Modeling Newswire Events using Neural Networks for Anomaly Detection

Pradeep Dasigi
Language Technologies Institute
5000 Forbes Avenue
Pittsburgh, PA 15213
USA
pdasigi@cs.cmu.edu

Eduard Hovy
Language Technologies Institute
5000 Forbes Avenue
Pittsburgh, PA 15213
USA
hovy@cmu.edu

Abstract

Automatically identifying anomalous newswire events is a hard problem. We discuss the complexity of the problem and introduce a novel technique to model events based on recursive neural networks to represent events as composition of their semantic arguments. Our model learns to differentiate between normal and anomalous events. We model anomaly detection as a binary classification problem and show that the model learns useful features to classify anomaly. We use headlines from the weird news category publicly available on newswire websites to extract anomalous training examples and those from Gigaword as normal examples. We evaluate the classifier on human annotated data and obtain an accuracy of 65.44%. We also show that our model is at least as competent as the least competent human annotator in anomaly detection.

1 Introduction

Understanding events is a fundamental prerequisite for deeper semantic analysis of language. We introduce the problem of automatic anomalous event detection in this paper and propose a novel event model that can learn to differentiate between normal and anomalous events. We generally define anomalous events as those that are unusual compared to the general state of affairs and might invoke surprise when reported. For example, given the event mention in the following sentence

Man recovering after being shot by his dog.

one might think it is strange because *dogs* are not expected to shoot *men*. But the mentions

Man recovering after being shot by cops.

Man recovering after being bitten by a dog.

are not as unusual as the previous one. While all three sentences are equally valid syntactically, and it is not unclear what any of them means, it is our knowledge about the role fillers —both individually and specifically in combination— that enables us to differentiate between normal and anomalous events. Hence we hypothesize that *anomaly is a result of unexpected or unusual combination of semantic role fillers*. Given this idea, an automatic anomaly detection algorithm has to encode the goodness of semantic role filler coherence.

It has to be noted that event level anomaly is not the same as semantic incoherence. An event constructed by randomly choosing words to form each of the semantic arguments is not anomalous since we cannot argue whether the event is normal or anomalous when it is unclear what the event means. Hence, we define anomalous events to be the sub class of those that are semantically coherent, but are unusual only based on real world knowledge.

Automatic anomalous event detection is a hard problem since determining what a good combination of role fillers requires deep semantic and pragmatic knowledge. Moreover, manual judgment of anomaly itself may be difficult and people often may not agree with each other in this regard. We describe the difficulty in human judgment in greater detail in Section 4.4. Automatic detection of anomaly requires encoding complex information, which has to be composed from the semantics of the individual words in the sentence. A fundamental problem in doing so is the sparsity in semantic space due to the discrete representations of meaning of words.

In this paper, we describe an attempt to model newswire events as a composition of the predicate with its semantic arguments. Our approach is based on the recent models used for semantic composition using recursive neural networks (RNN). It has been previously shown by Socher et al. (2010) and Socher et al. (2013b) among others that RNN can effectively deal with sparsity in semantic space by representing meaning at a higher level of abstraction than the surface forms of words, and thus being able to learn more general patterns. These models are very relevant to modeling event semantics because the sparsity problem ranges from polysemy and synonymy at the lexical semantic level to entity and event co-reference at the discourse level.

2 Background

2.1 Selectional Preference and Thematic Fit

Selectional preference, a notion introduced by Wilks (1973), refers to the phenomenon of the predicate and the fillers of its arguments affecting the likelihood of fillers of other arguments. Thus the idea is that predicate and the role fillers "prefer" some fillers for other roles. For example, given that the predicate is *writes*, the agent *author* prefers the patient *book*, while the agent *programmer* prefers the patient *code*. This idea is used by Elman (2009), and is very similar to the role-filler composition that we use for anomaly detection.

Erk et al. (2010) also model selectional preferences using vector spaces. They measure the goodness of the fit of a noun with a verb in terms of the similarity between the vector of the noun and some "exemplar" nouns taken by the verb in the same argument role. Baroni and Lenci (2010) also measure selectional preference similarly, but instead of exemplar nouns, they calculate a prototype vector for that role based on the vectors of the most common nouns occurring in that role for the given verb. Lenci (2011) builds on this work and models the phenomenon that the expectations of the verb or its role-fillers change dynamically given other role fillers.

2.2 Recursive Neural Networks

Recursive Neural Networks (RNN), first introduced by Goller and Kuchler (1996), are multilayer neural network models used for efficient processing of structured objects of arbitrary shape. These have been successfully used for modeling semantics of sentences of arbitrary length by Socher et al. (2010), for sentiment analysis by Socher et al. (2013b), for syntactic parsing by Socher et al. (2013a) and for learning morphologically aware word representations by Luong et al. (2013). RNN are attractive because they can encode compositions of meaning guided by syntax or some other linguistic structure known a priori. Moreover, they provide flexibility in terms of learning composition weights based on supervised or unsupervised objectives. Consequently RNN learn feature representations depending on the task. Hence, this is a good choice for modeling event composition.

In its simplest form, an RNN processes information backed by a Directed Acyclic Graph (DAG), where each node represents a neural network with the same parameters. The output produced at each intermediate step of encoding usually has the same dimensionality as each of the inputs, hence RNN projects the representation of a structure of arbitrary length into the same space as the inputs. This property is what makes RNN recursive. An example RNN with a binary DAG (tree) structure is shown in Figure 1. The activation from each neural network node is

$$c = g(y_1 \| y_2) = Sg(W(y_1 \| y_2) + b)$$

where $\|$ represents concatenation of vector representations of the inputs, $y_1, y_2 \in \mathbb{R}^{n \times 1}$ are the inputs, $W \in \mathbb{R}^{n \times 2n}$ is the composition weight matrix and $b \in \mathbb{R}^{n \times 1}$ is the bias. Sg is a element wise sigmoid

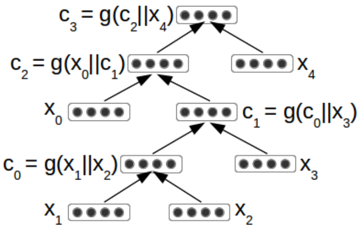

Figure 1: Example of a Recursive Neural Network backed by a binary tree

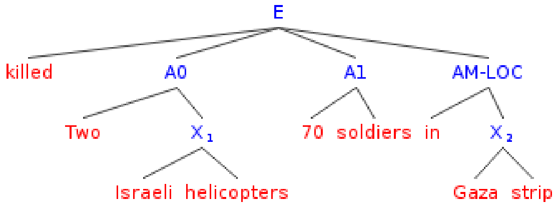

Figure 2: Example of an event tree

function. Apart from encoding the composition, RNN also produce a score of composition

$$s = S^{\intercal}c$$

where $S \in \mathbb{R}^{n\times 1}$ is a scoring operator and s is a score that shows how good the composition is. (Collobert et al., 2011) take an unsupervised approach to training RNN for semantic composition based on the contrastive estimation technique proposed by (Smith and Eisner, 2005) and assuming that any word and its context is a positive example and a random word in the same context is a negative training example. (Socher et al., 2013b) among others use a supervised objective that is based on the label error at the topmost node in the RNN. The parameters of the simplest model are W, b and S. For representation learning, the inputs x_i are also made parameters. Goller and Kuchler (1996) propose Backpropagation through structure (BPTS), that respects the underlying DAG structure during backpropagation of gradients.

3 Neural Event Model

We define an event as the pair $(V, \mathbf{A})$, where V is the predicate or a semantic verb[1], and $\mathbf{A}$ is the set of its semantic arguments like agent, patient, time, location, so on. Our aim is to obtain a vector representation of the event that is composed from representations of individual words, while explicitly guided by the semantic role structure. This representation can be understood as an embedding of the event in an event space.

Neural Event Model (NEM) is a kind of RNN that is guided by a tree representation of events like the one shown in Figure 2. The edges connected to the root of the tree correspond to the predicate and its semantic roles (arguments). All the other edges form binary sub-trees of arguments. NEM is a supervised model that learns to differentiate between anomalous and normal events by classifying the event embeddings. The inputs to NEM are the semantic arguments, and the representations of words in each argument. We recursively compose the words in each argument to obtain argument level representations, which are then composed to obtain an event embedding.

Intra-argument composition (called argument composition henceforth) is unsupervised, and we use contrastive estimation to learn the parameters. The structure of the binary tree backing argument composition is determined dynamically, composing at each stage the two nodes which give the best composition

[1] By semantic verb, we mean an action word whose syntactic category is not necessarily a verb. For example, in *Terrorist attacks on the World Trade Center..*, *attacks* is not a verb but is still an action word.

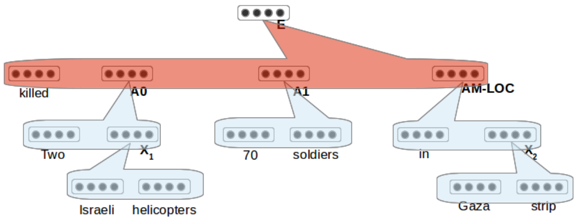

Figure 3: Neural Event Model: Encoding

score. Inter-argument composition (called event composition henceforth) is supervised and we use label error to learn the parameters. Figure 3 shows how NEM encodes the event shown in Figure 2. The blue boxes show argument composition and the red box shows event composition.

3.1 Training

NEM is trained in two phases. The first, argument composition, is unsupervised while the second, event composition, is supervised.

3.1.1 Argument Composition

An argument composition node takes inputs of dimensionality $2n$ and produces an composed output representation of dimensionality n and a composition score. Accordingly, we define the node in terms of the parameters $\theta_{arg} = \{W_{arg} \in \mathbb{R}^{n \times 2n}; b_{arg}, S_{arg} \in \mathbb{R}^{n \times 1}; V\}$ where W_{arg}, b_{arg} and S_{arg} are the composition weight, bias and the scoring operators respectively as described previously, and V is the set of representations of all the words in the vocabulary. All nodes performing argument composition use the same parameters. Training is done in contrastive estimation fashion and the objective is

$$\arg\min_{\theta_{arg}} J_{arg} = \arg\min_{\theta_{arg}} max(0, 1 - s + s_c)$$

where s is the score of the composition of the entire argument produced by the root node of the argument, and s_c is the score produced by randomly replacing one of the words in the argument at a time. The structure of the binary tree backing each argument is determined dynamically. This is done by starting with leaf nodes in the tree for each of the words in the argument, comparing the composition scores of every pair of adjacent leaf nodes, and actually composing the pair that gives the highest score, which gives a new node. The process is repeated until we build a complete binary tree for each argument.

3.1.2 Event Composition

Event composition takes argument representations and produces the event representation and label indicating whether the event is normal or anomalous. We define the event composition node in terms of the parameters $\theta_{event} = \{W_{event} \in \mathbb{R}^{n \times kn}; b_{event}, L_{event} \in \mathbb{R}^{n \times 1}\}$ where k is the number of semantic arguments per event. L_{event} is the label operator. The objective of this phase is

$$\arg\min_{\theta_{event}} J_{event} = \arg\min_{\theta_{event}} \left(-l \log h(e) + (1 - l) \log(1 - h(e))\right)$$

where l is the reference binary label indicating whether the event is normal or anomalous, e is the event representation and $h(e)$ is the output of the logistic function. Concretely,

$$h(e) = \frac{1}{1 + e^{-L_{event}^{\intercal} e}}$$

We implement the functions and perform stochastic gradient descent using Theano (Bergstra et al., 2010).

4 Experiments

4.1 Event Extraction

We extract events by running the Semantic Role Labeling (SRL) tool in SENNA (Collobert et al., 2011). SENNA uses PropBank (Palmer et al., 2005) style semantic tags. We consider only the roles *A0*, *A1*, *AM-TMP* and *AM-LOC* as the arguments of our events[2]. For example, the event in the tree shown in Figure 2 is extracted from the sentence

> *Two Israeli helicopters killed 70 soldiers in Gaza strip.*

and SENNA identifies the following as the semantic roles

> ***verb:****killed* ***A0:****Two Israeli helicopters* ***A1:****70 soldiers* ***AM-LOC:****in Gaza strip*

4.2 Data

Since the second phase of training NEM is supervised, we need newswire events that are normal and those that are anomalous. We crawl 3684 "weird news" headlines available publicly on the website of NBC news[3], such as the following:

- *India weaponizes world's hottest chili.*
- *Man recovering after being shot by his dog.*
- *Thai snake charmer puckers up to 19 cobras.*

We assume that the events extracted from this source, called NBC Weird Events (NWE) henceforth, are anomalous for training. NWE contains 4271 events extracted using SENNA's SRL. We use 3771 of those events as our negative training data, and the remaining for testing. Similarly, we extract events also from headlines in the AFE section of Gigaword, called Gigaword Events (GWE) henceforth. We assume these events are normal. To use as positive examples for training event composition, we sample roughly the same number of events from GWE as our negative examples from NWE. It has to be noted that each headline may contain multiple events and some may not contain events at all.

For argument composition, we use about 100k whole sentences from AFE headlines and the weird news headlines from which NWE are extracted. Since we are training argument composition, we do not use the event structure in the first phase. It has to be noted that all our training data are easily available and do not require any human annotation.

We test the performance of NEM on 1003 events which are not part of the training dataset. These events are sampled with equal probabilities from NWE and GWE and are human annotated for anomaly. Section 4.4 has details of the annotation task.

4.3 Word Vector Initialization

We initialize the vector representations of the words in our vocabulary using the embeddings available in SENNA 3.0 (Collobert et al., 2011) if available, and randomly if not. For event composition, if the event does not have a specific role filler, we input a zero vector for the role.

4.4 Annotation

We post the annotation of the test set containing 1003 events as Human Intelligence Tasks (HIT) on Amazon Mechanical Turk (AMT). We break the task into 20 HITs and ask the workers to select one of the four options - *highly unusual*, *strange*, *normal* and *cannot say* for each event. We ask them to select *highly unusual* when the event seems too strange to be true, *strange* if it seems unusual but still plausible, and *cannot say* only if the information present in the event is not sufficient to make a decision. We present each event along with the original headline and the semantic arguments. Along with marking

[2] These four types cover about 85% of all arguments in our training and test datasets.

[3] `http://www.nbcnews.com/html/msnbc/3027113/3032524/4429950/4429950_1.html`

Total number of annotators	22
Normal annotations	56.3%
Strange annotations	28.6%
Highly unusual annotations	10.3%
Cannot Say annotations	4.8%
Avg. events annotated per worker	344
4-way Inter annotator agreement (α)	0.34
3-way Inter annotator agreement (α)	0.56

Table 1: Annotation Statistics

one of the four options above, if an event is *strange* or *highly unusual*, we ask the annotators to select the parts of the headline that make it so. Since there can be multiple events in the headline, the annotators decision regarding the parts of the sentence that cause anomaly help us identify which particular event in the headline is anomalous.

Table 1 shows some statistics of the annotation task. We compute the Inter Annotator Agreement (IAA) in terms of Kripendorff's alpha (Krippendorff, 1980). The advantage of using this measure instead of the more popular Kappa is that the former can deal with missing information, which is the case with our task since annotators work on different overlapping subsets of the test set. The 4-way IAA shown in the table corresponds to agreement over the original 4-way decision (including *cannot say*) while the 3-way IAA is measured after merging the *highly unusual* and *strange* decisions.

Additionally we use MACE (Hovy et al., 2013) to assess the quality of annotation. MACE models the annotation task as a generative process of producing the observed labels conditioned on the true labels and the competence of the annotators, and predicts both the latent variables. The average of competence of annotators, a value that ranges from 0 to 1, for our task is 0.49 for the 4-way decision and 0.59 for the 3-way decision.

We generate true label predictions produced by MACE, discard the events for which the prediction remains to be *cannot say*, and use the rest as reference for evaluating NEM, which is described in Section 4.5. This leaves 949 events as our reference dataset, of which only 41% of the labels are *strange* or *highly unusual*. It has to be noted that even though our test set has equal size samples from both NWE and GWE, the true distribution is not uniform.

Language Model Separability Given the annotations, we test to see if the sentences corresponding to anomalous events can be separated from normal events by simpler features. We build a n-gram language model from the training data set used for argument composition and measure the perplexity of the sentences in the test set. Figure 4 shows a comparison of the perplexity scores for different labels. If the n-gram features are enough to separate different classes of sentences, one would expect the sentences corresponding to *strange* and *highly unusual* labels to have higher perplexity ranges than *normal* sentences, because the language model is built from a dataset that is expected to have a distribution of sentences where majority of them contain normal events. As it can be seen in Figure 4, except for a few outliers, most data points in all the categories are in similar perplexity ranges. Hence, sentences with different labels cannot be separated based on an n-gram language model features.

4.5 Evaluation

We evaluate the performance of event composition by comparing the predicted labels from the classifier against the ones given by MACE. We merge the two anomaly classes and calculate accuracy of the binary classifier, and the precision and recall of anomaly detection.

Baseline We compare the performance of our model against a baseline that is based on how well the semantic arguments in the event match the selectional preferences of the predicate. We measure selectional preference using Point-wise Mutual Information (PMI) (Church and Hanks, 1990) of the head words of each semantic argument with the predicate. The baseline model is built as follows. We perform

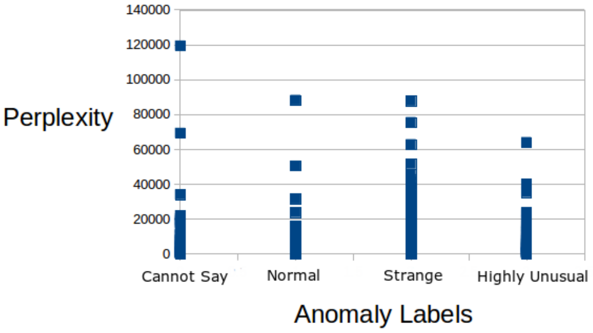

Figure 4: Comparison of perplexity scores for different labels

		NEM	Baseline
Accuracy		65.44%	45.22%
Anomalous	Precision	56.55%	36.30%
	Recall	48.22%	59.50%
Normal	Precision	64.62%	42.08%
	Recall	77.66%	33.60 %

Table 2: Classification Performance and Comparison with Baseline

dependency parsing using MaltParser (Nivre et al., 2007) on the sentences in the training data used in the first phase of training to obtain the head words of the semantic arguments. We then calculate the PMI values of all the pairs $< h_A, p >$ where h is the head word of argument A and p is the predicate of the event. For training our baseline classifier, we use the labeled training data from the event composition phase. The features to this classifier are the PMI measures of the $< h_A, p >$ pairs estimated from the larger dataset. The classifier thus trained to distinguish between anomalous and normal events is applied to the test set.

Table 2 shows the results and a comparison with the PMI based baseline. The accuracy of the baseline classifier is lower than 50%, which is the expected accuracy of a classifier that assigns labels randomly. The precision of that random classifier in predicting anomalous events is expected to be 41%, since that is the percentage of anomaly labels in our reference set as described in Section 4.4. The accuracy of NEM is higher than the baseline model. One possible reason for the PMI based baseline having higher recall in predicting anomaly and lower precision is that the statistics estimated from larger training data cannot be generalized to the test set due to sparsity issues. This indicates the advantage of using continuous representations at a higher level of abstraction as features for classification.

To further compare NEM with human annotators, we give to MACE, the binary labels produced by NEM along with the annotations and measure the competence. For the sake of comparison, we also give to MACE, a list of random binary labels as one of the annotations to measure the competence of a hypothetical worker that made random choices. These results are reported in Table 3. It can be seen that the performance of NEM is comparable at least to the least competent human.

5 Discussion and Future Work

The two evaluation experiments show that the neural network does learn to distinguish between normal and anomalous events. Future improvements to this model will include better event extraction techniques.

Since the current approach is supervised, the training data size for learning event composition is limited. We plan to develop unsupervised approaches that can learn good models of normal events, and detect anomalies based on how well new events fit in the model. One possible approach is to do learning

Human average	0.59
Human highest	0.70
Human lowest	0.26
Random	0.02
NEM	0.26

Table 3: Anomaly Detection Competence

based on contrastive estimation in the second phase as well. The assumption behind taking this approach for learning is that a randomly generated data point is likely to be a negative example, which is not necessarily true for learning event composition. Generating malformed events that are syntactically valid but anomalous without much human effort can greatly help in developing such an unsupervised algorithm.

One important aspect of anomaly that is currently not handled by NEM is the level of generality of the concepts the events contain. Usually more general concepts cause events to be more normal since they convey lesser information. For example, an American soldier shooting another American soldier may be considered unusual, while a soldier shooting another soldier may not be as unusual, and at the highest level of generalization, a person shooting another person is normal. This information of generality has to be incorporated into the event model. This can be achieved by integrating real world knowledge from knowledge bases like Wordnet (Miller, 1995) or from corpus statistics like the work by Lin (1998) into the event model. Bordes et al. (2011) learn continuous representations of entities and relations in knowledge bases. More recently, an alternative approach for doing the same was proposed by Chen et al. (2013). These representations can greatly help modeling events.

Finally, the idea of modeling event composition can help processing event data in general and can be applied to other tasks like finding co-referent events.

6 Conclusion

We introduced the problem of anomalous newswire event detection and illustrated its difficulty. Our approach is similar to the ones successfully used for modeling semantic composition. We showed that while our event composition model does learn to distinguish between normal and anomalous events, there is scope for improved models that can effectively incorporate real world information and can be trained in an unsupervised fashion. We note that in general event composition is more difficult than traditional semantic composition since the former also deals with pragmatics. Consequently the set of nonsensical events is different from the set of anomalous sentences, and while meaningless events and well composed normal events are two ends of the semantic spectrum, semantically valid anomalous events lie somewhere between them.

Acknowledgements

This research was supported in part by DARPA grant FA8750-12-2-0342 funded under the DEFT program.

References

Marco Baroni and Alessandro Lenci. 2010. Distributional memory: A general framework for corpus-based semantics. *Computational Linguistics*, 36(4):673–721.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. 2010. Theano: a cpu and gpu math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, volume 4.

Antoine Bordes, Jason Weston, Ronan Collobert, Yoshua Bengio, et al. 2011. Learning structured embeddings of knowledge bases. In *AAAI*.

Danqi Chen, Richard Socher, Christopher D Manning, and Andrew Y Ng. 2013. Learning new facts from knowledge bases with neural tensor networks and semantic word vectors. *arXiv preprint arXiv:1301.3618*.

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*, 16(1):22–29.

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

Jeffrey L Elman. 2009. On the meaning of words and dinosaur bones: Lexical knowledge without a lexicon. *Cognitive science*, 33(4):547–582.

Katrin Erk, Sebastian Padó, and Ulrike Padó. 2010. A flexible, corpus-driven model of regular and inverse selectional preferences. *Computational Linguistics*, 36(4):723–763.

Christoph Goller and Andreas Kuchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *Neural Networks, 1996., IEEE International Conference on*, volume 1, pages 347–352. IEEE.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *Proceedings of NAACL-HLT*, pages 1120–1130.

Klaus Krippendorff. 1980. *Content analysis: An introduction to its methodology*. Sage Publications (Beverly Hills).

Alessandro Lenci. 2011. Composing and updating verb argument expectations: A distributional semantic model. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 58–66. Association for Computational Linguistics.

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

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. *CoNLL-2013*, 104.

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

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 354–362. Association for Computational Linguistics.

Richard Socher, Christopher D Manning, and Andrew Y Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9.

Richard Socher, John Bauer, Christopher D Manning, and Andrew Y Ng. 2013a. Parsing with compositional vector grammars. In *In Proceedings of the ACL conference*. Citeseer.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1631–1642.

Yorick Wilks. 1973. Preference semantics. Technical report, DTIC Document.