Literal and Metaphorical Sense Identification through Concrete and Abstract Context

Peter D. Turney
Inst. for Info. Tech.
NRC Canada
Ottawa, Canada
peter.turney@nrc-cnrc.gc.ca

Yair Neuman
Dept. of Education
Ben-Gurion Univ.
Beer-Sheva, Israel
yneuman@bgu.ac.il

Dan Assaf
Dept. of Education
Ben-Gurion Univ.
Beer-Sheva, Israel
dan.assaf4@googlemail.com

Yohai Cohen
Gilasio Coding
Tel-Aviv, Israel
yohai@gilasio.com

Abstract

Metaphor is ubiquitous in text, even in highly technical text. Correct inference about textual entailment requires computers to distinguish the literal and metaphorical senses of a word. Past work has treated this problem as a classical word sense disambiguation task. In this paper, we take a new approach, based on research in cognitive linguistics that views metaphor as a method for transferring knowledge from a familiar, well-understood, or concrete domain to an unfamiliar, less understood, or more abstract domain. This view leads to the hypothesis that metaphorical word usage is correlated with the degree of abstractness of the word's context. We introduce an algorithm that uses this hypothesis to classify a word sense in a given context as either literal (denotative) or metaphorical (connotative). We evaluate this algorithm with a set of adjective-noun phrases (e.g., in *dark comedy*, the adjective *dark* is used metaphorically; in *dark hair*, it is used literally) and with the TroFi (Trope Finder) Example Base of literal and nonliteral usage for fifty verbs. We achieve state-of-the-art performance on both datasets.

1 Introduction

Metaphor is a natural consequence of our ability to reason by analogy (Gentner et al., 2001). It is so common in our daily language that we rarely notice it (Lakoff and Johnson, 1980). Identifying metaphorical word usage is important for reasoning about the implications of text.

Past work on the problem of distinguishing literal and metaphorical senses has approached it as a classical word sense disambiguation (WSD) task (Birke and Sarkar, 2006). Here, we take a different approach to the problem. Lakoff and Johnson (1980) argue that metaphor is a method for transferring knowledge from a concrete domain to an abstract domain. Therefore we hypothesize that the degree of abstractness in a word's context is correlated with the likelihood that the word is used metaphorically. This hypothesis is the basis for our algorithm for distinguishing literal and metaphorical senses.

Consider the following sentences:

L: He *shot down* my plane.
$\rightarrow C_1$: He *fired at* my plane.
$\nrightarrow A_1$: He *refuted* my plane.

M: He *shot down* my argument.
$\nrightarrow C_2$: He *fired at* my argument.
$\rightarrow A_2$: He *refuted* my argument.

The literal sense of *shot down* in L invokes knowledge from the domain of war. The metaphorical usage of *shot down* in M transfers knowledge from the concrete domain of war to the abstract domain of debate (Lakoff and Johnson, 1980).

The entailments of L and M depend on the intended senses of *shot down*. L entails the concrete *fired at* in C_1 (because, in order to literally shoot something down, you must first fire at it) but not the abstract *refuted* in A_1 (except perhaps as a joke). On the other hand, M entails *refuted* in A_2 but not *fired at* in C_2 (except perhaps as a novel metaphor).

In semiotics, Danesi (2003) argues that metaphor transfers *associations* from the source domain to the target domain. The metaphorical usage of *shot down* in M carries associations of violence and destruction that are not conveyed by A_2.

To make correct inferences about textual entailment, computers must be able to distinguish the literal and metaphorical senses of a word. Since recognizing textual entailment (RTE) is a core problem for NLP, with applications in Question Answering, Information Retrieval, Information Extraction, and Text Summarization, it follows that distinguishing literal and metaphorical senses is a problem for a wide variety of NLP tasks. The ability to recognize metaphorical word usage is a core requirement in the Intelligence Advanced Research Projects Activity (IARPA) Metaphor Program (Madrigal, 2011).[1]

Our approach to the problem of distinguishing literal and metaphorical senses is based on an algorithm for calculating the degree of abstractness of words. For instance, *plane* in L is rated 0.36396 (relatively concrete), whereas *argument* in M is rated 0.64617 (relatively abstract), which suggests that the verb *shot down* is used literally in L, whereas it is used metaphorically in M. Our abstractness rating algorithm is similar to Turney and Littman's (2003) algorithm for rating words according to their semantic orientation.

To classify a word usage as literal or metaphorical, based on the context, we use supervised learning with logistic regression. The abstractness rating algorithm is used to generate feature vectors from a word's context and training data is used to learn a logistic regression model that relates degrees of abstractness to the classes *literal* and *metaphorical*.

We evaluate our algorithm with three experiments. The first experiment involves one hundred adjective-noun phrases labeled *denotative* (literal) or *connotative* (metaphorical or nonliteral) by five annotators, according to the sense of the adjective.[2] For instance, *deep snow* is labeled *denotative* and *deep appreciation* is labeled *connotative*. The algorithm is able to predict the labels of the annotators with an average accuracy of 79%.

The next two experiments use the TroFi (Trope Finder) Example Base of literal and nonliteral usage for fifty verbs.[3] The fifty verbs occur in 3,737 sentences from The 1987-89 Wall Street Journal (WSJ) Corpus Release 1. In each sentence, the target verb is labeled *L* (literal) or *N* (nonliteral), according to the sense of the verb that is invoked by the sentence. A subset of twenty-five of the fifty verbs was used by Birke and Sarkar (2006).

In our second experiment, we duplicate the setup of Birke and Sarkar (2006) so that we can compare our results with theirs. In particular, a separate model is learned for each individual verb. We achieve an average f-score of 63.9%, compared to Birke and Sarkar's (2006) 64.9%.

In the third experiment, we train the algorithm on the twenty-five new verbs that were not used by Birke and Sarkar (2006) and then we test it on the old verbs. That is, the algorithm is tested with verbs that it has never seen before. The training verbs are merged to build a single model, instead of building a separate model for each individual verb. In this experiment, the average f-score is 68.1%.

The next section presents our algorithm for calculating the degree of abstractness of words. In Section 3, we review related work. The experiments are described in Section 4. We discuss the results of the experiments in Section 5 and conclude in Section 6.

2 Abstractness and Concreteness

Concrete words refer to things, events, and properties that we can perceive directly with our senses, such as *trees*, *walking*, and *red*.[4] Abstract words refer to ideas and concepts that are distant from immediate perception, such as *economics*, *calculating*, and *disputable*. In this section, we describe an algorithm that can automatically calculate a numerical rating of the degree of abstractness of a word on a scale from 0 (highly concrete) to 1 (highly abstract). For example, the algorithm rates *purvey* as 1, *donut* as 0, and *immodestly* as 0.5.

The algorithm is a variation of Turney and Littman's (2003) algorithm that rates words according to their semantic orientation. Positive semantic orientation indicates praise (*honest*, *intrepid*) and negative semantic orientation indicates criticism (*disturbing*, *superfluous*). The algorithm calculates the semantic orientation of a given word by comparing it to seven positive words and seven nega-

[1] See http://www.iarpa.gov/solicitations_metaphor.html.

[2] The labeled phrases are available from Yair Neuman.

[3] Available at http://www.cs.sfu.ca/ anoop/students/jbirke/.

[4] The word *red* has an abstract political sense, but our abstractness rating algorithm does not distinguish word senses. The more frequent concrete sense of *red* dominates, resulting in an abstractness rating of 0.24984 (highly concrete).

tive words that are used as paradigms of positive and negative semantic orientation:

Positive paradigm words: *good, nice, excellent, positive, fortunate, correct,* and *superior*.
Negative paradigm words: *bad, nasty, poor, negative, unfortunate, wrong,* and *inferior*.

Likewise, here we calculate the abstractness of a given word by comparing it to twenty abstract words and twenty concrete words that are used as paradigms of abstractness and concreteness.

Turney and Littman (2003) experimented with two measures of semantic similarity, pointwise mutual information (PMI) (Church and Hanks, 1989) and latent semantic analysis (LSA) (Landauer and Dumais, 1997). These measures take a pair of words as input and generate a numerical similarity rating as output. The semantic orientation of a given word is calculated as the sum of its similarity with the positive paradigm words minus the sum of its similarity with the negative paradigm words. Likewise, here we calculate the abstractness of a given word by the sum of its similarity with twenty abstract paradigm words minus the sum of its similarity with twenty concrete paradigm words. We then use a linear normalization to map the calculated abstractness value to range from 0 to 1.

Our algorithm for calculating abstractness uses a form of LSA to measure semantic similarity. This is described in detail in Section 2.1. Although Turney and Littman (2003) manually selected their fourteen paradigm words, here we use a supervised learning algorithm to choose our forty paradigm words, as explained in Section 2.2.

The MRC Psycholinguistic Database Machine Usable Dictionary (Coltheart, 1981) includes 4,295 words rated with degrees of abstractness by human subjects in psycholinguistic experiments.[5] The ratings range from 158 (highly abstract) to 670 (highly concrete). Table 1 gives some examples.

We used half of the 4,295 MRC words to train our supervised learning algorithm and the other half to validate the algorithm. On the testing set, the algorithm attains a correlation of 0.81 with the dictionary ratings. This indicates that the algorithm agrees well with human judgements of the degrees of abstractness of words.

[5]Available at http://ota.oucs.ox.ac.uk/headers/1054.xml.

Abstract Words	Rating	Concrete Words	Rating
as	158	ape	654
of	180	grasshopper	660
apt	183	tomato	662
however	186	milk	670

Table 1: Examples of abstract and concrete words from the MRC Dictionary (Coltheart, 1981).

2.1 Measuring Semantic Similarity

The variation of LSA that we use here is similar to Rapp's (2003) work. We modeled our similarity measure on Rapp's due to the high score of 92.5% that he achieved on a set of 80 multiple-choice synonym questions from the Test of English as a Foreign Language (TOEFL). The core idea is to represent words with vectors and calculate the similarity of two words by the cosine of the angle between the two corresponding vectors. The values of the elements in the vectors are derived from the frequencies of the words in a large corpus of text. This general approach is known as a Vector Space Model (VSM) of semantics (Salton et al., 1975).

We began with a corpus of 5×10^{10} words (280 gigabytes of plain text) gathered from university websites by a webcrawler.[6] We then indexed this corpus with the Wumpus search engine (Büttcher and Clarke, 2005).[7] We selected our vocabulary from the terms (words and phrases) in the WordNet lexicon.[8] By querying Wumpus, we obtained the frequency of each WordNet term in our corpus. We selected all WordNet terms with a frequency of 100 or more in our corpus. This resulted in a set of 114,501 terms. Next we used Wumpus to search for up to 10,000 phrases per term, where a phrase consists of the given term plus four words to the left of the term and four words to the right of the term. These phrases were used to build a word–context frequency matrix **F** with 114,501 rows and 139,246 columns. A row vector in **F** corresponds to a term in WordNet and the columns in **F** correspond to contexts (the words to the left and right of a given term in a given phrase) in which the term appeared.

The columns in **F** are unigrams (single words) in WordNet with a frequency of 100 or more in the corpus. A given unigram is represented by two

[6]Collected by Charles Clarke at the University of Waterloo.
[7]Wumpus is available at http://www.wumpus-search.org/.
[8]WordNet is available at http://wordnet.princeton.edu/.

columns, one marked *left* and one marked *right*. Suppose r is the term corresponding to the i-th row in $\mathbf{F}$ and c is the term corresponding to the j-th column in $\mathbf{F}$. Let c be marked *left*. Let f_{ij} be the cell in the i-th row and j-th column of $\mathbf{F}$. The numerical value in the cell f_{ij} is the number of phrases found by Wumpus in which the center term was r and c was the unigram closest to r on the left side of r. That is, f_{ij} is the frequency with which r was found in the context c in our corpus.

A new matrix $\mathbf{X}$, with the same number of rows and columns as in $\mathbf{F}$, was formed by calculating the Positive Pointwise Mutual Information (PPMI) of each cell in $\mathbf{F}$ (Turney and Pantel, 2010). The function of PPMI is to emphasize cells in which the frequency f_{ij} is statistically surprising, and hence particularly informative. This matrix was then smoothed with a truncated Singular Value Decomposition (SVD), which decomposes $\mathbf{X}$ into the product of three matrices $\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathsf{T}}$. Finally, the terms were represented by the matrix $\mathbf{U}_k \mathbf{\Sigma}_k^p$, which has 114,501 rows (one for each term) and k columns (one for each latent contextual factor). The semantic similarity of two terms is given by the cosine of the two corresponding rows in $\mathbf{U}_k \mathbf{\Sigma}_k^p$. For more detail, see Turney and Pantel (2010).

There are two parameters in $\mathbf{U}_k \mathbf{\Sigma}_k^p$ that need to be set. The parameter k controls the number of latent factors and the parameter p adjusts the weights of the factors, by raising the corresponding singular values in $\mathbf{\Sigma}_k^p$ to the power p. The parameter k is well-known in the literature on LSA, but p is less familiar. The use of p was suggested by Caron (2001). Based on our past experience, we set k to 1000 and p to 0.5. We did not explore any alternative settings of these parameters for measuring abstractness.

2.2 Measuring Abstractness

Now that we have $\mathbf{U}_k \mathbf{\Sigma}_k^p$, all we need in order to measure abstractness is some paradigm words. We used the MRC Psycholinguistic Database Machine Usable Dictionary (Coltheart, 1981) to guide our search for paradigm words. We split the 4,295 MRC words into 2,148 for training (searching for paradigm words) and 2,147 for testing (evaluation of the final set of paradigm words). We began with an empty set of paradigm words and added words from the 114,501 rows of $\mathbf{U}_k \mathbf{\Sigma}_k^p$, one word at a time, alternating between adding a word to the concrete paradigm words and then adding a word to the abstract paradigm words. At each step, we added the paradigm word that resulted in the highest Pearson correlation with the ratings of the training words. This is a form of greedy forward search without backtracking. We stopped the search after forty paradigm words were found, in order to prevent overfitting of the training data.

Table 2 shows the forty paradigm words and the order in which they were selected. At each step, the correlation increases on the training set, but eventually it must decrease on the testing set. After forty steps, the training set Pearson correlation was 0.8600. At this point, we stopped the search for paradigm words and calculated the testing set Pearson correlation, which was 0.8064. This shows a small amount of overfitting of the training data. The testing set Spearman correlation was 0.8216.

For another perspective on the performance of the algorithm, we measured its accuracy on the testing set, by creating a binary classification task from the testing data. We calculated the median of the ratings of the 2,147 words in the test set. Every word with an abstractness above the median was assigned to the class 1 and every word with an abstractness below the median was assigned to the class 0. We then used the algorithm to guess the rating of each word in the test set, calculated the median guess, and likewise assigned the guesses to classes 1 and 0. The guesses were 84.65% accurate.

After generating the paradigm words with the training set and evaluating them with the testing set, we then used them to assign abstractness ratings to every term in the matrix. The result of this is that we now have a set of 114,501 terms (words and phrases) with abstractness ratings ranging from 0 to 1.[9] Based on the testing set performance, we estimate these 114,501 ratings would have a Pearson correlation of 0.81 with human ratings and an accuracy of 85% on binary (*abstract* or *concrete*) classification.

We chose to limit the search to forty paradigm words based on our past experience with semantic orientation (Turney and Littman, 2003). To validate this choice, we allowed the algorithm to continue

[9]The 114,501 rated terms are available from Peter Turney.

Concrete Paradigm Words			Abstract Paradigm Words		
Order	Word	Correlation	Order	Word	Correlation
1	donut	0.4447	2	sense	0.6165
3	antlers	0.6582	4	indulgent	0.6973
5	aquarium	0.7150	6	bedevil	0.7383
7	nursemaid	0.7476	8	improbable	0.7590
9	pyrethrum	0.7658	10	purvey	0.7762
11	swallowwort	0.7815	12	pigheadedness	0.7884
13	strongbox	0.7920	14	ranging	0.7973
15	sixth-former	0.8009	16	quietus	0.8067
17	restharrow	0.8089	18	regularisation	0.8123
19	recorder	0.8148	20	creditably	0.8188
21	sawmill	0.8212	22	arcella	0.8248
23	vulval	0.8270	24	nonproductive	0.8299
25	tenrecidae	0.8316	26	couth	0.8340
27	hairpiece	0.8363	28	repulsion	0.8400
29	sturnus	0.8414	30	palsgrave	0.8438
31	gadiformes	0.8451	32	goof-proof	0.8469
33	cobbler	0.8481	34	meshuga	0.8503
35	bullet	0.8521	36	dillydally	0.8538
37	dioxin	0.8550	38	reliance	0.8570
39	usa	0.8585	40	lumbus	0.8600

Table 2: The forty paradigm words and the Pearson correlation on the training set.

searching until one hundred paradigm words were found. This resulted in a training set Pearson correlation of 0.8963, but the testing set correlation was only 0.8097, which shows a significant amount of overfitting of the training data. Although the testing set correlation is slightly higher with one hundred paradigm words, we chose to base the following experiments on the forty paradigm words, because the difference between 0.8064 and 0.8097 is not significant, and the gap between the training and testing correlation (0.8963 versus 0.8097) indicates a problematic amount of overfitting. Furthermore, the execution time of the algorithm increases as the paradigm set increases.

We generated abstractness ratings for a large vocabulary of 114,501 words in order to maximize the variety of text genres and the range of applications for which our list of abstractness ratings would be useful. As a consequence of this large vocabulary, many of the words in Table 2 are rare and obscure; however, the measure of quality of the algorithm is the correlation with the testing set (0.81), not the familiarity of the words in the table. We include the table here so that other researchers can experiment with these paragidm words. The table may give some insight into the internal functioning of the algorithm, but the main output of the algorithm is the list of 114,501 words with abstractness ratings, not the list of paradigm words in Table 2.

3 Related Work

Here we discuss related work on metaphor and then work on measuring abstractness. As far as we know, our approach is the first in computational linguistics to bring these two themes together, although the connection is well-known in cognitive linguistics (Lakoff and Johnson, 1980) and cognitive psychology (Gentner et al., 2001).

3.1 Metaphor

The most closely related work is Birke and Sarkar's (2006) research on distinguishing literal and nonliteral usage of verbs. A later paper (Birke and Sarkar, 2007) provides more detail on their active learning system, briefly mentioned in the earlier paper. Birke and Sarkar (2006; 2007) treat the problem as a classical word sense disambiguation task (Navigli, 2009). A model is learned for each verb independently from the other verbs. This approach cannot handle a new verb without additional training.

Hashimoto and Kawahara (2009) discuss work on a similar problem, distinguishing idiomatic usage from literal usage. They also approach this as a classical word sense disambiguation task. Idioms are somewhat different from metaphors, in that the meaning of an idiom (e.g., *kick the bucket*) is often difficult to derive from the meanings of the component words, unlike most metaphors.

Nissim and Markert (2003) use supervised learning to distinguish metonymic usage from literal usage. They take a classical WSD approach, learning a separate model for each target word. As with Birke and Sarkar (2006; 2007) and Hashimoto and Kawahara (2009), the core idea is to learn to classify word usage from similarity of context. Unlike these approaches, our algorithm generalizes beyond the specific semantic content of the context, paying attention only to the degrees of abstractness of the context.

Martin (1992) presents a knowledge-based approach to interpreting metaphors. This approach requires complex hand-coded rules, which are specific to a given domain (e.g., interpreting metaphorical questions from computer users, such as, "How can I *kill* a process?", in an online help system). The knowledge base cannot handle words that are not hand-coded in its rules and a new set of rules must be constructed for each new application domain.

Dolan (1995) describes an algorithm for extracting metaphors from a dictionary. Some suggestive examples are given, but the algorithm is not evaluated in any systematic way.

Mason (2004) takes a corpus-based approach to metaphor. His algorithm is based on a statistical approach to discovering the selectional restrictions of verbs. It then uses these restrictions to discover metaphorical mappings, such as, "Money flows like a liquid." Although the system can discover some metaphorical mappings, it was not designed to distinguish literal and metaphorical usages of words.

3.2 Abstractness

Changizi (2008) uses the hypernym hierarchy in WordNet to calculate the abstractness of a word. A word near the top of the hierarchy is considered abstract and a word near the bottom is considered concrete. It seems to us that the WordNet hypernym hierarchy captures the general–specific continuum, which might not be the same as the abstract–concrete continuum. It would be interesting to see how much correspondence there is between Changizi's measure of abstractness and the ratings in the MRC Psycholinguistic Database Machine Usable Dictionary (Coltheart, 1981). Also, note that adjectives and adverbs are outside of WordNet's hypernym hierarchy, and thus cannot be rated by Changizi's algorithm.

Xing et al. (2010) also use WordNet, but in a different way. They define the *concreteness* of a word sense (a WordNet synset) to be 1 if the given word sense is a hyponym of *physical entity* in the WordNet hypernym hierarchy; otherwise the *concreteness* is 0. We believe that, although physical entities are concrete, so are *redness* and *walking*, which are not hyponyms of *physical entity*. The category *physical entity* only partially captures concreteness.

4 Experiments

In the following experiments, we use the abstractness ratings of Section 2.2 to generate features for supervised machine learning. The learning algorithm we apply is logistic regression (Le Cessie and Van Houwelingen, 1992), as implemented in Weka (Witten and Frank, 2005).[10] In all experiments, we used the Weka parameter settings $R = 0.2$ (for robust ridge regression) and $M = -1$ (for unlimited iterations).

4.1 Adjectives

For this experiment, we selected five adjectives, *dark*, *deep*, *hard*, *sweet*, and *warm*. For each of the five adjectives, we identified twenty word pairs in which the first word is the adjective and the second word is a noun. These pairs were identified through the Corpus of Contemporary American English (COCA)[11] (Davies, 2009) by seeking the nouns that follow each adjective in the corpus and sorting the candidate adjective-noun pairs by frequency. We required a minimum pointwise mutual information (PMI) of 3 between the adjective and the noun. In some of the pairs, the adjective was

[10]Weka is available at http://www.cs.waikato.ac.nz/ml/weka/.

[11]Available at http://www.americancorpus.org/.

used in a denotative (literal) sense (*dark hair*) and in others it was used in a connotative (nonliteral) sense (*dark humor*). Table 3 gives some examples.

Adjective-Noun Pairs	Noun Abstractness
dark glasses	0.26826
dark chocolate	0.28211
dark energy	0.66207
dark mood	0.61858

Table 3: Some examples of adjective-noun pairs and the abstractness rating of the noun.

In this experiment, we used the abstractness rating of the noun (the context) to predict whether the adjective (the target) was used in a metaphorical or literal sense. Table 3 supports this idea, but it is easy to find counterexamples. Although *dark mood* is metaphorical, *bad mood* is literal. The difference is that *dark* has an abstractness rating of 0.43356 (relatively concrete), whereas *bad* has an abstractness rating of 0.63326 (relatively abstract). Metaphor results when a concrete word is imported into an abstract context (Lakoff and Johnson, 1980). Ideally, we should be comparing the abstractness of the target to the abstractness of the context. However, in our data, the target words are mostly concrete; thus we can focus on the context and ignore the target. We discuss this point further in Section 5.

Five judges, undergraduate students in psychology, were asked to judge whether the use of the adjective is a denotation or a connotation. The instructions were as follows:

> *Denotation* is the most direct or specific meaning of a word or expression while *connotation* is the meaning suggested by the word that goes beyond its literal meaning. For instance, the meaning of *bitter* is denotative in *bitter lemon* and connotative in *bitter relations*. In each of the following pairs, you will be asked to judge whether (1) the meaning of the first word is *denotative* or *connotative* and (2) to what extent it is denotative or connotative on a scale ranging from 1 to 4.

The judges were blind to the research hypothesis. Each judge received a booklet with the items organized by the groups of adjectives and presented in a random order. Overall, each subject was asked to evaluate one hundred pairs. Interjudge reliability was high, with Cronbach's Alpha equal to 0.95.

Our feature vectors for each pair contained only one element, the abstractness rating of the noun in the pair. We used logistic regression with ten-fold cross-validation to predict each judge's *denotative* and *connotative* labels. The results are summarized in Table 4. On average, we were able to predict a judge's labels with 79% accuracy.

Judge	Accuracy	Majority
1	0.730	0.590
2	0.810	0.570
3	0.840	0.560
4	0.790	0.510
5	0.780	0.520
Average	0.790	0.550

Table 4: The accuracy of logistic regression at predicting the labels of each judge.

Table 4 also shows the size of the majority class (the most common label) for each judge. For all of the judges, the accuracy was significantly greater than the size of the majority class (Fisher Exact test, 95% confidence level). The results support our hypothesis that the abstractness of the context is predictive of whether an adjective is used in a literal or metaphorical sense.

4.2 Known Verbs

For this experiment, we used the TroFi (Trope Finder) Example Base of literal and nonliteral usage for fifty verbs.[12] To compare our results with Birke and Sarkar's (2006) results, we use the same subset of twenty-five of the fifty verbs. These twenty-five verbs appear in 1,965 sentences, manually labeled *L* (literal) or *N* (nonliteral), according to the sense of the target verb. The verbs also appeared in some sentences labeled *U* (unannotated), but we ignored these sentences (although they could be useful for semi-supervised learning).

The label *nonliteral* is intended to be a broad category that includes *metaphorical* as a special case. Other types of nonliteral usage include *idiomatic* and *metonymical*, but it seems that most of the *nonliteral* cases in TroFi are in fact *metaphorical*, and

[12]Available at http://www.cs.sfu.ca/ anoop/students/jbirke/.

hence our hypothesis about the correlation of abstract context with metaphorical sense is appropriate for classifying the TroFi sentences.

Two examples of sentences from TroFi follow. Both contain the target verb *absorb*. The first sentence is *literal* and the second is *nonliteral*.

***L*:** An Energy Department spokesman says the sulfur dioxide might be simultaneously recoverable through the use of powdered limestone, which tends to *absorb* the sulfur.

***N*:** He said that MMWEC will have to *absorb* only \$4 million in additional annual costs now paid by the Vermont utilities.

To generate feature vectors for the sentences, we first applied the OpenNLP part-of-speech tagger to the sentences.[13] We then looked for each word in our list of 114,501 abstractness ratings (Section 2.2). If the word was not found in the list, we applied the Morpha morphological analyzer to identify the stem of the word (e.g., the stem of *managing* is *manage*) (Minnen et al., 2001).[14] We then looked for the stem in our list. If it was still not found, we skipped it.

For each sentence, we created a vector with five features:

1. the average abstractness ratings of all nouns, excluding proper nouns
2. the average abstractness ratings of all proper nouns
3. the average abstractness ratings of all verbs, excluding the target verb
4. the average abstractness ratings of all adjectives
5. the average abstractness ratings of all adverbs

When there were no words for a given part of speech, we set the average to a default value of 0.5. Two examples of feature vectors follow, corresponding to the two TroFi sentences above.

***L*:** $\langle 0.3873, 0.5397, 0.6375, 0.2641, 0.5835\rangle$
***N*:** $\langle 0.6120, 0.3726, 0.6699, 0.5612, 0.5000\rangle$

The intuition here is that the weight of each context word, in predicting the class of the target verb, may depend on the part of speech of the context word. We leave it to the logistic regression algorithm to determine the appropriate weighting, based on the training data. (See Table 7 in the next section.)

Following Birke and Sarkar's (2006) approach, we treated each group of sentences for a given target verb as a separate learning problem. For each verb, we used ten-fold cross-validation to learn and test logistic regression models. To measure the performance of the models, we used three different scores, macro-averaged accuracy and two forms of macro-averaged f-score.

Birke and Sarkar (2006) explain their scoring as follows:

> *Literal recall* is defined as *(correct literals in literal cluster / total correct literals)*. *Literal precision* is defined as *(correct literals in literal cluster / size of literal cluster)*. If there are no literals, *literal recall* is 100%; *literal precision* is 100% if there are no nonliterals in the literal cluster and 0% otherwise. The *f-score* is defined as *(2 · precision · recall) / (precision + recall)*. Nonliteral precision and recall are defined similarly. Average precision is the average of literal and nonliteral precision; similarly for average recall. For overall performance, we take the f-score of average precision and average recall.

The overall score is a macro-average, in which each verb has equal weight, regardless of how many sentences it appears in.

Every verb in TroFi has at least one *literal* usage and one *nonliteral* usage, so there is no issue with the definition of recall as 100% when there are no literals or no nonliterals. However, we believe that the definition of precision as 100% when no sentence is assigned to the literal or nonliteral cluster gives too high a score to the trivial algorithm of always guessing the majority class. The minority class will then always have a precision of 100%. Therefore we use a modified f-score in which the precision of a class is 0% if the algorithm never guesses that class. We refer to Birke and Sarkar's (2006) score as *f-score (0/0 = 1)* and to our own score as *f-score (0/0 = 0)*.

Table 5 summarizes our results. *Concrete-Abstract* refers to our own algorithm. *Birke-Sarkar*

[13]Available at http://incubator.apache.org/opennlp/.
[14]Available at http://www.informatics.susx.ac.uk/research/groups/nlp/carroll/morph.html.

refers to the best result reported by Birke and Sarkar (2006), using a form of active learning. *Majority Class* is the simple strategy of always guessing the majority class. *Probability Matching* is the strategy of randomly guessing each class with a probability equal to the size of the class.

Algorithm	Accuracy	F-score (0/0=0)	F-score (0/0=1)
Concrete-Abstract	0.734	0.631	0.639
Birke-Sarkar	*NA*	*NA*	0.649
Majority Class	**0.697**	**0.408**	0.629
Probability Matching	**0.605**	**0.500**	**0.500**

Table 5: The performance with known verbs.

We used a paired t-test to evaluate the statistical significance of the results in Table 5. The numbers are in bold font when the performance of an algorithm is significantly below the performance of Concrete-Abstract. In no case is any score significantly above the performance of Concrete-Abstract, at the 95% confidence level. *NA* indicates scores that were not calculated by Birke and Sarkar (2006).

4.3 Unknown Verbs

For the final experiment, we again used the TroFi Example Base, but with a different experimental setup. Instead of ten-fold cross-validation, we used the twenty-five verbs in Birke and Sarkar (2006) for testing (we call these the *old* verbs) and the other twenty-five verbs (the *new* verbs) for training. The twenty-five old (testing) verbs appear in 1,965 sentences and the twenty-five new (training) verbs appear in 1,772 sentences. For this experiment, we no longer learn a separate logistic regression model for each verb. All of the 1,772 training sentences are used together to learn a single logistic regression model, which is then evaluated on the testing sentences.

Table 6 summarizes our results. Since the testing set is exactly the same as in Section 4.2, we can compare the performance directly with the performance in the preceding section and with Birke and Sarkar's (2006) results.

Again, we used a paired t-test to evaluate the statistical significance of the results in Table 6. The numbers are in bold font when the performance of an algorithm is significantly below the performance of Concrete-Abstract. In no case is any score significantly above the performance of Concrete-Abstract, at the 95% confidence level.

Algorithm	Accuracy	F-score (0/0=0)	F-score (0/0=1)
Concrete-Abstract	0.686	0.673	0.681
Birke-Sarkar	*NA*	*NA*	0.649
Majority Class	0.697	**0.408**	**0.629**
Probability Matching	**0.605**	**0.500**	**0.500**

Table 6: The performance with unknown verbs.

Table 7 shows the coefficients in the logistic regression model that was learned on the training data. The items numbered from 1 to 5 are the five features described in Section 4.2. The sixth item is the constant term in the regression equation. We see that the abstractness of the nouns (excluding proper nouns) has the largest weight in predicting whether the target verb is in class *N*.

	Feature	Coefficient
1	AvgNounAbs	11.4117
2	AvgPropAbs	0.7250
3	AvgVerbAbs	-0.5528
4	AvgAdjAbs	1.1478
5	AvgAdvAbs	-0.2013
6	Intercept	-5.9436

Table 7: The logistic regression coefficients for class *N*.

5 Discussion

It is a strength of our approach that it can classify verbs that it has never seen before, as we see in Section 4.3. The feature vectors in all three experiments are based only on the context; the target adjective or verb is not used in the vectors. This avoids the need for gathering training data on every verb or adjective for which we want to determine whether it is being used metaphorically or literally, since the algorithm is not sensitive to the specific target word.

On the other hand, the performance might improve if the target word were included in the feature vectors. If metaphor is a method for transferring knowledge from concrete domains to abstract domains, then it follows that highly abstract target words will tend to be used literally in most contexts. For instance, the highly abstract verb *epitomize* (with an abstractness rating of 0.85861) is perhaps almost always used in a literal sense. Therefore it would seem that the abstractness rating of the target word could be a useful clue for determining whether the sense is literal or metaphorical.

We experimented with including the abstractness rating of the target word as a feature, but the impact on performance was not significant for either the adjectives or the verbs. We hypothesize that this may be due to the relatively narrow range in the abstractness of the adjectives and verbs in our data. The abstractness ratings of the adjectives vary from 0.43356 for *dark* to 0.56637 for *hard*. The abstractness ratings of the fifty verbs range from 0.28756 for *plant* to 0.71628 for *lend*, but 80% of the verbs lie in the range from 0.41879 for *fly* to 0.59912 for *rest*. It seems possible that the abstractness rating of the target word would be useful with a dataset in which the target's abstractness varied substantially.

In future work, we would like to gather data for target words with a wider range of abstractness. We expect that such data would show some benefit to including information on the abstractness of the target word in the feature vector.

We also expect that a hybrid of classical word sense disambiguation, such as Birke and Sarkar's (2006) algorithm, with abstractness ratings would perform better than either approach alone. Abstractness may provide a good rough estimate of whether a word usage is literal or metaphorical, but it seems likely that knowledge of the specific target word in question will be required for a highly precise answer. This is another worthwhile topic for future research.

Currently there is no algorithm that identifies what kind of concepts and relations are grafted from the source domain to the target domain by metaphorical inference. The algorithm presented in this paper may be used within a constraints-based model of metaphor (Neuman and Nave, 2009) to address this challenge.

Recently there has been some interest in *visualness*, *picturability*, and *imagability*, the degree to which a word is associated with visual imagery (Deschacht and Moens, 2007). Although Xing et al. (2010) use the term *concreteness* in their work, their research is concerned with predicting the difficulty of queries for image retrieval. It could be argued that Xing et al. should be trying to capture *imagability*, not *concreteness*.

The MRC Psycholinguistic Database (Coltheart, 1981) includes words rated for *imagability*. Our algorithm for rating the abstractness of words (Section 2) could easily be trained with the MRC imagability ratings instead of the abstractness ratings. In future work, it would be interesting to evaluate imagability ratings on the TroFi Example Base. It would also be worthwhile to see whether our algorithm can be adapted for image retrieval (Xing et al., 2010) and image annotation (Deschacht and Moens, 2007).

6 Conclusion

Metaphor is ubiquitous, yet recognizing textual entailment is a challenge when words are used metaphorically. An algorithm for distinguishing metaphorical and literal senses of a word will facilitate correct textual inference, which will improve the many NLP applications that depend on textual inference.

We have introduced a new algorithm for measuring the degree of abstractness of a word. Inspired by research in cognitive linguistics (Lakoff and Johnson, 1980), we hypothesize that the degree of abstractness of the context in which a given word appears is predictive of whether the word is used in a metaphorical or literal sense. This hypothesis is supported by three experiments.

A strength of this approach to the problem of distinguishing metaphorical and literal senses is that it readily generalizes to new words, outside of the training data. We do not claim that abstractness is a complete solution to the problem, but it may be a valuable component in any practical system for processing metaphorical text.

Acknowledgments

Part of the work of Yair Neuman, Dan Assaf, and Yohai Cohen has been supported by a grant from the Israel Ministry of Defense. Thanks to the EMNLP reviewers for their helpful comments.

References

Julia Birke and Anoop Sarkar. 2006. A clustering approach for the nearly unsupervised recognition of nonliteral language. In *Proceedings of the 11th Conference of the European Chapter of the Association for*

Computational Linguistics (EACL 2006), 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 at HLT/NAACL-07*, pages 21–28.

Stefan Büttcher and Charles Clarke. 2005. Efficiency vs. effectiveness in terabyte-scale information retrieval. In *Proceedings of the 14th Text REtrieval Conference (TREC 2005)*, Gaithersburg, MD.

John Caron. 2001. Experiments with LSA scoring: Optimal rank and basis. In *Proceedings of the SIAM Computational Information Retrieval Workshop*, pages 157–169, Raleigh, NC.

Mark Changizi. 2008. Economically organized hierarchies in WordNet and the Oxford English Dictionary. *Cognitive Systems Research*, 9(3):214–228.

Kenneth Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Conference of the Association of Computational Linguistics*, pages 76–83, Vancouver, British Columbia.

Max Coltheart. 1981. The MRC psycholinguistic database. *Quarterly Journal of Experimental Psychology*, 33A(4):497–505.

Marcel Danesi. 2003. Metaphorical "networks" and verbal communication: A semiotic perspective on human discourse. *Sign Systems Studies*, 31:341–363.

Mark Davies. 2009. The 385+ million word Corpus of Contemporary American English (1990–2008+): Design, architecture, and linguistic insights. *International Journal of Corpus Linguistics*, 14(2):159–190.

Koen Deschacht and Marie-Francine Moens. 2007. Text analysis for automatic image annotation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 1000–1007.

William B. Dolan. 1995. Metaphor as an emergent property of machine-readable dictionaries. In *Proceedings of the AAAI 1995 Spring Symposium Series: Representation and Acquisition of Lexical Knowledge: Polysemy, Ambiguity and Generativity*, pages 27–32.

Dedre Gentner, Brian F. Bowdle, Phillip Wolff, and Consuelo Boronat. 2001. Metaphor is like analogy. In D. Gentner, K. J. Holyoak, and B. N. Kokinov, editors, *The analogical mind: Perspectives from Cognitive Science*, pages 199–253. MIT Press, Cambridge, MA.

Chikara Hashimoto and Daisuke Kawahara. 2009. Compilation of an idiom example database for supervised idiom identification. *Language Resources and Evaluation*, 43(4):355–384.

George Lakoff and Mark Johnson. 1980. *Metaphors We Live By*. University Of Chicago Press, Chicago, IL.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240.

Saskia Le Cessie and J.C. Van Houwelingen. 1992. Ridge estimators in logistic regression. *Applied Statistics*, 41(1):191–201.

Alexis Madrigal. 2011. Why are spy researchers building a 'Metaphor Program'? *The Atlantic*, May 25.

James H. Martin. 1992. Computer understanding of conventional metaphoric language. *Cognitive Science*, 16(2):233–270.

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

Guido Minnen, John Carroll, and Darren Pearce. 2001. Applied morphological processing of English. *Natural Language Engineering*, 7(3):207–223.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys*, 41(2):1–69.

Yair Neuman and Ophir Nave. 2009. Metaphor-based meaning excavation. *Information Sciences*, 179:2719–2728.

Malvina Nissim and Katja Markert. 2003. Syntactic features and word similarity for supervised metonymy resolution. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*, pages 56–63, Sapporo, Japan.

Reinhard Rapp. 2003. Word sense discovery based on sense descriptor dissimilarity. In *Proceedings of the Ninth Machine Translation Summit*, pages 315–322.

Gerard Salton, Anita Wong, and Chung-Shu Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Peter D. Turney and Michael L. Littman. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4):315–346.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37:141–188.

Ian H. Witten and Eibe Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, San Francisco.

Xing Xing, Yi Zhang, and Mei Han. 2010. Query difficulty prediction for contextual image retrieval. In *Advances in Information Retrieval*, volume 5993 of *Lecture Notes in Computer Science*, pages 581–585. Springer.