^{20} pCor(word, abstract\ paradigm) - \sum_{k=1}^{20} pCor(word, concrete\ paradigm) \quad (1)$$

where *word* is a semantic vector of the target word, *abstract paradigm* is a semantic vector of a very abstractive word (e.g., sense), *concrete paradigm* is a vector of a very concrete word (e.g., donut) and $pCor$ is the Pearson correlation. The abstract scale rank is normalized between zero and one. The semantic vectors are taken from a pre-processed matrix based on words' co-occurrences. The full list of 20 abstract paradigm words and 20 concrete paradigm words is given in (Turney et al., 2011). This feature is applied to every word in a given expression.
- *Abstract Scale difference*. The absolute difference between the abstractness levels of every two words in an expression.
- *Abstract Scale Average / Variance*. The average / variance of the abstractness levels of all frequent words in the sentence that contains the candidate expression.

We also define a set of statistical-based features that can indicate a conceptual mapping between different word categories. These features include word-level features, document-level features, and domain-level features. Word-level features are features which are associated with the semantic meaning of a given word. Document-level features are features which are associated with the entire document's meaning (document which contains the expression) and domain-level features are features, which are associated with the entire domain corpus. The statistical-based features are defined below:

Word-level features

- *Semantic Relation*. The semantic relation value between every two words in a given expression. This value is taken from the associated entry of this words pair in the pre-processed *matA* (the large NxN matrix). Low values are an indication for low semantic relation between two words and thus imply metaphoric behavior (the conceptual mapping definition) and vice versa.
- *Semantic Relation Average / Variance*. The average / variance of the semantic relation values between every two words in the sentence that contains the candidate expression (including the expression itself).

- *Cosine-similarity*. The cosine similarity between every two words in a given expression. The cosine similarity is between the two vectors associated with the given words pair, taken from the pre-processed *matB* (the low dimensionality matrix). Low values indicate low conceptual relation between two words and thus imply metaphoric behavior.
- *Cosine-similarity Average / Variance*. The average / variance of the cosine similarity between every two words in a given expression. Low values indicate low conceptual relation between two words and thus imply metaphoric behavior.
- *First k Words*. For each two words in a given expression, we first find the *k* most similar words to the noun (if both words are nouns then the subject noun is selected). We then calculate the average cosine similarity of the second word to those *k* words and use it as a feature. A similarity between two words is computed as the cosine similarity of their associated vectors in *matB*. The value $k = 30$ was chosen based on the quality of classification results.

Document-level features

- *First k Documents*. Same as *First k Words*, based on document vectors rather than word vectors. Considering a document as a topic/s oriented, its representation by its word vectors average is actually the semantic representation of its topic/s. Here we also set *k* to 30 after manual calibration considering the results quality (F-measure).
- *Documents Jaccard Similarity*. It is calculated between semantically related neighborhoods of each two words in a given expression. The word neighborhood is defined as a window of ± 5 words surrounding a given word in the same sentence. In the previous features, we detect a conceptual mapping between two words by enriching *one* of the words with its semantically related neighborhood. In this feature, we take this idea one-step further and enrich *both words* with their semantically related neighborhoods. The feature is the Jaccard similarity of these two neighborhoods, calculated as follows:

$$\frac{SDT(w1,T) \cap SDT(w2,T)}{SDT(w1,T) \cup SDT(w2,T)}. \quad (2)$$

Where *SDT(x)* is a group of documents that are related to word *x* according to a predefined threshold *T*. In our experiments, we manually set *T* to 0.35 after trying different values.

Domain-level features

- *Domain Corpus Frequency*. The normalized frequency of a word in the domain corpus. This feature is extracted for every word in the candidate expression. A low word's frequency in a given corpus can indicate metaphoric behavior since the word is not strongly related to the given domain and thus is used there as a metaphor.
- *Domain Corpus Frequency Difference*. The absolute value of the difference between the frequencies of every two words in a given expression.
- *Positive Point-wise Mutual Information*. This feature (based on Turney and Pantel, 2010) is applied to every two words in a given expression.

3.4 Feature and Model Selection

We used the Wrapper method of (Kohavi & John, 1997) to select the best features for each expression type. This method gets as input a classifier (e.g., decision tree) and a criterion to maximize (e.g., accuracy), and finds the subset of features that maximizes this criterion. We applied this method to each expression type. The criterion we used was the F-measure. We applied the following classifiers: Logistic Regression, Naïve Bayes, K-Nearest Neighbors (KNN), Voting Features Intervals (VFI), Random Forest, Random Tree, and J-48. We also combined each one of them with the AdaBoost algorithm.

All algorithms were run on Weka (Hall, et al., 2009), an open source implementation of Machine Learning algorithms. For each domain and expression type, we used the Wrapper method with 10-fold cross-validation to choose the algorithm and the feature set that provided the maximum average F-measure value over the ten splits of the dataset.

4 Experimental Results

4.1 Corpora

Our results in this paper are based on the Reuters Corpus, which was previously used as a domain corpus in (Neuman, et al., 2013). It consists of 342,000 English documents, which include 3.9 million sentences.

We used the same annotated corpus as in (Neuman, et al., 2013). The annotated corpus was constructed by extracting from the domain corpus sentences containing one of the five target nouns related to the concepts of "government" and "governance" in both literal and metaphorical sense: Father, God, Governance, Government, and Mother. Every selected sentence was parsed with the Stanford Part-of-Speech Tagger (de Marneffe & Manning, 2008). Candidate expressions having one of the three syntactic structures were independently denoted as metaphoric or literal by four human judges who were given the following definition of a metaphoric expression:

Literal is the most direct or specific meaning of a word or expression. *Metaphorical is the meaning suggested by the word that goes beyond its literal sense*.

The annotators were also given examples of literal and metaphorical expressions of types 1, 2, and 3. Inter-annotator agreement, measured in terms of Cronbach's alpha, was 0.78, 0.80, and 0.82 for type I, II, and III, respectively (Neuman, et al., 2013).

Finally, an expression was labeled as metaphoric if at least three judges out of four considered it as such; otherwise, it was labelled as literal. **Table 1** shows the distribution of expressions by their type and label (Literal / Metaphorical) in the set of annotated sentences.

Table 1. Candidate Expressions, Reuters Annotated Set (Neuman, et al., 2013)

Annotators Decision\ Type	Type 1	Type 2	Type 3
Literal	40 (33.3%)	242 (48.6%)	576 (75.8%)
Metaphorical	80 (66.7%)	256 (51.4%)	184 (24.2%)
Total	120	498	760

4.2 Comparative Results

In this section, we present the comparative results of MIL vs. two state-of-the-art algorithms - Concrete-Abstract (Conc-Abs) of Turney, et al. (2011) and CCO of Neuman, et al. (2013), which so far have reported the best performance, in terms of the F-measure on all three types of metaphoric expressions. The following performance measures were calculated using 10-fold cross-validation: precision, recall, and F-measure. The comparative results are shown in Tables 2-4 for expressions of type 1, 2, and 3, respectively. The Conc-Abs and CCO results are replicated from (Neuman, et al., 2013) and they refer to the same domain (Reuters) and the same set of annotated sentences.

Table 2. Comparative results type 1, Reuters

	MIL	Conc-Abs	CCO
Precision	86	76.5	83.9
Recall	92.5	76.5	97.5
F-measure	89.2	76.5	90.1

Table 3. Comparative results type 2, Reuters

	MIL	Conc-Abs	CCO
Precision	65.2	63.9	76.1
Recall	77	67.2	82
F-measure	70.6	65.4	78.9

Table 4. Comparative results type 3, Reuters

	MIL	Conc-Abs	CCO
Precision	46.8	0	54.4
Recall	39.7	0	43.5
F-measure	42.9	0	48.3

The MIL algorithm has clearly outperformed the Concrete-Abstract algorithm in terms of F-measure with an advantage of 12.7%, 5.2%, and 42.9% for expression types

1, 2, and 3, respectively. Moreover, all type 3 expressions were identified by the Concrete-Abstract algorithm as literal leading to the F-measure of zero. The most likely reason for that is that our dataset rarely contains expressions where the noun is an abstract noun (e.g., the noun "thoughts" in the expression "dark thoughts" is an abstract noun) and most metaphoric expressions contain concrete nouns (e.g., the noun "heart" in the expression "broken heart" is a concrete noun). Since Conc-Abs relies on a single feature, which is the noun's abstractness level, it cannot detect a metaphor in these cases. However, CCO outperformed MIL, especially in type 2 and type 3 expressions. Unlike MIL, which is a supervised learning approach, CCO is a rule-based method, requiring for each new language and domain a significant amount of manual expert labor along with multiple high-quality knowledge resources, which are unavailable for most human languages. The F-measure results reported by (Tsvetkov, et al., 2014) for type 2 and type 3 expressions (76%) are also better than the results reached by MIL, but their system is dependent upon a massive knowledge resource –WordNet.

Table 5 shows the list of features and the classifier selected by the Wrapper method of (Kohavi & John, 1997) for each expression type. We can conclude from the selected feature list that the feature *First k Documents* is a general feature, since it has been selected in all three expression types. The following features have been selected for two expression types out of three: *Domain Corpus Frequency* (Types 1 and 3), *Cosine-similarity Variance* (Types 1 and 3), and *Cosine-similarity Average* (Types 2 and 3). These results imply that for detecting conceptual mapping between two words in a given expression, it can be useful to consider the semantic neighborhood of each word as well as its frequency in the domain corpus.

Table 5. Features and classifier selected by the Wrapper method, Reuters

Expression Type	Selected Features	Selected Classifier
Type 1	Semantic Relation First *k* Documents Domain Corpus Frequency Abstract Scale Average Cosine-similarity Variance	Random Forest
Type 2	Abstract Scale First *k* Documents First *k* Words Semantic Relation Average Cosine-similarity Average	AdaBoost with Naïve Base
Type 3	Cosine-similarity First k Documents Abstract Scale difference Documents' Jaccard Similarity Domain Corpus Frequency Domain Corpus Frequency Difference Cosine-similarity Average Cosine-similarity Variance	AdaBoost with VFI

5 Conclusion

In this paper, we have presented a novel supervised learning approach for automatic metaphor identification in three syntactic structure types. We have extended the single feature set used by Turney, et al. (2011) with a large amount of statistical features. We have shown a significant improvement vs. a learning-based algorithm (Concrete-Abstract). However, MIL was outperformed by a rule-based algorithm (CCO), which applies a set of rules to a candidate expression in order to determine if it is a literal or not. In CCO, the rules are generated separately for each of the three expression types, and if a candidate expression satisfies all of them, it is labelled as literal. Otherwise, it is labeled as a metaphor. Although CCO outperformed MIL, it has some major disadvantages. One of the major disadvantages is that the rules are based on a relatively large amount of linguistic resources, including COCA (Corpus of Contemporary American English http://www.ngrams.info/), ConceptNet (http://conceptnet5.media.mit.edu/), WordNet (https://wordnet.princeton.edu/), and Wiktionary (https://en.wiktionary.org/wiki/English).

Future research on using statistical features for metaphor detection may include experimentation with additional predictive features, domains, and languages. Transfer learning across different domains may also be explored.

References

Krishnakumaran, S., & Zhu, X. (2007). Hunting Elusive Metaphors Using Lexical Resources. *Proceedings of the Workshop on Computational Approaches to Figurative Language*, (pp. 13-20). Stroudsburg, PA, USA.

Birke , J., & Sarkar, A. (2006). A Clustering Approach for the Nearly Unsupervised Recognition of Nonliteral Language. *In Proceedings of EACL-06*, (pp. 329–336).

Caron, J. (2001). Experiments with LSA scoring: Optimal rank and basis. *Proceedings of the SIAM Computational Information Retrieval Workshop*.

de Marneffe, M.-C., & Manning, C. D. (2008). The Stanford typed dependencies representation. *Proceedings of the workshop on Cross-Framework and Cross-Domain Parser Evaluation* (pp. 1-8). Association for Computational Linguistics.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 391-407.

Fellbaum, C. (1999). *WordNet: An Electronic Lexical Database*. Blackwell Publishing Ltd.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations, 11*(1).

Hovy, D., Srivastava, S., Jauhar, S. K., Sachan, M., Goyal, K., Li, H., . . . Hovy, E. (2013). Identifying metaphorical word use with tree kernels. *Proceedings of the First Workshop on Metaphor in NLP*, (pp. 52-57).

Karov, Y., & Edelman, S. (1998). Similarity-based word sense disambiguation. *Computational Linguistics*, 41-59.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence, 97*(1-2), 273–324.

Neuman, Y., Assaf, D., Cohen, Y., Last, M., Argamon, S., Howard, N., & Frieder, O. (2013). Metaphor Identification in Large Texts Corpora. *PLOS ONE*, 1-9.

Shutova, E., Korhonen, A., & Teufel, S. (2013). Statistical Metaphor Processing. *Computational Linguistics*, 301-353.

Shutova, E., Sun , L., & Korhonen, A. (2010). Metaphor Identification Using Verb and Noun Clustering. *COLING '10 Proceedings of the 23rd International Conference on Computational Linguistics*, (pp. 1002-1010). Beijing.

Tsvetkov, Y., Boytsov, L., Gershman, A., Nyberg, E., & Dyer, C. (2014). Metaphor Detection with Cross-Lingual Model Transfer. *ACL-2014*, (pp. 248-258).

Turney, P. D., & Pantel, P. (2010). From Frequency to Meaning : Vector Space Models of Semantics. *Journal of artificial intelligence research*, 1-48.

Turney, P., Neuman, Y., Assaf, D., & Cohen, Y. (2011). Literal and Metaphorical Sense Identification through Concrete and Abstract Context. *Proceedings of the 2011 Conference on the Empirical Methods in Natural Language Processing*, (pp. 680-690).