Identifying Metaphorical Word Use with Tree Kernels

Dirk Hovy[1] **Shashank Srivastava**[2] **Sujay Kumar Jauhar**[2] **Mrinmaya Sachan**[2]
Kartik Goyal[2] **Huiying Li**[2] **Whitney Sanders**[2] **Eduard Hovy**[2]
(1) ISI, University of Southern California, Marina del Rey
(2) LTI, Carnegie Mellon University, Pittsburgh
dirkh@isi.edu, {shashans,sjauhar,mrinmays,kartikgo,huiyingl,wsanders,hovy}@cs.cmu.edu

Abstract

A metaphor is a figure of speech that refers to one concept in terms of another, as in "He is such a *sweet* person". Metaphors are ubiquitous and they present NLP with a range of challenges for WSD, IE, etc. Identifying metaphors is thus an important step in language understanding. However, since almost any word can serve as a metaphor, they are impossible to list. To identify metaphorical use, we assume that it results in unusual semantic patterns between the metaphor and its dependencies. To identify these cases, we use SVMs with tree-kernels on a balanced corpus of 3872 instances, created by bootstrapping from available metaphor lists.[1] We outperform two baselines, a sequential and a vector-based approach, and achieve an F1-score of 0.75.

1 Introduction

A metaphor is a figure of speech used to transfer qualities of one concept to another, as in "He is such a sweet person". Here, the qualities of "sweet" (the *source*) are transferred to a person (the *target*). Traditionally, linguistics has modeled metaphors as a mapping from one domain to another (Lakoff and Johnson, 1980).

Metaphors are ubiquitous in normal language and present NLP with a range of challenges. First, due to their very nature, they cannot be interpreted at face value, with consequences for WSD, IE, etc. Second, metaphors are very productive constructions, and almost any word can be used metaphorically (e.g., "This is the **Donald Trump** of sandwiches."). This property makes them impossible to pre-define or list. Third, repeated use of a metaphor eventually solidifies it into a fixed expression with the metaphorical meaning now accepted as just another sense, no longer recognized as metaphorical at all. This gradient makes it hard to determine a boundary between literal and metaphorical use of some expressions. Identifying metaphors is thus a difficult but important step in language understanding.[2]

Since many words can be productively used as new metaphors, approaches that try to identify them based on lexical features alone are bound to be unsuccessful. Some approaches have therefore suggested considering distributional properties and "abstractness" of the phrase (Turney et al., 2011). This nicely captures the contextual nature of metaphors, but their ubiquity makes it impossible to find truly "clean" data to learn the separate distributions of metaphorical and literal use for each word. Other approaches have used pre-defined mappings from a source to a target domain, as in "X is like Y", e.g., "emotions are like temperature" (Mason, 2004). These approaches tend to do well on the defined mappings, but they do not generalize to new, creative metaphors. It is doubtful that it is feasible to list all possible mappings, so these approaches remain brittle.

In contrast, we do not assume any predefined mappings. We hypothesize instead that if we interpreted every word literally, metaphors will manifest themselves as unusual semantic compositions. Since these compositions most frequently occur

[1] Available at http://www.edvisees.cs.cmu.edu/metaphordata.tar.gz

[2] Shutova (2010) distinguishes between metaphor *identification* (which she calls recognition) and *interpretation*. We are solely concerned with the former.

in certain syntactic relations, they are usually considered semantic preference violations; e.g., in the metaphorical "You will have to **eat** your words", the food-related verb heads a noun of communication. In contrast, with the literal sense of "eat" in "You will have to **eat** your peas", it heads a food noun. This intuition is the basis of the approaches in (Iverson and Helmreich, 1991; Krishnakumaran and Zhu, 2007; Baumer et al., 2010; Turney et al., 2011).[3] We generalize this intuition beyond preference selections of verbs and relational nouns.

Given enough labeled examples of a word, we expect to find distinctive differences in the compositional behavior of its literal and metaphorical uses in certain preferred syntactic relationships. *If we can learn to detect such differences/anomalies, we can reliably identify metaphors.* Since we expect these patterns in levels other than the lexical level, the approach expands well to creative metaphors.

The observation that the anomaly tends to occur between syntactically related words makes dependency tree kernels a natural fit for the problem. Tree kernels have been successfully applied to a wide range of NLP tasks that involve (syntactic) relations (Culotta and Sorensen, 2004; Moschitti, 2006; Qian et al., 2008; Giuliano et al., 2009; Mirroshandel et al., 2011).

Our contributions in this paper are:

- we annotate and release a corpus of 3872 instances for supervised metaphor classification
- we are the first to use tree kernels for metaphor identification
- our approach achieves an F1-score of 0.75, the best score of of all systems tested.

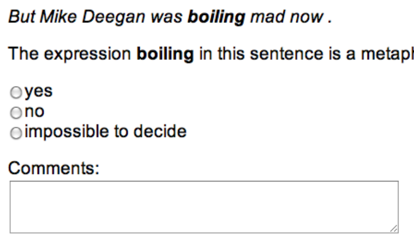
2 Data

2.1 Annotation

We downloaded a list of 329 metaphor examples from the web[4]. For each expression, we extracted sentences from the Brown corpus that contained the seed (see Figure 1 for an example). To decide whether a particular instance is used metaphorically, we set up an annotation task on Amazon Mechanical Turk (AMT).

Annotators were asked to decide whether a highlighted expression in a sentence was used metaphorically or not (see Figure 2 for a screenshot). They were prompted to think about whether the expression was used in its original meaning.[5] In some cases, it is not clear whether an expression is used metaphorically or not (usually in short sentences such as "That's sweet"), so annotators could state that it was not possible to decide. We paid $0.09 for each set of 10 instances.

Each instance was annotated by 7 annotators. Instances where the annotators agreed that it was impossible to tell whether it is a metaphor or not were discarded. Inter-annotator agreement was 0.57, indicating a difficult task. In order to get the label for each instance, we weighted the annotator's answers using MACE (Hovy et al., 2013), an implementation of an unsupervised item-response model. This weighted voting produces more reliable estimates than simple majority voting, since it is capable of sorting out unreliable annotators. The final corpus consisted of 3872 instances, 1749 of them labeled as metaphors.


*But Mike Deegan was **boiling** mad now .*

The expression **boiling** in this sentence is a metaphor:

○ yes
○ no
○ impossible to decide

Comments:

Figure 2: Screenshot of the annotation interface on Amazon's Mechanical Turk

We divided the data into training, dev, and test sets, using a 80-10-10 split. All results reported here were obtained on the test set. Tuning and development was only carried out on the dev set.

2.2 Vector Representation of Words

The same word may occur in a literal and a metaphorical usage. Lexical information alone is

[3]A similar assumption can be used to detect the literal/non-literal uses of idioms (Fazly et al., 2009).

[4]`http://www.metaphorlist.com` and `http://www.macmillandictionaryblog.com`

[5]While this is somewhat imprecise and not always easy to decide, it proved to be a viable strategy for untrained annotators.

*A **bright** idea.*

“ Peter is the **bright** , sympathetic guy when you ’re doing a deal , ” says one agent .	*yes*
Below he could see the **bright** torches lighting the riverbank .	*no*
Her **bright** eyes were twinkling .	*yes*
Washed , they came out surprisingly clear and **bright** .	*no*

Figure 1: Examples of a metaphor seed, the matching Brown sentences, and their annotations

thus probably not very helpful. However, we would like to capture semantic aspects of the word and represent it in an expressive way. We use the existing vector representation SENNA (Collobert et al., 2011) which is derived from contextual similarity. In it, semantically similar words are represented by similar vectors, without us having to define similarity or looking at the word itself. In initial tests, these vectors performed better than binary vectors straightforwardly derived from features of the word in context.

2.3 Constructing Trees

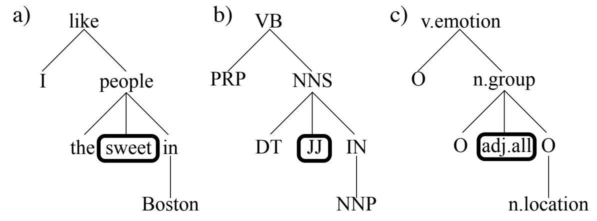


Figure 3: Graphic demonstration of our approach. a) dependency tree over words, with node of interest labeled. b) as POS representation. c) as supersense representation

The intuition behind our approach is that metaphorical use differs from literal use in certain syntactic relations. For example, the only difference between the two sentences “I like the sweet people in Boston” and “I like the sweet pies in Boston” is the head of “sweet”. Our assumption is that—given enough examples—certain patterns emerge (e.g., that “sweet” in combination with food nouns is literal, but is metaphorical if governed by a noun denoting people).

We assume that these patterns occur on different levels, and mainly between syntactically related words. We thus need a data representation to capture these patterns. We borrow its structure from dependency trees, and the different levels from various annotations. We parse the input sentence with the FANSE parser (Tratz and Hovy, 2011)[6]. It provides the dependency structure, POS tags, and other information.

To construct the different tree representations, we replace each node in the tree with its word, lemma, POS tag, dependency label, or supersense (the WordNet lexicographer name of the word’s first sense (Fellbaum, 1998)), and mark the word in question with a special node. See Figure 3 for a graphical representation. These trees are used *in addition* to the vectors.

This approach is similar to the ones described in (Moschitti et al., 2006; Qian et al., 2008; Hovy et al., 2012).

2.4 Classification Models

A tree kernel is simply a similarity matrix over tree instances. It computes the similarity between two trees T_1, T_2 based on the number of shared subtrees.

We want to make use of the information encoded in the different tree representations during classification, i.e., a forest of tree kernels. We thus combine the contributions of the individual tree representation kernels via addition. We use kernels over the lemma, POS tag, and supersense tree representations, the combination which performed best on the dev set in terms of accuracy.

We use the SVMlight TK implementation by Moschitti (2006).[7] We left most parameters set to default values, but tuned the weight of the contribution of the trees and the cost factor on the dev set. We set the multiplicative constant for the trees to 2.0, and the cost factor for errors on positive examples to 1.7.

[6]`http://www.isi.edu/publications/licensed-sw/fanseparser/index.html`

[7]`http://disi.unitn.it/moschitti/Tree-Kernel.htm`

If we assume any word can be used metaphorically, we ultimately want to label every word in a sentence, so we also evaluate a sequential model, in this case a CRF. We use CRFsuite (Okazaki, 2007)[8] to implement the CRF, and run it with averaged perceptron. While the CRF produces labels for *every* word, we only evaluate on the words that were annotated in our corpus (to make it maximally comparable), and use the same representations (lemma, POS and SST) of the word and its parent as features as we did for the SVM. Training method and feature selection were again tuned on the dev set to maximize accuracy.

3 Experiments

system	**acc**	**P**	**R**	**F1**
BL_{all}	0.49	0.49	1.0	0.66
$\text{BL}_{most\ freq.\ class}$	0.70	0.66	0.65	0.65
CRF	0.69*	**0.74***	0.50	0.59
$\text{SVM}_{vector-only}$	0.70*	0.63*	0.80	0.71
SVM_{+tree}	**0.75***	0.70*	**0.80**	**0.75***

Table 1: Accuracy, precision, recall, and F1 for various systems on the held-out test set. Values significantly better than baseline at $p < .02$ are marked * (two-tailed t-test).

We compare the performance of two baselines, the CRF model, vanilla SVM, and SVM with tree kernels and report accuracy, precision, recall, and F1 (Table 1).

The first baseline (BL_{all}) labels every instance as metaphor. Its accuracy and precision reflect the metaphor ratio in the data, and it naturally achieves perfect recall. This is a rather indiscriminate approach and not very viable in practice, so we also apply a more realistic baseline, labeling each word with the class it received most often in the training data ($\text{BL}_{most\ freq.\ class}$). This is essentially like assuming that every word has a default class. Accuracy and precision for this baseline are much better, although recall naturally suffers.

The CRF improves in terms of accuracy and precision, but lacks the high recall the baseline has, resulting in a lower F1-score. It does yield the highest precision of all models, though. So while not capturing every metaphor in the data, it is usually correct if it does label a word as metaphor.

SVMlight allows us to evaluate the performance of a classification using *only* the vector representation ($\text{SVM}_{vector-only}$). This model achieves better accuracy and recall than the CRF, but is less precise. Accuracy is the same as for the most-frequent-class baseline, indicating that the vector-based SVM learns to associate a class with each lexical item. Once we add the tree kernels to the vector (SVM_{+tree}), we see considerable gains in accuracy and precision. This confirms our hypothesis that metaphors are not only a lexical phenomenon, but also a product of the context a word is used in. The contextual interplay with their dependencies creates patterns that can be exploited with tree kernels. We note that the SVM with tree kernels is the only system whose F1 significantly improves over the baseline (at $p < .02$).

Testing with one tree representation at a time, we found the various representations differ in terms of informativeness. Lemma, POS, and supersense performed better than lexemes or dependency labels (when evaluated on the dev set) and were thus used in the reported system. Combining more than one representation in the same tree to form compound leaves (e.g. lemma+POS, such as "man-NN") performed worse in all combinations tested. We omit further details here, since the combinatorics of these tests are large and yield only little insight.

Overall, our results are similar to comparable methods on balanced corpora, and we encourage the evaluation of other methods on our data set.

4 Related Work

There is plenty of research into metaphors. While many are mainly interested in their general properties (Shutova, 2010; Nayak, 2011), we focus on the ones that evaluate their results empirically.

Gedigian et al. (2006) use a similar approach to identify metaphors, but focus on frames. Their corpus is with about 900 instances relatively small. They improve over the majority baseline, but only report accuracy. Both their result and the baseline are in the 90s, which might be due to the high number of metaphors (about 90%). We use a larger,

[8] `http://www.chokkan.org/software/crfsuite/`

more balanced data set. Since accuracy can be uninformative in cases of unbalanced data sets, we also report precision, recall, and F1.

Krishnakumaran and Zhu (2007) also use semantic relations between syntactic dependencies as basis for their classification. They do not aim to distinguish literal and metaphorical use, but try to differentiate various types of metaphors. They use a corpus of about 1700 sentences containing different metaphors, and report a precision of 0.70, recall of 0.61 (F1 = 0.65), and accuracy of 0.58.

Birke and Sarkar (2006) and Birke and Sarkar (2007) present unsupervised and active learning approaches to classifying metaphorical and literal expressions, reporting F1 scores of 0.54 and 0.65, outperforming baseline approaches. Unfortunately, as they note themselves, their data set is “not large enough to [...] support learning using a supervised learning method” (Birke and Sarkar, 2007, 22), which prevents a direct comparison.

Similarly to our corpus construction, (Shutova et al., 2010) use bootstrapping from a small seed set. They use an unsupervised clustering approach to identify metaphors and report a precision of 0.79, beating the baseline system by a wide margin. Due to the focus on corpus construction, they cannot provide recall or F1. Their approach considers only pairs of a single verbs and nouns, while we allow for any syntactic combination.

Tree kernels have been applied to a wide variety of NLP tasks (Culotta and Sorensen, 2004; Moschitti et al., 2006; Qian et al., 2008; Hovy et al., 2012). They are specifically adept in capturing long-range syntactic relationships. In our case, we use them to detect anomalies in syntactic relations.

5 Conclusion

Under the hypothesis that the metaphorical use of a word creates unusual patterns with its dependencies, we presented the first tree-kernel based approach to metaphor identification. Syntactic dependencies allow us to capture those patterns at different levels of representations and identify metaphorical use more reliably than non-kernel methods. We outperform two baselines, a sequential model, and purely vector-based SVM approaches, and reach an F1 of 0.75. Our corpus is available for download at `http://www.edvisees.cs.cmu.edu/metaphordata.tar.gz` and we encourage the research community to evaluate other methods on it.

Acknowledgements

The authors would like to thank the reviewers for helping us clarify several points and giving constructive input that helped to improve the quality of this paper. This work was (in part) supported by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Defense US Army Research Laboratory contract number W911NF-12-C-0025. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DoD/ARL, or the U.S. Government.

References

Eric P.S. Baumer, James P. White, and Bill Tomlinson. 2010. Comparing semantic role labeling with typed dependency parsing in computational metaphor identification. In *Proceedings of the NAACL HLT 2010 Second Workshop on Computational Approaches to Linguistic Creativity*, pages 14–22. Association for Computational Linguistics.

Julia Birke and Anoop Sarkar. 2006. A clustering approach for the nearly unsupervised recognition of nonliteral language. In *Proceedings of EACL*, volume 6, pages 329–336.

Julia Birke and Anoop Sarkar. 2007. Active learning for the identification of nonliteral language. In *Proceedings of the Workshop on Computational Approaches to Figurative Language*, pages 21–28. Association for Computational Linguistics.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Aron Culotta and Jeffrey Sorensen. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics.

Afsaneh Fazly, Paul Cook, and Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics*, 35(1):61–103.

Christiane Fellbaum. 1998. *WordNet: an electronic lexical database*. MIT Press USA.

Matt Gedigian, John Bryant, Srini Narayanan, and Branimir Ciric. 2006. Catching metaphors. In *Proceedings of the 3rd Workshop on Scalable Natural Language Understanding*, pages 41–48.

Claudio Giuliano, Alfio Massimiliano Gliozzo, and Carlo Strapparava. 2009. Kernel methods for minimally supervised wsd. *Computational Linguistics*, 35(4).

Dirk Hovy, James Fan, Alfio Gliozzo, Siddharth Patwardhan, and Christopher Welty. 2012. When Did that Happen? — Linking Events and Relations to Timestamps. In *Proceedings of EACL*.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning Whom to trust with MACE. In *Proceedings of NAACL HLT*.

Eric Iverson and Stephen Helmreich. 1991. Non-literal word sense identification through semantic network path schemata. In *Proceedings of the 29th annual meeting on Association for Computational Linguistics*, pages 343–344. Association for Computational Linguistics.

Saishuresh Krishnakumaran and Xiaojian Zhu. 2007. Hunting elusive metaphors using lexical resources. In *Proceedings of the Workshop on Computational approaches to Figurative Language*, pages 13–20. Association for Computational Linguistics.

George Lakoff and Mark Johnson. 1980. *Metaphors we live by*, volume 111. University of Chicago Press.

Zachary J. Mason. 2004. CorMet: a computational, corpus-based conventional metaphor extraction system. *Computational Linguistics*, 30(1):23–44.

Seyed A. Mirroshandel, Mahdy Khayyamian, and Gholamreza Ghassem-Sani. 2011. Syntactic tree kernels for event-time temporal relation learning. *Human Language Technology. Challenges for Computer Science and Linguistics*, pages 213–223.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2006. Tree kernel engineering for proposition re-ranking. *MLG 2006*, page 165.

Alessandro Moschitti. 2006. Making Tree Kernels Practical for Natural Language Learning. In *In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*.

Sushobhan Nayak. 2011. Towards a grounded model for ontological metaphors. In *Student Research Workshop*, pages 115–120.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of Conditional Random Fields (CRFs).

Longhua Qian, Guodong Zhou, Fang Kong, Qiaoming Zhu, and Peide Qian. 2008. Exploiting constituent dependencies for tree kernel-based semantic relation extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 697–704. Association for Computational Linguistics.

Ekaterina Shutova, Lin Sun, and Anna Korhonen. 2010. Metaphor identification using verb and noun clustering. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 1002–1010. Association for Computational Linguistics.

Ekaterina Shutova. 2010. Models of metaphor in nlp. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 688–697. Association for Computational Linguistics.

Stephen Tratz and Eduard Hovy. 2011. A fast, accurate, non-projective, semantically-enriched parser. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1257–1268. Association for Computational Linguistics.

Peter D. Turney, Yair Neuman, Dan Assaf, and Yohai Cohen. 2011. Literal and metaphorical sense identification through concrete and abstract context. In *Proceedings of the 2011 Conference on the Empirical Methods in Natural Language Processing*, pages 680–690.