mrs2vec: Word Embedding with Semantic Contexts

Rik Koncel-Kedziorski
University of Washington, USA
kedzior@uw.edu

Abstract

Mapping words to real-valued vectors for use in NLU applications has attracted renewed interest from researchers in the past few years due to advances in neural techniques. Previous work has incorporated forms of syntactic and semantic knowledge into the word-embedding process with positive results. None, however, has leveraged a syntactic resource as precise as the English Resource Grammar (ERG). Moreover, the ERG parse of a sentence contains a semantic graph derived from the syntax which expresses compositional semantic relations between the entities and events denoted. This paper outlines *mrs2vec*, a method for training word embeddings from semantic dependencies as given by the ERG. I show how these embeddings compare to state-of-the-art embeddings which incorporate syntactic and non-compositional lexical semantic knowledge.

1 Introduction

Many NLU applications could benefit from an accurate understanding of word meaning, but providing such an understanding has proved challenging. Modern research into this provisioning primarily takes two forms. One set of approaches focus on recording human knowledge of word-meaning in resources which computers can easily utilize (e.g. WordNet (Miller, 1995), Framenet (Baker et al., 1998), Freebase (Bollacker et al., 2008), etc). Another set of approaches make use of the distributional hypothesis that words in similar contexts share similar meanings (Lin, 1998; Mikolov et al., 2013a). These methods observe words in contexts throughout a large corpus and utilize expensive algorithms to map them into a high-dimensional space.

Continuous space word vectors produced by neural networks have proved to be a powerful statistical method for providing information about a word's meaning. These methods make use of the "unreasonable effectiveness" of neural networks to map words to points in a relatively low dimensional vector space (Karpathy, 2015). These mappings have been shown to translate intuitive relationships between words into spatial relationships between points in the vector space (Mikolov et al., 2013a). Recently, Levy and Goldberg (2014) showed that using syntactic dependency contexts (rather than the traditional adjacency constraints) in the training of embeddings improved performance on a word similarity judgment task. Additionally, incorporating non-compositional semantic knowledge from lexical knowledge bases into word embeddings has also been shown to improve embeddings (Faruqui et al., 2015).

This paper details a method called *mrs2vec* which incorporates semantic knowledge and syntactic structure to produce neural word embeddings. A corpus of semantic graphs of sentences from English Wikipedia are used as training data for the skip-gram word embedding algorithm (Mikolov et al., 2013a). These graphs are produced by the English Resource Grammar (Flickinger, 2000; Flickinger, 2011). The embeddings produced by the *mrs2vec* method are compared with embeddings trained from syntactic dependency graphs. Additionally, I report the results of augmenting both dependency-based and *mrs2vec* embeddings with information from lexical semantic knowledge bases.

My results show that *mrs2vec* embeddings perform comparably on standard word similarity tasks. However, standard techniques for augmenting word embeddings with lexical seman-

tic knowledge do not to have the same consistent positive impact on the *mrs2vec* embeddings as they do on dependency-based embeddings. I also outline several other advantages of these embeddings. Notably, *mrs2vec* embeddings provide separate vectors for differing parts of speech of orthographically identical words. In limited cases, separate vectors are also provided for different word senses. More importantly, *mrs2vec* embeddings can be integrated into other systems which rely on the ERG for syntactic and semantic parsing.

2 Background

The previous work from which this one draws inspiration is divided into three subsections below, covering first lexical knowledge bases, ERG-specific background knowledge, and methods for word vectorization.

2.1 Knowledge Bases

Knowledge bases (KB) abound in NLU applications. Projects like Freebase (Bollacker et al., 2008) attempt to encode human knowledge about a diverse set of concepts in a semi-structured graph database. Objects (real world entities, topics, or concepts) are related to each other through a diverse collection of edge labels. The information from Freebase has been used to improve systems for tasks such as joint coreference resolution and named entity linking (Hajishirzi et al., 2013) and distantly supervised relation extraction (Mintz et al., 2009). However, the knowledge in these resources mostly concerns specific important people, places, or events (for example, celebrities). Lexical knowledge about the meanings of common words is often excluded, minimizing their utility for general purpose NLU.

However, there are KBs which provide such lexical information. Chief among these is WordNet (Miller, 1995), developed at Princeton University and introduced in 1995. Prior to WordNet, machine readable dictionaries were the primary way to incorporate some knowledge about words and their meaning into an algorithm. WordNet is organized according to psycholinguistic principles of word similarity (Miller et al., 1990). Each surface form is associated with one or more senses. Senses are organized into sets of synonyms, and synonym sets are then related to each other by hyper/hyponym relations where these relations obtain. WordNet has been used in numerous NLU tasks and applications, but its being used to augment word embeddings through a process called *retrofitting* described by Faruqui et al. (2015) (reviewed below) is especially relevant to the current work. I compare the effect of retrofitting both *mrs2vec* word embeddings and dependency based embeddings with relations from WordNet.

The success and shortcomings of WordNet prompted other similar projects. WordNet's synonym sets and other relations do not capture necessary facts about verbs such as the predicate argument structures associated with specific verb senses or the syntactic configurations into which a verb may enter (for example, if and how its valence can be changed). VerbNet (Schuler, 2005), a collaboration directed by University of Colorado's Martha Palmer, is intended as a broad-coverage, comprehensive collection of verbs and their argument structures that fills in some of the gaps of WordNet. The purpose of VerbNet is to capture the more complicated surface realizations of verb senses. In its current form, VerbNet organizes a lexicon of 3769 verb lemmas covering 5257 senses into 247 first-level classes. These classes are an extension and refinement of those outlined by Beth Levin's investigation of English verb classes (Levin, 1993). Syntactic descriptions of possible surface realizations of verbal argument structure, as well as semantic constraints on argument types, are also detailed.

FrameNet (Baker et al., 1998) is based on linguist Charles Fillmore's theory of meaning called "Frame Semantics" (Fillmore, 1982). A *frame* is an event, relation, or entity and its participants (called *frame elements*). For example, if a sentence describes someone cooking food, this instantiates an *Apply_heat* frame and the frame elements *Cook, Food, Heating_instrument,* and *Container*. The FrameNet corpus consists of over 170,000 sentences manually annotated from a stock of over 1000 frames. Automatic tools such as Frame-Semantic parsers have been developed based on this data (Das et al., 2014), but their coverage is limited. In this work, I also compare the effect of retrofitting both *mrs2vec* embeddings and dependency-based embeddings with FrameNet knowledge.

Another semantic relation that could be leveraged for lexical knowledge is the paraphrase relation. The Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) includes a collection of over 220

million English paraphrase pairs, including over 8 million lexical pairs. These paraphrases were extracted automatically from parallel texts. In a large multi-lingual corpus, if two English words map to the same non-English word, the English words are considered paraphrases of each other. The PPDB also contains phrasal paraphrases obtained by the same method, but they are not used in this work. Here, I retrofit *mrs2vec* and dependency embeddings with PPDB knowledge for comparison.

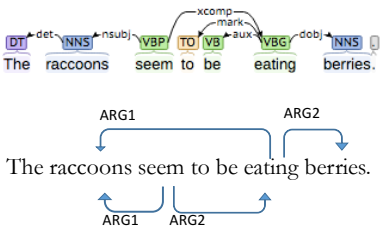

Figure 1: Comparison of Stanford syntactic (above) and ERG semantic (below) dependencies

2.2 English Resource Grammar

This work utilizes the English Resource Grammar (Flickinger, 2000; Flickinger, 2011) to provide syntactic and semantic information for improving word embeddings. The ERG is a hand-built grammar for the English language that covers a variety of genres and domains. It has been in continual development by specialists for over 20 years and at present (in its version 1212 release) it contains 222 phrase structure, 84 lexical rules and roughly 38,000 lexical entries of 1000 lexical types (Flickinger et al., 2014). Due to its complexity and active, attentive development, the ERG is able to produce precise parses which cover a large percentage of formal and informal English phenomena.

The syntactic formalism of the ERG is Head-driven Phrase Structure Grammar (HPSG) (Pollard and Sag, 1994), a constraint-based, lexical generative grammatical theory which allows for both parsing and generation. In an HPSG grammar specification, lexical entries are represented as *feature structures* (rather than the POS node labels commonly used in other grammatical formalisms). This increased expressiveness allows for more accurately capturing the valid syntactic configurations of a language. Parsing proceeds by unification of feature structures according to the grammar rules until a feature structure representing the sentence is (or is not) produced.

HPSG rules and lexical entries include a semantic component, thereby allowing for the composition of a semantic graph of a parsed sentence from its syntactic derivation. The ERG makes use of Minimal Recursion Semantics (MRS), a compositional semantic formalism (Copestake et al., 2005). The basic unit of MRS is the Elementary Predication (EP), defined as a single relation and its associated arguments. For example, the EP corresponding to the lexeme *eat* in Figure 1 can be represented as a two-place relation between the raccoons (or technically, a semantic variable representing the entity denoted by *the raccoons*) and the berries they eat.

MRS is an attempt to be both linguistically accurate and computationally tractable. One way it accomplishes this is by allowing for underspecification of quantifier scope, which has traditionally prohibited computational tractability. Correct quantifier scoping can require extra-sentential context, but ignoring quantifier scoping altogether would introduce unacceptable ambiguity. MRS resolves this by defining scope of EPs only as exactly as the sentence does, i.e. allowing for ambiguity only where it exists. Although in this work we ignore scopal information, future work or downstream tasks could benefit from taking this important feature of MRS into account.

2.3 Neural Word Embeddings

Neural word embeddings are an immediate byproduct of neural language models. A neural language model is a language model produced using a neural network to model the complex, long-distance dependencies of natural language text. Bengio et al. (2003), in their foundational work on the subject, describe the procedure for building a neural language model. Each word from the target vocabulary is associated with a random real-valued vector. The probability function of word sequences is expressed as a function of the vectors associated with words in the sequence. The network model then learns both good word vectors (embeddings) and parameters of the probability function by observing large tracts of running text.

Often in neural language modeling, deep network structure is employed. Deep networks are those with several intermediate or "hidden" layers between the input and output. Deep networks

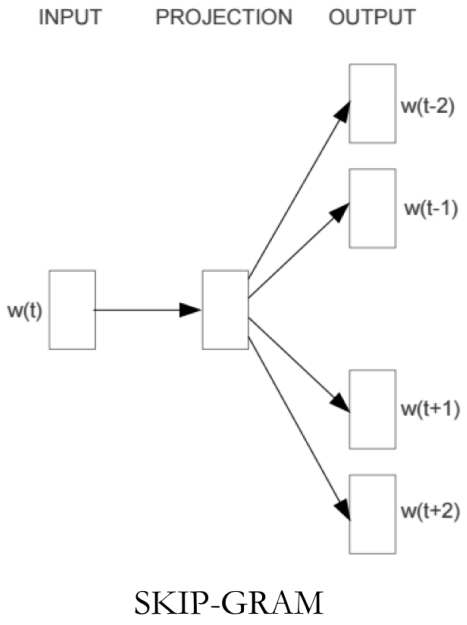

Figure 2: The Skip-gram Model architecture (Mikolov et al., 2013b). Words are represented as vectors (boxes) which interact with the network.

are trained by backpropagating errors to maximize the log-likelihood of their training data. Due to the complexity of backpropagating errors through a deep network structure, training a deep neural network language model is significantly computationally expensive. However, if the goal is only to produce high quality embeddings (rather than an entire neural language model), then the process can be made significantly faster by simplifying the model in the manner discussed below.

2.3.1 Skip-Gram Model

Recently, Mikolov et al. (2013a) showed that high-quality word embeddings can be produced quickly by reducing the complexity of the neural network. To do so, they remove the non-linear hidden layers of the network and share the projection layer between all words in the vocabulary.

They discuss two methods for training such embeddings. In the Continuous Bag-of-Words setup, the training goal is to predict the middle word of a $2n + 1$-long span of words based on the others. They also define the Skip-gram model, where the training goal is to predict, from the middle word, the surrounding words in the window. An overview of the Skip-gram model can be seen in Figure 2.

Typically, Skip-gram models are trained on large corpus of text, which we can consider as a sequence of words $(w_1, w_2, ..., w_T)$.[1] A window size c is selected, and the training objective is to then maximize the average log probability of the data. This objective is shown in Equation 1.

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c<j<c, j\neq 0}\log p(w_{t+j}|w_t) \qquad (1)$$

$p(w_{t+j}|w_t)$ is typically defined using the following formulation of the softmax function:

$$p(w_O|w_I) = \frac{\exp({v'_{w_O}}^{\top} v'_{w_I})}{\sum_{k=1}^{K}\exp({v'_{w_k}}^{\top} v_{w_I})} \qquad (2)$$

Here, v_w and v'_w are the input and output vector representations of w, and K is the cardinality of the vocabulary. $p(w_{t+j}|w_t)$ can be approximated by hierarchical softmax when the computational cost of full softmax is too high (such as over large vocabularies).

It is also possible to approximate the softmax function with a technique called Noise Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2012). Here, "noise" examples are added to the training data and the training objective becomes distinguishing the "true" data-points from the noise. Because simplifying assumptions can be made with this new objective, and through the use of Monte Carlo approximation, NCE can be less expensive than learning probabilities using a softmax function. Because the gradient of the NCE training objective is 0 when the probability distribution it assigns to words and their contexts is the same as the empirical distribution, NCE is an effective approximation of the full softmax (Dyer, 2014).

Mikolov et al. (2013b) outline a simplification of NCE called Negative Sampling that preserves the quality of the learned vector representation of words at the expense of accurately approximately modeling the log probability of the softmax. They replace the $\log p(w_{t+j}|w_t)$ term in the Skip-gram objective with the Negative Sampling objective:

$$\log\sigma({v'_{w_O}}^{\top} v'_{w_I}) + \sum_{i=1}^{k}\mathrm{E}_{w_i\sim P_n(w)}[\log\sigma(-{v'_{w_i}}^{\top} v_{w_I})] \qquad (3)$$

[1] It is traditional to use w_t to indicate the target word at time t, as if words are being processed in a time sequence. In this notation, T is used to indicate the cardinality of the corpus.

Here, σ is a sigmoid function, and P_n is a "noise" distribution of randomly assigned word/context pairs. Ideally, these pairs are unattested in the training corpus. For each attested example, k "noise" pairs are selected. Negative Sampling in this way is faster than NCE for the purposes of training word vector representations, but cannot be applied to other tasks where a full language model is required.

2.3.2 Extensions of the Skip-Gram model

Several researchers have tried to improve neural word embeddings by incorporating various forms of knowledge. Levy and Goldberg (2014) use dependency structure to train skip-gram embeddings. An outline of their training model is shown in Figure 3.

Using the fast arc-eager parser described in Goldberg and Nivre (2012), the authors construct training dependency triples from the text of English Wikipedia. Each triple (g, r, d) connects a governing word g to a dependent d with an arc labeled r. Noting that there is no prohibition in the skip-gram specification on words and contexts being drawn from different vocabularies, they define separate target and context vocabularies to allow for dependency labels to inform the embedding process. The context vocabulary consists of dependents concatenated with an arc label and governors concatenated with an arc label and a special inverse symbol to indicate that the governor-dependent relationship is inverted in this example. Thus, each (g, r, d) triple is converted into the training pairs (d, gr) and (g, dz^{-1}).

The modified objective, accounting for the differing target and contexts vocabularies, is as follows:

$$\frac{1}{2|\mathrm{C}|}\Big(\sum_{g\in V}\sum_{(g,r,d)\in \mathrm{C}} \log p(g|dr) + \sum_{d\in V}\sum_{(g,r,d)\in \mathrm{C}} \log p(d|gr^{-1})\Big) \quad (4)$$

Here we let C represent the set of training triples, and V the vocabulary we want to embed in a vector space. Training proceeds as outlined above, using negative sampling to facilitate fast parameter estimation.

Levy and Goldberg (2014) claim that their vectors display "functional" similarity instead of the "broad topical" similarity captured by linear context models. They argue that words which appear

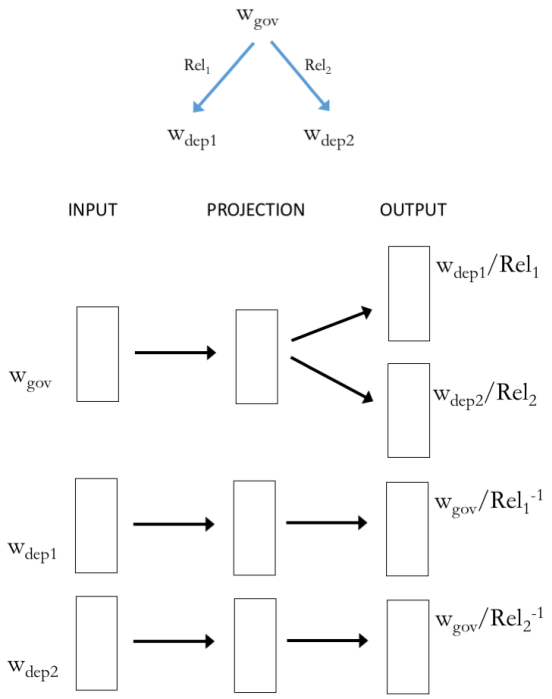

Figure 3: The Dependency-based Skip-gram model architecture. The dependency structure shown at the top in blue results in the training processes shown beneath.

in a similar set of dependency relations to context words have similar function in their sentences, for example as subjects or as objects of prepositions. Words in similar windows have no such guarantee. For instance, both adverbs and objects can immediately follow many verbs, and often adverbs that follow verbs will be topically related to those verb's objects. But adverbs and objects do not have the same function in a sentence, and so Levy and Goldberg (2014) argue that the similarity between topically-related adverbs and objects is less significant than the similarity shared by objects that can be selected by the same verb. Their improved results on a word similarity judgment task bolster their argument.

Qualitative evidence for the value of functional similarity is offered through model introspection. A skip-gram model with a 5-word linear context will select *gainsville, fla* and *jacksonville* as the nearest neighbors to *florida*. While these words (and abbreviations) are related, they are not of the same semantic type (namely, states) as the target word. The dependency-based context model instead selects *texas, louisana,* and *georgia*. The

current work builds on the intuition that relational structure between words can be leveraged, but makes use of a semantic structure rather than a syntactic one.

It's worth noting that the use of dependency information in distributional semantic models has a long history, dating back more than 20 years. Lin (1998) uses dependency triples to improve an information-theoretic similarity metric. Here, counts of *governor, relation, dependent* triples are extracted from a large corpus. The information content of a triple and its count (written as $I(g, r, d)$ for governor g, relation r, and dependent d) is defined as the mutual information between the governor and dependent, or

$$I(g, r, d) = \log \frac{||(g, r, d)|| \times ||(*, r, *)||}{||(g, r, *)|| \times ||(*, r, d)||} \quad (5)$$

where $||(g, r, d)||$ is the count of the triple (g, r, d) in the corpus, and a wild-card $*$ indicated aggregation of all triples matching the remaining specification (so that, e.g. $||(*,$*nsubj*,*dog*$)||$ is the count of all occurrences in the corpus of *dog* related to anything by the *nsubj* relation).

In addition to dependency information, other forms of linguistic knowledge have been incorporated into the word embedding process. Yu and Dredze (2014) seeks to incorporate semantic knowledge into word vectors for improved performance. They define an objective function which maximizes the likelihood of a word given the words to which it is related in a knowledge base. They combine this objective with the Continuous Bag-Of-Words neural embedding objective to arrive at a joint context/semantic embedding model. They evaluate their embeddings using the PPDB and WordNet as lexical semantic resources and show that their results improve over baselines on several tasks.

A further effort to incorporate semantic knowledge into word embeddings was outlined in Faruqui et al. (2015). In that project, word embeddings are refined using a technique called retrofitting. Retrofitting is a method for moving the vectors for words that are associated in a knowledge base closer together. Retrofitting is more flexible than the method introduced in Yu and Dredze (2014) because it is quicker and can be done as a post-processing step. Any neural embeddings, produced from any training data by any method, can be retrofit with a semantic lexicon. Faruqui et al. (2015) show how incorporating WordNet, Framenet, and the PPDB into a variety of vectors affects (and often improves) results on various syntactic and semantic similarity tasks. I explore the possibility of retrofitting semantic-based embeddings to the WordNet and the PPDB and compare against dependency-based embeddings retrofit with these knowledge-bases.

3 Motivation

Previous work has endeavored to augment the distributional word knowledge captured by the skip gram word embedding strategy with various forms of human knowledge about language. In the case of the dependency-based embeddings of Levy and Goldberg (2014), syntactic knowledge is added via the dependency parser (trained on the arc labeling decisions made by human annotators.) In the case of the retrofitting work of Faruqui et al. (2015), manually constructed lexical knowledge bases like WordNet and Framenet are employed. With regard to the PPDB, it is important to note that this resource leverages parallel bilingual texts. The ability to translate between languages employs conscious, expert knowledge about the structure and possibilities of both languages.

The current work uses the ERG as a source of expert human knowledge about language to improve word vectors. The ERG has several advantages over the sources previously used. The dependency parser whose output Levy and Goldberg (2014) incorporate into their word vectors was introduced in Goldberg and Nivre (2012). This parser is fast while maintaining high accuracy, but it, like most other off-the-shelf dependency parsers available, is trained on sections 2-22 of the Penn-WSJ Treebank (Marcus et al., 1993), which is converted to dependencies by the deterministic rules outlined in De Marneffe et al. (2006). The variety of language of this training corpus is quite limited; for example, it is known to contain very few questions. Dependency parses of questions (and presumably those of many other constructions which do not obtain in newswire text) from parsers trained on this data are notorious for their poor quality.

The ERG, on the other hand, is constantly being developed to account for ever more grammatical phenomena in a principled and justifiable way by a number of researchers. This effort is facilitated by the Redwoods treebanking project, which

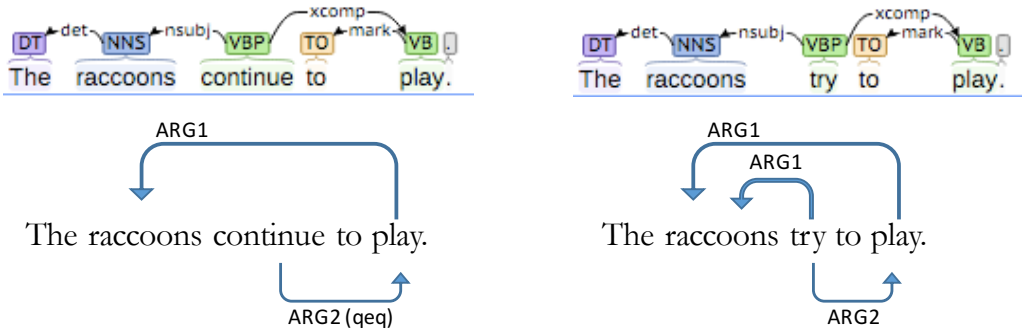

Figure 4: Comparison of analyses for raising (left) and control (right) verbs. Note the syntactic symmetry in the dependency structure above. The semantic structures assigned to each sentence, however, differ due to the nature of the matrix verbs.

emphasizes incremental improvement and reuse of existing annotation decisions (Flickinger et al., In press). A key component of this annotation process is presenting annotators with a collection of machine produced parses for disambiguation. Thus, I expect that the syntactic knowledge encoded in the ERG is better than that provided by dependency parsers.

A goal of Levy and Goldberg (2014)'s work was to capture functional word similarity rather than topical similarity. Dependency structure was shown to aid in this goal. I hypothesize that co-occurrence in semantic structures can better capture functional similarity. The ERG is used to produce a semantic graph of each of a collection of sentences.

This semantic graph is composed directly from the syntactic analysis of the sentence, through the application of rules of semantic composition.

Figure 1 shows a comparison of the (Stanford) dependency graph of a sentence and an ERG semantic graph of the same sentence. Note that the syntactic structure for this sentence does not directly link *raccoons* to the embedded verb *eat*. As such, Levy and Goldberg (2014) will not consider this verb as a context for *raccoons*. The graph produced by the ERG, however, does make this important connections. I expect that embeddings which account for such information will facilitate accuracy in downstream semantic tasks.

Consider also the raising and control structures shown in Figure 4. The Stanford dependency structure for these two sentences are identical. Yet on the left, the raising verb *continue* expresses a property of the event described in its complement. In this case, the raccoons, who were playing, continue to do so. Continuation is not a property of the raccoons, but rather the playing event as a whole. The ERG links the EP for *continue* with that for *play* via the argument structure of *continue* (modulo quantifiers, as there are multiple possible scopings and so a need for underspecification). It is not linked to the EP for *raccoons* because this sentence is not about the continuation of the raccoons (i.e. their continued existence).

In the right hand example, the control verb *try* is linked to both the EP for *raccoons* and *play* because it describes a relation between the raccoons and playing. The sentence describes a "trying" event, and it is the raccoons who try, and so it is their EP that fills the ARG1 role of the *try* predicate. Similarly, the "trying" event has as its goal "playing", and so the EP for *play* is linked as the ARG2 of *try*.

The ERG encodes this difference in the semantics produced for raising and control sentences. This is a more accurate reflection of the semantic nature of these verbs than can be obtained by looking purely at the parse structure. It is hoped that this semantic accuracy will translate into better word vectors for downstream tasks. For instance, if we wanted to find a word similar to *raccoons*, the raising dependency structure might guide us toward things that also serve as the subject of *continue*, such as *rainstorms* or *trials*. These are less desirable than words which can serve as the ARG1 of *play*, such as *goats* or *children*.

Now suppose we are automatically generating a story about goats, and, noting their similarity to raccoons, we are basing our plot events on verbs similar to those connected with *raccoons* in our training data. From the dependency structures of

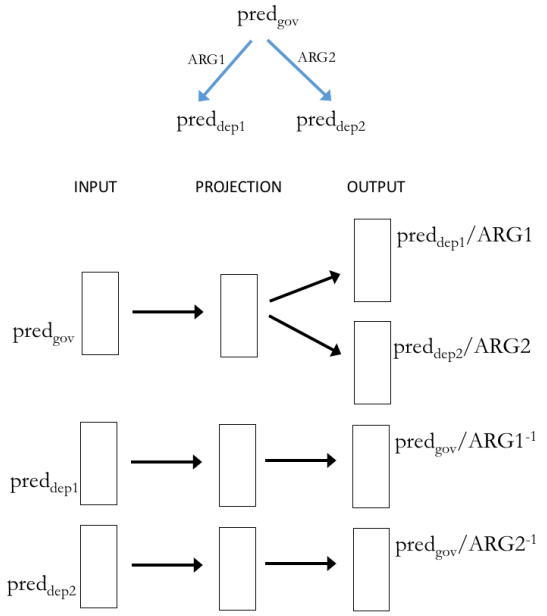

Figure 5: The *direct* model of *mrs2vec*.

these examples, we would be limited to events similar to "continuing" and "trying", which include boring activities like contriving and recommencing. However, the semantic graphs connect *raccoons* to *play*, leading us to the more interesting similar events of "singing" and "umpiring". Using these more interesting events, we can generate a better story for, for example, a dynamic video game plot or a grade-school child's customized educational materials. These are just a few examples of the value of semantically accurate embeddings on downstream tasks, but others can be imagined.

4 Methodology

In this work, I make use of the WikiWoods corpus (version 1212) (Flickinger et al., 2010) as training data for the word embeddings. WikiWoods contains ERG syntactic parses and semantic graphs of more than 50 million sentences taken from English Wikipedia. I use the Elementary Dependency Structure (EDS) format of the semantic graph, a variable-free semantic dependency graph (Oepen et al., 2004).

I make use of the generalized skip-gram model presented in Levy and Goldberg (2014) with slight modifications to accommodate the information from the MRS. The *mrs2vec* models are constructed by optimizing the same objective function as the dependency-based models, listed in Equation 4.

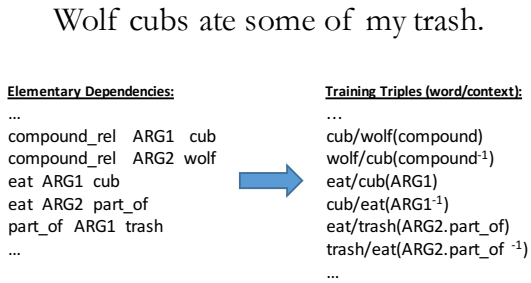

Figure 6: Example of collapsing abstract predicates. Here, the *compound_rel* and *part_of* predicates are collapsed into the context.

Two methods of transforming EDS into training data are used. By the *direct* method, each EDS triple is converted directly into the training pairs. An EDS describing an EP g whose rth argument is the EP d is transformed into training pairs (g, dr) and (d, gr^{-1}). Here, items in the context vocabulary are concatenations of an EP with an argument index. A graphical representation of this method is shown in Figure 5.

By the *collapsed* method, abstract predicates, corresponding to non-lexical semantic qualities such as passivization or implicit quantifiers, are "collapsed" into the argument label.[2] Figure 7 shows the architecture of the collapsed model. Similar to collapsed dependencies, collapsing abstract predicates allows for closer association of contentful lexemes. For instance, in a noun-noun compound, the EPs corresponding to each noun are related only though the argument structure of a special compound predicate. Using the *direct* method, no training triple would relate the nouns to each other. Instead, I collapse the compound predicate into the argument label. I treat the noun EP that is the ARG1 of the compound as the governor and the ARG2 as the dependent (see Figure 6). Another example is the *part-of* relationship, which serves often as an argument of a verb and takes the head of a noun phrase as its argument. I directly connect the verb and noun with a context that collapses the *part-of* relationship with whatever argument it is of the verb.

[2] Abstract predicates are contrasted with surface predicates representing overt words in the sentence.

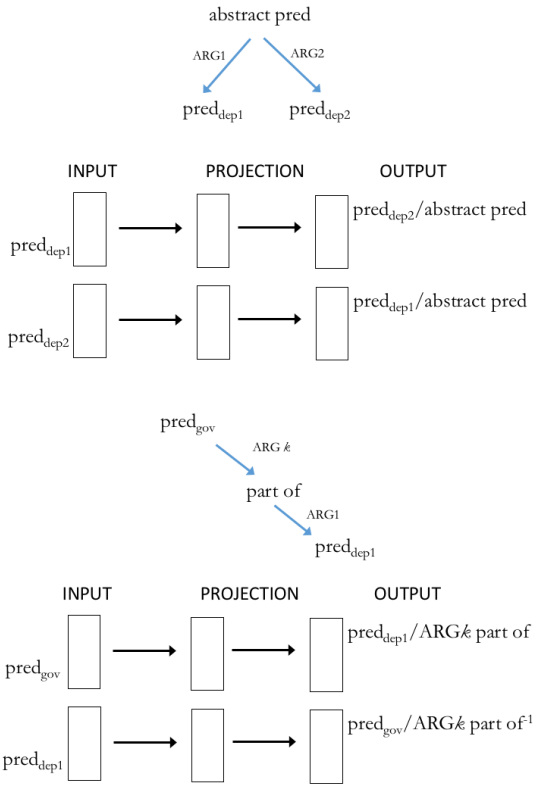

Figure 7: The *collapsed* model of *mrs2vec*. Dashed lines represent information in the MRS graph which has been "collapsed" into new edges.

5 Experiments

For quantitative evaluation, I use two common word similarity datasets: WS 353, containing 353 word pairs with human relatedness judgments; and SimLex999, designed with a stricter notion of similarity than WS 353. The models are asked to provide a similarity score (cosine distance) for each word pair.

The datasets used here are typical in word embedding literature. However, they do not provide any context for a word. Rather, two words are listed and scored in isolation. Due to lexical ambiguity, it is sometimes the case that a word will map to several EPs in the ERG lexicon. When this occurs, I evaluate similarity for all possibilities and choose the highest score.

I compare against embeddings trained using syntactic information from dependency parses. I use the pre-trained 300-dimensional embeddings made available by Levy and Goldberg (2014).[3] These embeddings are trained using English Wikipedia, similarly to *mrs2vec*, but using syntactic dependency parses of the sentences rather than the ERG semantic graphs. Both the syntactic dependency based embeddings and the *mrs2vec* embeddings use mid-2013 snapshots of English Wikipedia as their underlying corpus. As such, they should be roughly comparable in terms of vocabulary and contexts. However, while the WikiWoods corpus only contains sentences for which an ERG parse is available (about 85% of the corpus), the dependency-parsed Wikipedia will have some analysis for all sentences.

The syntactic dependency based embeddings are compared with three settings for transforming EDS to training data described above. In Table 1, the direct transformation is reported as *direct* and the collapsing of abstract predicates as *collapsed*. The *svo* setting is a variant of the *direct* transformation where only EDS triples containing a verbal predication are retained for training.[4]

To evaluate the potential for subsequently retrofitting *mrs2vec* embeddings, I compare the *direct* embeddings retrofit with WordNet and PPDB data to dependency based embeddings retrofit with WordNet and PPDB. This is accomplished using the code made available by Faruqui et al. (2015) with slight modifications to enable the retrofitting of predicate embeddings where needed.

I report two metrics in Table 1. First, in keeping with the tradition of work on word embeddings, I report Spearman's coefficient, a measure of the correlation between model and human judgments. I also report the accuracy on a pairwise ranking task (PWR) which I believe is more in line with current uses of word embeddings. For target word *T* and options *A* and *B* where the human-reported similarity of *T* and *A* is greater than that of *T* and *B*, the model is correct if it ranks *A* as more similar than *B* and wrong otherwise. In this task, no *A* and *B* are used if they both are equally similar to *T* by the human standard.

While the Spearman coefficient is a standard evaluation for word embeddings, it has the disadvantage of capturing unintended relationships in the data. For example, *attend* and *arrive*, verbs, are rated roughly twice as similar as *groom* and *bride* or *bride* and *princess* in the SimLex dataset.

[3] These embeddings can be found at `https://levyomer.wordpress.com/2014/04/25/dependency-based-word-embeddings/`

[4] This setting was suggested by Guy Emerson (personal communication) and is the subject of a forthcoming work on using MRS for word embeddings.

	WS 353		SimLex	
	ρ	PWR	ρ	PWR
Dep	**0.622**	0.727	**0.447**	**0.638**
svo	0.41	0.68	0.279	0.61
direct	0.516	**0.744**	0.313	0.614
collapsed	0.508	0.729	0.299	0.622

Table 1: Comparison of syntactic dependency (Dep) and several *mrs2vec* varieties on the WS 353 and SimLex-99 Datasets. ρ, is the Spearman coefficient, and PWR is the pairwise ranking task accuracy.

However, the human evaluators were never asked to rank the similarity of *attend* and *arrive* relative to the similarity of *groom* and *bride*. It's hard to imagine making so complex a judgment as "how much more similar are a pair of verbs than a pair of nouns". Yet the Spearman coefficient will punish a model that fails to make this judgment accurately.

Moreover, I cannot think of a task which requires such a complex similarity judgment. Rather, the use of word embeddings in tasks such as unsupervised semantic role labeling as reported in Luan et al. (2016) involves considering the similarity of a target word and a set of candidates, not relative similarities between pairs of words of different parts of speech. Thus, I introduce the PWR evaluation as a proxy for candidate selection use cases of word embeddings. The PWR captures the value of a word embedding method for a variety of tasks where direct comparison of candidates to a single target is required.

As is evidenced in Table 1, the *mrs2vec* embeddings perform reasonably well at this task. The *direct* training method outperforms all models in the pairwise ranking task over the WS 353 data. Syntactic dependency based vectors retrofit with the PPDB outscore all other models elsewhere.

In order to determine if the retrofitting method can be used to improve *mrs2vec* embeddings, I retrofit the *direct* embeddings with each of the PPDB and WordNet. A comparison of the results is given in Table 3. Moderate gains can be seen on the SimLex dataset, but no consistent pattern of improvement emerges. The dependency based embeddings, however, show consistent improvement when retrofit with the PPDB. I believe the reason *mrs2vec* embeddings don't show such an improvement is due to the fact that the vectors correlate to predicates rather than surface forms.

	WS 353		SimLex	
	ρ	PWR	ρ	PWR
Dep + WN	0.542	0.71	0.497	0.666
Dep + FN	0.458	0.711	0.359	0.632
Dep + PPDB	**0.631**	**0.741**	**0.506**	**0.672**
direct + WN	0.39	0.702	0.366	0.625
direct + FN	0.40	0.724	0.254	0.616
direct + PPDB	0.445	0.702	0.357	0.629

Table 2: Results of retrofitting dependency based and *mrs2vec* embeddings with WordNet (WN), FrameNet (FN), and the Paraphrase Database (PPDB).

Target	Dep	**Target**	*direct*
florida	texas louisiana georgia california	Florida	California Texas Minnesota Nevada
water	seawater groundwater rainwater floodwater feedwater brine wastewater sewage	water (noun)	seawater groundwater rainwater brine (noun)
		water (verb)	oil (verb) grass (verb) irrigate sugar (verb)
fish	fishes shrimp shellfish jellyfish salmon finfish lobster halibut	fish (noun)	arthropod seabird shrimp (noun) lobster
		fish (verb)	farm (verb) whale (verb) fish for hunt (verb)
try	tries tried trys trying	try (verb)	attempt (verb) endeavor (verb) set out loath (adj)

Table 3: Target words and the most similar words given by different embeddings.

The retrofitting process involves surface form relations, often given without enough context to produce an accurate parse. Absent these parses, the predicates that can be matched to PPDB entries are often incorrect, resulting in diminished performance. A good direction for future work would be to investigate better ways for leveraging surface form word knowledge to improve predicate embeddings.

6 Qualitative Evaluation

Introspection into the nature of the embeddings reveals the power of the semantic method used in *mrs2vec*. Looking at Table 3, both the dependency-based embeddings and *mrs2vec* em-

beddings return state names as the most similar words to *Florida*. However, only *mrs2vec* contains separate vectors for orthographically identical words with different parts of speech, as each form of these words will instantiate a different EP in the semantic graph. For example, words like *water* and *fish* can be used as both nouns and verbs. *mrs2vec* embeddings return separate points in the vector space for each use, and so we can query the model for words most similar to the verb or noun form independently. Dependency-based vectors are unable to do this, resulting in one sense of the word being under-represented in the similarity list. Additionally, information from multiple inflections of a word are collapsed into a single *mrs2vec* vector. For example, *try*, *tries*, and *trying* all map to the same vector. By contrast, inflected forms of *try* are the 4 nearest neighbors given by dependency-based vectors. Improvements to dependency based vectors such as lemmatization and part of speech tagging could yield similar results to *mrs2vec* embeddings.

7 Conclusion

This paper has described *mrs2vec*, a method for producing word embeddings from automatically derived semantic graphs of sentences. I have demonstrated that these embeddings perform comparably to other state-of-the-art embeddings on out-of-context word similarity tasks, and I have provided additional insight into the value of these embeddings through model introspection.

Extensions of this method include on training a vocabulary of semantic subgraphs rather than single predicates for calculating phrasal vectors, or composing these vectors for tasks like language modeling or sentence similarity. As the full power of these embeddings is not evident by the single word similarity task I have set up here, my hope is that this method can be of better use in downstream applications where the compositional structure of language is important.

References

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The Berkeley Framenet Project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A Neural Probabilistic Language Model. *Journal of Machine Learning Research*, 3:1137–1155.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database For Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A Sag. 2005. Minimal Recursion Semantics: An Introduction. *Research on Language and Computation*, 3(2-3):281–332.

Dipanjan Das, Desai Chen, André FT Martins, Nathan Schneider, and Noah A Smith. 2014. Frame-Semantic Parsing. *Computational Linguistics*, 40(1):9–56.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Chris Dyer. 2014. Notes on Noise Contrastive Estimation and Negative Sampling. *arXiv preprint arXiv:1410.8251*.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting Word Vectors to Semantic Lexicons. *NAACL*.

Charles Fillmore. 1982. Frame Semantics. *Linguistics in the Morning Calm*, pages 111–137.

Dan Flickinger, Stephan Oepen, and Gisle Ytrestøl. 2010. WikiWoods: Syntacto-Semantic Annotation for English Wikipedia. In *LREC*.

Dan Flickinger, Emily M Bender, and Stephan Oepen. 2014. Towards an Encyclopedia of Compositional Semantics: Documenting the Interface of the English Resource Grammar. In *LREC*, pages 875–881.

Dan Flickinger, Stephan Oepen, and Emily M Bender. In press. Sustainable Development and Refinement of Complex Linguistic Annotations at Scale. In Nancy Ide and James Pustejovsky, editors, *andbook of Linguistic Annotation Science*. Springer.

Dan Flickinger. 2000. On building a more effcient grammar by exploiting types. *Natural Language Engineering*, 6(01):15–28.

Dan Flickinger. 2011. Accuracy vs. Robustness in Grammar Engineering. *Language from a Cognitive Perspective: Grammar, Usage, and Processing*, pages 31–50.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *HLT-NAACL*, pages 758–764.

Yoav Goldberg and Joakim Nivre. 2012. A Dynamic Oracle for the Arc-Eager Dependency Parsing. In *COLING*, pages 959–976.

Michael U Gutmann and Aapo Hyvärinen. 2012. Noise-Contrastive Estimation of Unnormalized Statistical Models, with Applications to Natural Image Statistics. *The Journal of Machine Learning Research*, 13(1):307–361.

Hannaneh Hajishirzi, Leila Zilles, Daniel S Weld, and Luke S Zettlemoyer. 2013. Joint Coreference Resolution and Named-Entity Linking with Multi-Pass Sieves. In *EMNLP*, pages 289–299.

Andrej Karpathy. 2015. The Unreasonable Effectiveness of Recurrent Neural Networks. `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`.

Beth Levin. 1993. *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago press.

Omer Levy and Yoav Goldberg. 2014. Dependency-Based Word Embeddings. In *ACL*, pages 302–308.

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.

Yi Luan, Yangfeng Ji, Hannaneh Hajishirzi, and Boyang Li. 2016. Multiplicative Representations for Unsupervised Semantic Role Induction. In *ACL*. Association for Computational Linguistics.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, June.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient Estimation of Word Representations in Vector Space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An on-line lexical database. *International journal of lexicography*, 3(4):235–244.

George A Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *ACL*, pages 1003–1011. Association for Computational Linguistics.

Stephan Oepen, Dan Flickinger, Kristina Toutanova, and Christopher D Manning. 2004. LinGO Redwoods. *Research on Language and Computation*, 2(4):575–596.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press and CSLI Publications, Chicago, IL and Stanford, CA.

Karin Kipper Schuler. 2005. VerbNet: A broad-coverage, comprehensive verb lexicon.

Mo Yu and Mark Dredze. 2014. Improving Lexical Embeddings with Semantic Knowledge. In *ACL*, pages 545–550.