# Better Word Representations with Recursive Neural Networks for Morphology

**Minh-Thang Luong** **Richard Socher** **Christopher D. Manning**
Computer Science Department Stanford University, Stanford, CA, 94305
{lmthang, manning}@stanford.edu richard@socher.org

## Abstract

Vector-space word representations have been very successful in recent years at improving performance across a variety of NLP tasks. However, common to most existing work, words are regarded as independent entities without any explicit relationship among morphologically related words being modeled. As a result, rare and complex words are often poorly estimated, and all unknown words are represented in a rather crude way using only one or a few vectors. This paper addresses this shortcoming by proposing a novel model that is capable of building representations for morphologically complex words from their morphemes. We combine recursive neural networks (RNNs), where each morpheme is a basic unit, with neural language models (NLMs) to consider contextual information in learning morphologically-aware word representations. Our learned models outperform existing word representations by a good margin on word similarity tasks across many datasets, including a new dataset we introduce focused on rare words to complement existing ones in an interesting way.

## 1 Introduction

The use of word representations or word clusters pretrained in an unsupervised fashion from lots of text has become a key "secret sauce" for the success of many NLP systems in recent years, across tasks including named entity recognition, part-of-speech tagging, parsing, and semantic role labeling. This is particularly true in deep neural network models (Collobert et al., 2011), but it is also true in conventional feature-based models (Koo et al., 2008; Ratinov and Roth, 2009).

Deep learning systems give each word a distributed representation, i.e., a dense low-dimensional real-valued vector or an embedding. The main advantage of having such a distributed representation over word classes is that it can capture various dimensions of both semantic and syntactic information in a vector where each dimension corresponds to a latent feature of the word. As a result, a distributed representation is compact, less susceptible to data sparsity, and can implicitly represent an exponential number of word clusters.

However, despite the widespread use of word clusters and word embeddings, and despite much work on improving the learning of word representations, from feed-forward networks (Bengio et al., 2003) to hierarchical models (Morin, 2005; Mnih and Hinton, 2009) and recently recurrent neural networks (Mikolov et al., 2010; Mikolov et al., 2011), these approaches treat each full-form word as an independent entity and fail to capture the explicit relationship among morphological variants of a word.[1] The fact that morphologically complex words are often rare exacerbates the problem. Though existing clusterings and embeddings represent well frequent words, such as "distinct", they often badly model rare ones, such as "distinctiveness".

In this work, we use recursive neural networks (Socher et al., 2011b), in a novel way to model morphology and its compositionality. Essentially, we treat each morpheme as a basic unit in the RNNs and construct representations for morphologically complex words on the fly from their morphemes. By training a neural language model (NLM) and integrating RNN structures for complex words, we utilize contextual information in

[1] An almost exception is the word clustering of (Clark, 2003), which does have a model of morphology to encourage words ending with the same suffix to appear in the same class, but it still does not capture the relationship between a word and its morphologically derived forms.

an interesting way to learn morphemic semantics and their compositional properties. Our model has the capability of building representations for any new unseen word comprised of known morphemes, giving the model an infinite (if still incomplete) covered vocabulary.

Our learned representations outperform publicly available embeddings by a good margin on word similarity tasks across many datasets, which include our newly released dataset focusing on rare words (see Section 5). The detailed analysis in Section 6 reveals that our models can blend well syntactic information, i.e., the word structure, and the semantics in grouping related words.[2]

## 2 Related Work

Neural network techniques have found success in several NLP tasks recently such as sentiment analysis at the sentence (Socher et al., 2011c) and document level (Glorot et al., 2011), language modeling (Mnih and Hinton, 2007; Mikolov and Zweig, 2012), paraphrase detection (Socher et al., 2011a), discriminative parsing (Collobert, 2011), and tasks involving semantic relations and compositional meaning of phrases (Socher et al., 2012).

Common to many of these works is use of a distributed word representation as the basic input unit. These representations usually capture local cooccurrence statistics but have also been extended to include document-wide context (Huang et al., 2012). Their main advantage is that they can both be learned unsupervisedly as well as be tuned for supervised tasks. In the former training regiment, they are evaluated by how well they can capture human similarity judgments. They have also been shown to perform well as features for supervised tasks, e.g., NER (Turian et al., 2010).

While much work has focused on different objective functions for training single and multi-word vector representations, very little work has been done to tackle sub-word units and how they can be used to compute syntactic-semantic word vectors. Collobert et al. (2011) enhanced word vectors with additional character-level features such as capitalization but still can not recover more detailed semantics for very rare or unseen words, which is the focus of this work.

This is somewhat ironic, since working out correct morphological inflections was a very central problem in early work in the parallel distributed processing paradigm and criticisms of it (Rumelhart and McClelland, 1986; Plunkett and Marchman, 1991), and later work developed more sophisticated models of morphological structure and meaning (Gasser and Lee, 1990; Gasser, 1994), while not providing a compositional semantics nor working at the scale of what we present.

To the best of our knowledge, the work closest to ours in terms of handing unseen words are the factored NLMs (Alexandrescu and Kirchhoff, 2006) and the compositional distributional semantic models (DSMs) (Lazaridou et al., 2013). In the former work, each word is viewed as a vector of features such as stems, morphological tags, and cases, in which a single embedding matrix is used to look up all of these features.[3] Though this is a principled way of handling new words in NLMs, the by-product word representations, i.e. the concatenations of factor vectors, do not encode in them the compositional information (they are stored in the NN parameters). Our work does not simply concatenate vectors of morphemes, but rather combines them using RNNs, which captures morphological compositionality.

The latter work experimented with different compositional DSMs, originally designed to learn meanings of phrases, to derive representations for complex words, in which the base unit is the morpheme similar to ours. However, their models can only combine a stem with an affix and does not support recursive morpheme composition. It is, however, interesting to compare our neural-based representations with their DSM-derived ones and cross test these models on both our rare word similarity dataset and their nearest neighbor one, which we leave as future work.

Mikolov et al. (2013) examined existing word embeddings and showed that these representations already captured meaningful syntactic and semantic regularities such as the singular/plural relation that $x_{\text{apple}}$ - $x_{\text{apples}} \approx x_{\text{car}}$ - $x_{\text{cars}}$. However, we believe that these nice relationships will not hold for rare and complex words when their vectors are poorly estimated as we analyze in Section 6. Our model, on the other hand, explicitly represents these regularities through morphological structures of words.

[2]The rare word dataset and trained word vectors can be found at `http://nlp.stanford.edu/~lmthang/morphoNLM`.

[3](Collobert et al., 2011) used multiple embeddings, one per discrete feature type, e.g., POS, Gazeteer, etc.

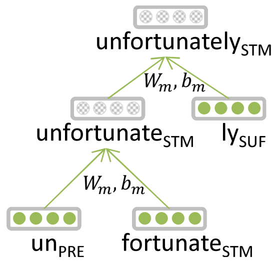

Figure 1: **Morphological Recursive Neural Network**. A vector representation for the word "unfortunately" is constructed from morphemic vectors: $un_{\texttt{pre}}$, $fortunate_{\texttt{stm}}$, $ly_{\texttt{suf}}$. Dotted nodes are computed on-the-fly and not in the lexicon.

## 3 Morphological RNNs

Our morphological Recursive Neural Network (*morphoRNN*) is similar to (Socher et al., 2011b), but operates at the morpheme level instead of at the word level. Specifically, morphemes, the minimum meaning-bearing unit in languages, are modeled as real-valued vectors of parameters, and are used to build up more complex words. We assume access to a dictionary of morphemic analyses of words, which will be detailed in Section 4.

Following (Collobert and Weston, 2008), distinct morphemes are encoded by column vectors in a morphemic embedding matrix $\boldsymbol{W_e} \in \mathbb{R}^{d\times|\mathbb{M}|}$, where $d$ is the vector dimension and $\mathbb{M}$ is an ordered set of all morphemes in a language.

As illustrated in Figure 1, vectors of morphologically complex words are gradually built up from their morphemic representations. At any local decision (a dotted node), a new parent word vector ($\boldsymbol{p}$) is constructed by combining a stem vector ($\boldsymbol{x}_{\texttt{stem}}$) and an affix vector ($\boldsymbol{x}_{\texttt{affix}}$) as follow:

$$\boldsymbol{p} = f(\boldsymbol{W_m}[\boldsymbol{x}_{\texttt{stem}};\boldsymbol{x}_{\texttt{affix}}] + \boldsymbol{b_m}) \qquad (1)$$

Here, $\boldsymbol{W_m} \in \mathbb{R}^{d\times 2d}$ is a matrix of morphemic parameters while $\boldsymbol{b_m} \in \mathbb{R}^{d\times 1}$ is an intercept vector. We denote an element-wise activation function as $f$, such as $\tanh$. This forms the basis of our morphoRNN models with $\boldsymbol{\theta} = \{\boldsymbol{W_e}, \boldsymbol{W_m}, \boldsymbol{b_m}\}$ being the parameters to be learned.

### 3.1 Context-insensitive Morphological RNN

Our first model examines how well morphoRNNs could construct word vectors simply from the morphemic representation *without referring to any context information*. Input to the model is a reference embedding matrix, i.e. word vectors trained by an NLM such as (Collobert and Weston, 2008) and (Huang et al., 2012). By assuming that these reference vectors are right, the goal of the model is to construct new representations for morphologically complex words from their morphemes that closely match the corresponding reference ones.

Specifically, the structure of the context-insensitive morphoRNN (*cimRNN*) is the same as the basic morphoRNN. For learning, we first define a cost function $s$ for each word $x_i$ as the squared Euclidean distance between the newly-*constructed* representation $\boldsymbol{p}_c(x_i)$ and its *reference* vector $\boldsymbol{p}_r(x_i)$: $s(x_i) = \|\boldsymbol{p}_c(x_i) - \boldsymbol{p}_r(x_i)\|_2^2$.

The objective function is then simply the sum of all individual costs over $N$ training examples, plus a regularization term, which we try to minimize:

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} s(x_i) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|_2^2 \qquad (2)$$

### 3.2 Context-sensitive Morphological RNN

The cimRNN model, though simple, is interesting to attest if morphemic semantics could be learned solely from an embedding. However, it is limited in several aspects. Firstly, the model has no chance of improving representations for rare words which might have been poorly estimated. For example, "distinctness" and "unconcerned" are very rare, occurring only 141 and 340 times in Wikipedia documents, even though their corresponding stems "distinct" and "concern" are very frequent (35323 and 26080 respectively). Trying to construct exactly those poorly-estimated word vectors might result in a bad model with parameters being pushed in wrong directions.

Secondly, though word embeddings learned from an NLM could, in general, blend well both the semantic and syntactic information, it would be useful to explicitly model another kind of syntactic information, the word structure, as we train our embeddings. Motivated by these limitations, we propose a context-sensitive morphoRNN (*csmRNN*) which integrates RNN structures into NLM training, allowing for contextual information being taken into account in learning morphemic compositionality. Specifically, we adopt the NLM training approach proposed in (Collobert et al., 2011) to learn word embeddings, but build representations for complex words from their morphemes. During learning, updates at the top level of the neural network will be back-propagated all the way till the morphemic layer.

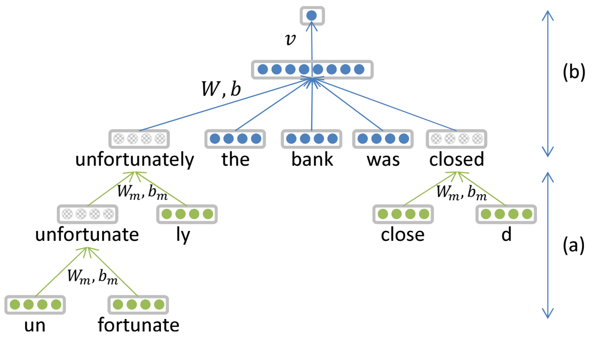

Figure 2: **Context-sensitive morphological RNN** has two layers: (a) the *morphological* RNN, which constructs representations for words from their morphemes and (b) the *word-based* neural language which optimizes scores for relevant ngrams.

Structure-wise, we stack the NLM on top of our morphoRNN as illustrated in Figure 2. Complex words like "unfortunately" and "closed" are constructed from their morphemic vectors, $un_{\mathtt{pre}}$ + $fortunate_{\mathtt{stm}}$ + $ly_{\mathtt{suf}}$ and $close_{\mathtt{stm}}$ + $d_{\mathtt{suf}}$, whereas simple words[4], i.e. stems, and affixes could be looked up from the morphemic embedding matrix $\boldsymbol{W_e}$ as in standard NLMs. Once vectors of all complex words have been built, the NLM assigns a score for each ngram $n_i$ consisting of words $x_1, \ldots, x_n$ as follows:

$$s\left(n_i\right) = \boldsymbol{v}^\top f(\boldsymbol{W}[\boldsymbol{x}_1; \ldots; \boldsymbol{x}_n] + \boldsymbol{b})$$

Here, $\boldsymbol{x}_j$ is the vector representing the word $x_j$. We follow (Huang et al., 2012) to use a simple feed-forward network with one $h$-dimensional hidden layer. $\boldsymbol{W} \in \mathbb{R}^{h \times nd}$, $\boldsymbol{b} \in \mathbb{R}^{h \times 1}$, and $\boldsymbol{v} \in \mathbb{R}^{h \times 1}$ are parameters of the NLM, and $f$ is an element-wise activation function as in Eq. (1). We adopt a ranking-type cost in defining our objective function to minimize as below:

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{N} \max\{0, 1 - s\left(n_i\right) + s\left(\overline{n}_i\right)\} \quad (3)$$

Here, $N$ is the number of all available ngrams in the training corpus, whereas $\overline{n}_i$ is a "corrupted" ngram created from $n_i$ by replacing its last word with a random word similar in spirit to (Smith and Eisner, 2005). Our model parameters are $\boldsymbol{\theta} = \{\boldsymbol{W_e}, \boldsymbol{W_m}, \boldsymbol{b_m}, \boldsymbol{W}, \boldsymbol{b}, \boldsymbol{v}\}$.

Such a ranking criterion influences the model to assign higher scores to valid ngrams than to invalid ones and has been demonstrated in (Collobert et al., 2011) to be both efficient and effective in learning word representations.

[4]"fortunate", "the", "bank", "was", and "close".

## 3.3 Learning

Our models alternate between two stages: (1) *forward pass* – recursively construct morpheme trees (cimRNN, csmRNN) and language model structures (csmRNN) to derive scores for training examples and (2) *back-propagation pass* – compute the gradient of the corresponding object function with respect to the model parameters.

For the latter pass, computing the objective gradient amounts to estimating the gradient for each individual cost $\frac{\partial s(x)}{\partial \boldsymbol{\theta}}$, where $x$ could be either a word (cimRNN) or an ngram (csmRNN). We have the objective gradient for the cimRNN derived as:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{i=1}^{N} \frac{\partial s\left(x_i\right)}{\partial \boldsymbol{\theta}} + \lambda \boldsymbol{\theta}$$

In the case of csmRNN, since the objective function in Eq. (3) is not differentiable, we use the subgradient method (Ratliff et al., 2007) to estimate the objective gradient as:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{i:1-s(n_i)+s(\overline{n}_i)>0} -\frac{\partial s\left(n_i\right)}{\partial \boldsymbol{\theta}} + \frac{\partial s\left(\overline{n}_i\right)}{\partial \boldsymbol{\theta}}$$

Back-propagation through structures (Goller and Küchler, 1996) is employed to compute the gradient for each individual cost with similar formulae as in (Socher et al., 2010). Unlike their RNN structures over sentences, where each sentence could have an exponential number of derivations, our morphoRNN structure per word is, in general, deterministic. Each word has a single morphological tree structure which is constructed from the main morpheme (the stem) and gradually appended affixes in a fixed order (see Section 4 for more details). As a result, both our forward and backward passes over morphological structures are efficient with no recursive calls implementation-wise.

## 4 Unsupervised Morphological Structures

We utilize an unsupervised morphological segmentation toolkit, named Morfessor by Creutz and Lagus (2007), to obtain segmentations for words in our vocabulary. Morfessor segments words in two stages: (a) it recursively splits words to minimize an objective inspired by the minimum description length principle and (b) it labels morphemes with tags `pre` (prefixes), `stm` (stems), and `suf` (suffixes) using hidden Markov models.

Morfessor captures a general word structure of the form $(\texttt{pre}^*\ \texttt{stm}\ \texttt{suf}^*)^+$, which is handy for words in morphologically rich languages like Finnish or Turkish. However, such general form is currently unnecessary in our models as the morphoRNNs assume input of the form $\texttt{pre}^*\ \texttt{stm}\ \texttt{suf}^*$ for efficient learning of the RNN structures: a stem is always combined with an affix to yield a new stem.[5] We, thus, postprocess as follows:

(1) Restrict segmentations to the form $\texttt{pre}^*\ \texttt{stm}\{1,2\}\ \texttt{suf}^*$: allow us to capture compounds.

(2) Split hyphenated words *A*-*B* as $A_{\texttt{stm}}\ B_{\texttt{stm}}$.

(3) For a segmentation with two stems, $\texttt{pre}^*\ A_{\texttt{stm}}\ B_{\texttt{stm}}\ \texttt{suf}^*$, we decide if one could be a main stem while the other could functions as an affix.[6] Otherwise, we reject the segmentation. This will provide us with more interesting morphemes such as $\mathit{al}_{\texttt{pre}}$ in Arabic names (al-jazeera, al-salem) and $\mathit{related}_{\texttt{suf}}$ in compound adjectives (health-related, government-related).

(4) To enhance precision, we reject a segmentation if it has either an affix or an unknown stem (not a word by itself) whose type count is below a predefined *threshold*[7].

The final list of affixes produced is given in Table 1. Though generally reliable, our final segmentations do contain errors, most notably non-compositional ones, e.g. $\mathit{de}_{\texttt{pre}}\ \mathit{fault}_{\texttt{stm}}\ \mathit{ed}_{\texttt{suf}}$ or $\mathit{re}_{\texttt{pre}}\ \mathit{turn}_{\texttt{stm}}\ \mathit{s}_{\texttt{suf}}$. With a sufficiently large number of segmentation examples, we hope that the model would be able to pick up general trends from the data. In total, we have about 22K complex words out of a vocabulary of 130K words.

Examples of words with interesting affixes are given in Table 2. Beside conventional affixes, non-conventional ones like "0" or "mc" help further categorize rare or unknown words into meaningful groups such as measurement words or names.

| **Prefixes** | **Suffixes** |
|---|---|
| 0 *al* all *anti auto co counter cross de dis* electro end *ex* first five focus four half high hyper ill *im in inter ir* jan jean long low market mc micro mid *multi* neuro newly no *non* off one *over post pre pro re* second *self semi* seven short six state *sub super* third three top *trans* two *un under* uni well | *able al ally* american *ance ate ation* backed bank based born controlled *d* dale down *ed en er es* field ford free *ful* general head *ia ian ible ic* in *ing isation ise ised ish ism ist ity ive ization ize ized izing* land led *less* ling listed *ly* made making man *ment ness* off on out owned related *s* ship shire style ton town up us ville wood |

Table 1: List of prefixes and suffixes discovered – conventional affixes in English are italicized.

| **Affix** | **Words** |
|---|---|
| 0 | 0-acre, 0-aug, 0-billion, 0-centistoke |
| anti | anti-immigrant, antipsychotics |
| counter | counterexample, counterinsurgency |
| hyper | hyperactivity, hypercholesterolemia |
| mc | mcchesney, mcchord, mcdevitt |
| bank | baybank, brockbank, commerzbank |
| ford | belford, blandford, carlingford |
| land | adventureland, bodoland, bottomland |
| less | aimlessly, artlessness, effortlessly |
| owned | bank-owned, city-owned disney-owned |

Table 2: Sample affixes and corresponding words.

## 5 Experiments

As our focus is in learning morphemic semantics, we do not start training from scratch, but rather, initialize our models with existing word representations. In our experiments, we make use of two publicly-available embeddings (50-dimensional) provided by(Collobert et al., 2011) (denoted as *C&W*)[8] and Huang et al. (2012) (referred as *HSMN*)[9].

Both of these representations are trained on Wikipedia documents using the same ranking-type cost function as in Eq. (3). The latter further utilizes global context and adopts a multi-prototype approach, i.e. each word is represented by multiple vectors, to better capture word semantics in various contexts. However, we only use their single-prototype embedding[10] and as we train, we

[5]When multiple affixes are present, we use a simple heuristic to first merge suffixes into stems and then combine prefixes. Ideally, we would want to learn and generate an order for such combination, which we leave for future work.

[6]We first aggregate type counts of pairs (*A*, left) and (*B*, right) across all segmentations with two stems. Once done, we label A as `stm` and B as `suf` if count (*B*, right) $> 2 \times$ count (*A*, left), and conversely, we label them as $A_{\texttt{pre}}\ B_{\texttt{stm}}$ if count (*A*, left) $> 2 \times$ count(*B*, right). Our rationale was that affixes occur more frequently than stems.

[7]Set to 15 and 3 for affixes and stems respectively.

[8]`http://ronan.collobert.com/senna/`.

[9]`http://www-nlp.stanford.edu/~ehhuang/`.

[10]The embedding obtained just before the clustering step to build multi-prototype representation.

do not consider the global sentence-level context information. It is worth to note that these aspects of the HSMN embedding – incorporating global context and maintaining multiple prototypes – are orthogonal to our approach, which would be interesting to investigate in future work.

For the context-sensitive morphoRNN model, we follow Huang et al. (2012) to use the April 2010 snapshot of the Wikipedia corpus (Shaoul and Westbury, 2010). All paragraphs containing non-roman characters are removed while the remaining text are lowercased and then tokenized. The resulting clean corpus contains about 986 million tokens. Each digit is then mapped into 0, i.e. 2013 will become 0000. Other rare words not in the vocabularies of C&W and HSMN are mapped to an UNKNOWN token, and we use <s> and </s> for padding tokens representing the beginning and end of each sentence.

Follow (Huang et al., 2012)'s implementation, which our code is based on initially, we use 50-dimensional vectors to represent morphemic and word embeddings. For cimRNN, the regularization weight $\lambda$ is set to $10^{-2}$. For csmRNN, we use 10-word windows of text as the local context, 100 hidden units, and no weight regularization.

## 5.1 Word Similarity Task

Similar to (Reisinger and Mooney, 2010) and (Huang et al., 2012), we evaluate the quality of our morphologically-aware embeddings on the popular *WordSim-353* dataset (Finkelstein et al., 2002), WS353 for short. In this task, we compare correlations between the similarity scores given by our models and those rated by human.

To avoid overfitting our models to a single dataset, we benchmark our models on a variety of others including *MC* (Miller and Charles, 1991), *RG* (Rubenstein and Goodenough, 1965), *SCWS**[11] (Huang et al., 2012), and our new rare word (*RW*) dataset (details in §5.1.1). Information about these datasets are summarized in Table 3

We also examine these datasets from the "rareness" aspect by looking at distributions of words across frequencies as in Table 4. The first bin counts unknown words in each dataset, while the remaining bins group words based on their frequencies extracted from Wikipedia documents. It is interesting to observe that WS353, MC, RG contain very frequent words and have few complex words (only WS353 has).[12] SCWS* and RW have a more diverse set of words in terms of frequencies and RW has the largest number of unknown and rare words, which makes it a challenging dataset.

[11]SCWS* is a modified version of the Stanford's contextual word similarities dataset. The original one utilizes surrounding contexts in judging word similarities and includes pairs of identical words, e.g. financial *bank* vs. river *bank*. We exclude these pairs and ignore the provided contexts.

| | **pairs** | **type** | **raters** | **scale** | **Complex words** token | type |
|---|---|---|---|---|---|---|
| WS353 | 353 | 437 | 13-16 | 0-10 | 24 | 17 |
| MC | 30 | 39 | 38 | 0-4 | 0 | 0 |
| RG | 65 | 48 | 51 | 0-4 | 0 | 0 |
| SCWS* | 1762 | 1703 | 10 | 0-10 | 190 | 113 |
| **RW (new)** | 2034 | 2951 | 10 | 0-10 | 987 | 686 |

Table 3: **Word similarity datasets** and their statistics: number of pairs/raters/type counts as well as rating scales. The number of complex words are shown as well (both type and token counts). RW denotes our new rare word dataset.

| | **All words** | **Complex words** |
|---|---|---|
| WS353 | 0 \| 0 / 9 / 87 / 341 | 0 \| 0 / 1 / 6 / 10 |
| MC | 0 \| 0 / 1 / 17 / 21 | 0 \| 0 / 0 / 0 / 0 |
| RG | 0 \| 0 / 4 / 22 / 22 | 0 \| 0 / 0 / 0 / 0 |
| SCWS* | 26 \| 2 / 140 / 472 / 1063 | 8 \| 2 / 22 / 44 / 45 |
| RW | 801 \| 41 / 676 / 719 / 714 | 621 \| 34 / 311 / 238 / 103 |

Table 4: **Word distribution by frequencies** – distinct words in each dataset are grouped based on frequencies and counts are reported for the following bins : unknown | [1, 100] / [101, 1000] / [1001, 10000] / [10001, $\infty$). We report counts for all words in each dataset as well as complex ones.

### 5.1.1 Rare Word Dataset

As evidenced in Table 4, most existing word similarity datasets contain frequent words and few of them possesses enough rare or morphologically complex words that we could really attest the expressiveness of our morphoRNN models. In fact, we believe a good embedding in general should be able to learn useful representations for not just frequent words but also rare ones. That motivates us to construct another dataset focusing on rare words to complement existing ones.

Our dataset construction proceeds in three stages: (1) select a list of rare words, (2) for each of the rare words, find another word (not necessarily rare) to form a pair, and (3) collect human judgments on how similar each pair is.

[12]All these counts are with respect to the vocabulary list in the C&W embedding (we obtain similar figures for HSMN).

| | **(5, 10]** | **(10, 100]** | **(100, 1000]** |
|---|---|---|---|
| **un-** | untracked unrolls undissolved | unrehearsed unflagging unfavourable | unprecedented unmarried uncomfortable |
| **-al** | apocalyptical traversals bestowals | acoustical extensional organismal | directional diagonal spherical |
| **-ment** | obtainment acquirement retrenchments | discernment revetment rearrangements | confinement establishment management |

| **word$_1$** | untracked | unflagging | unprecedented | apocalyptical | organismal | diagonal | obtainment | discernment | confinement |
|---|---|---|---|---|---|---|---|---|---|
| **word$_2$** | inaccessible | constant | new | prophetic | system | line | acquiring | knowing | restraint |

Table 5: **Rare words** (top) – *word*$_1$ by affixes and frequencies and sample **word pairs** (bottom).

**Rare word selection**: our choices of rare words (*word*$_1$) are based on their frequencies – based on five bins (5, 10], (10, 100], (100, 1000], (1000, 10000], and the affixes they possess. To create a diverse set of candidates, we randomly select 15 words for each configuration (a frequency bin, an affix). At the scale of Wikipedia, a word with frequency of 1-5 is most likely a junk word, and even restricted to words with frequencies above five, there are still many non-English words. To counter such problems, each word selected is required to have a non-zero number of synsets in WordNet(Miller, 1995).

Table 5 (top) gives examples of rare words selected and organized by frequencies and affixes. It is interesting to find out that words like *obtainment* and *acquirement* are extremely rare (not in traditional dictionaries) but are perfectly understandable. We also have less frequent words like *revetment* from French or *organismal* from biology.

**Pair construction**: following (Huang et al., 2012), we create pairs with interesting relationships for each *word*$_1$ as follow. First, a WordNet synset of *word*$_1$ is randomly selected, and we construct a set of candidates which connect to that synset through various relations, e.g., hypernyms, hyponyms, holonyms, meronyms, and attributes. A *word*$_2$ is then randomly selected from these candidates, and the process is repeated another time to generate a total of two pairs for each *word*$_1$. Sample word pairs are given in Table 5 in which *word*$_2$ includes mostly frequent words, implying a balance of words in terms of frequencies in our dataset. We collected 3145 pairs after this stage

**Human judgment**: we use Amazon Mechanical Turk to collect 10 human similarity ratings on a scale of [0, 10] per word pair.[13] Such procedure has been demonstrated by Snow et al. (2008) in replicating ratings for the MC dataset, achieving close inter-annotator agreement with expert raters. Since our pairs contain many rare words which are challenging even to native speakers, we ask raters to indicate for each pair if they do not know the first word, the second word, or both. We use such information to collect reliable ratings by either discard pairs which many people do not know or collect additional ratings to ensure we have 10 ratings per pair.[14] As a result, only 2034 pairs are retained.

## 5.2 Results

We evaluate the quality of our morphoRNN embeddings through the word similarity task discussed previously. The Spearman's rank correlation is used to gauge how well the relationship between two variables, the similarity scores given by the NLMs and the human annotators, could be described using a monotonic function.

Detailed performance of the morphoRNN embeddings trained from either the HSMN or the C&W embeddings are given in Table 7 for all datasets. We also report baseline results (rows *HSMN, C&W*) using these initial embeddings alone, which interestingly reveals strengths and weaknesses of existing embeddings. While HSMN is good for datasets with frequent words (WS353, MC, and RG), its performances for those with more rare and complex words (SCWS$^*$ and RW) are much inferior than those of C&W, and vice versa. Additionally, we consider two slightly more competitive baselines (rows *+stem*) based on the morphological segmentation of unknown words: instead of using a universal vector representing all unknown words, we use vectors representing the stems of unknown words. These baselines yield slightly better performance for the SCWS$^*$ and RW datasets while the trends we mentioned earlier remain the same.

Our first model, the context-insensitive morphoRNN (cimRNN), outperforms its corresponding baseline significantly over the rare word

[13]We restrict to only US-based workers with 95% approval rate and ask for native speakers to rate 20 pairs per hit.

[14]In our later experiments, an aggregated rating is derived for each pair. We first discard ratings not within one standard deviation of the mean, and then estimate a new mean from the remaining ones to use as an aggregated rating.

| **Words** | **C&W** | **C&W + cimRNN** | **C&W + csmRNN** |
|---|---|---|---|
| commenting | insisting insisted focusing hinted | republishing accounting expounding | commented comments criticizing |
| comment | commentary rant statement remark | commentary rant statement remark | rant commentary statement anecdote |
| distinctness | morphologies pesawat clefts | modality indistinct tonality spatiality | indistinct distinctiveness largeness uniqueness |
| distinct | different distinctive broader narrower | different distinctive broader divergent | divergent diverse distinctive homogeneous |
| unaffected | unnoticed dwarfed mitigated | disaffected unconstrained uninhibited | undesired unhindered unrestricted |
| affected | caused plagued impacted damaged | disaffected unaffected mitigated disturbed | complicated desired constrained reasoned |
| unaffect | $\emptyset$ | affective affecting affectation unobserved | affective affecting affectation restrictive |
| affect | exacerbate impacts characterize | affects affectation exacerbate characterize | decrease arise complicate exacerbate |
| heartlessness | $\emptyset$ | fearlessness vindictiveness restlessness | depersonalization terrorizes sympathizes |
| heartless | merciless sadistic callous mischievous | merciless sadistic callous mischievous | sadistic callous merciless hideous |
| heart | death skin pain brain life blood | death skin pain brain life blood | death brain blood skin lung mouth |
| saudi-owned | avatar mohajir kripalani fountainhead | saudi-based somaliland al-jaber | saudi-based syrian-controlled syrian-backed |
| short-changed | kindled waylaid endeared peopled | conformal conformist unquestionable | short-termism short-positions self-sustainable |

Table 6: **Nearest neighbors**. We show morphologically related words and their closest words in different representations ("unaffect" is a pseudo-word; $\emptyset$ marks no results due to unknown words).

| | **WS353** | **MC** | **RG** | **SCWS*** | **RW** |
|---|---|---|---|---|---|
| HSMN | 62.58 | 65.90 | 62.81 | 32.11 | 1.97 |
| +stem | 62.58 | 65.90 | 62.81 | 32.11 | 3.40 |
| +cimRNN | 62.81 | 65.90 | 62.81 | 32.97 | 14.85 |
| +csmRNN | **64.58** | **71.72** | **65.45** | **43.65** | **22.31** |
| C&W | 49.77 | 57.37 | 49.30 | 48.59 | 26.75 |
| +stem | 49.77 | 57.37 | 49.30 | **49.05** | 28.03 |
| +cimRNN | 51.76 | 57.37 | 49.30 | 47.00 | 33.24 |
| +csmRNN | **57.01** | **60.20** | **55.40** | 48.48 | **34.36** |

Table 7: **Word similarity task** – shown are Spearman's rank correlation coefficient ($\rho \times 100$) between similarity scores assigned by neural language models and by human annotators. *stem* indicates baseline systems in which unknown words are represented by their stem vectors. *cimRNN* and *csmRNN* refer to our context insensitive and sensitive morphological RNNs respectively.

dataset. The performance is constant for MC and RG (with no complex words) and modestly improved for MC (with some complex words – see Table 4). This is expected since the cimRNN model only concerns about reconstructing the original embedding (while learning word structures), and the new representation mostly differs at morphologically complex words. For SCWS*, the performance, however, decreases when training with C&W, which perhaps is due to: (a) the baseline performance of C&W for SCWS* is competitive and (b) the model trades off between learning syntactics (the word structure) and capturing semantics, which requires context information.

On the other hand, the context-sensitive morphoRNN (csmRNN) consistently improves correlations over the cimRNN model for all datasets, demonstrating the effectiveness of using surrounding contexts in learning both morphological syntactics and semantics. It also outperforms the corresponding baselines by a good margin for all datasets (except for SCWS*). This highlights the fact that our method is reliable and potentially applicable for other embeddings.

## 6 Analysis

To gain a deeper understanding of how our morphoRNN models have "moved" word vectors around, we look at nearest neighbors of several complex words given by various embeddings, where cosine similarity is used as a distance metric. Examples are shown in Table 6 for three representations: C&W and the context-insensitive/sensitive morphoRNN models trained on the C&W embedding.[15]

Syntactically, it is interesting to observe that the cimRNN model could well enforce structural agreement among related words. For example, it returns *V-ing* as nearest neighbors for "commenting" and similarly, *JJ-ness* for "fearlessness", an unknown word that C&W cannot handle. However, for those cases, the nearest neighbors are badly unrelated.

On the semantic side, we notice that when structural agreement is not enforced, the cimRNN model tends to cluster words sharing the same stem together, e.g., rows with words of the form *__affect__*.[16] This might be undesirable when we want to differentiate semantics of words sharing the same stem, e.g. "affected" and "unaffected".

The csmRNN model seems to balance well between the two extremes (syntactic and semantic) by taking into account contextual information

[15]Results of HSMN-related embeddings are not shown, but similar trends follow.

[16]"unaffect" is a pseudo-word that we inserted.

when learning morphological structures. It returns neighbors of the same structure *un___ed* for “unaffected”, but does not include any negation of “affected” in the top 10 results when “affected” is queried.[17] Even better, the answers for “distinctness” have blended well both types of results.

## 7 Conclusion

This paper combines recursive neural networks (RNNs) and neural language models (NLMs) in a novel way to learn better word representations. Each of these components contributes to the learned syntactic-semantic word vectors in a unique way. The RNN explicitly models the morphological structures of words, i.e., the syntactic information, to learn morphemic compositionality. This allows for better estimation of rare and complex words and a more principled way of handling unseen words, whose representations could be constructed from vectors of known morphemes.

The NLMs, on the other hand, utilize surrounding word contexts to provide further semantics to the learned morphemic representations. As a result, our context-sensitive morphoRNN embeddings could significantly outperform existing embeddings on word similarity tasks for many datasets. Our analysis reveals that the model could blend well both the syntactic and semantic information in clustering related words. We have also made available a word similarity dataset focusing on rare words to complement existing ones which tend to include frequent words.

Lastly, as English is still considered limited in terms of morphology, our model could potentially yield even better performance when applied to other morphologically complex languages such as Finnish or Turkish, which we leave for future work. Also, even within English, we expect our model to be value to other domains, such as bio-NLP with complicated but logical taxonomy.

## Acknowledgements

We thank the anonymous reviewers for their feedback and Eric Huang for making his various pieces of code available to us as well as answering our questions on different datasets. Stanford University gratefully acknowledges the support of the Defense Advanced Research Projects Agency (DARPA) Deep Exploration and Filtering of Text (DEFT) Program under Air Force Research Laboratory (AFRL) contract no. FA8750-13-2-0040 and the DARPA Broad Operational Language Translation (BOLT) program through IBM. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the US government.

[17]“disaffected” is ranked 5th for the first query while “affecting” occurs at position 8 for the latter.

## References

Andrei Alexandrescu and Katrin Kirchhoff. 2006. Factored neural language models. In *NAACL*.

Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Alexander Clark. 2003. Combining distributional and morphological information for part of speech induction. In *EACL*.

R. Collobert and J. Weston. 2008. A unified architecture for natural language processing: deep neural networks with multitask learning. In *ICML*.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

R. Collobert. 2011. Deep learning for efficient discriminative parsing. In *AISTATS*.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4:3:1–3:34.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. 2002. Placing search in context: the concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Michael Gasser and Chan-Do Lee. 1990. A short-term memory architecture for the learning of morphophonemic rules. In *NIPS*.

Michael Gasser. 1994. Acquiring receptive morphology: A connectionist model. In *ACL*.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*.

C. Goller and A. Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. *IEEE Transactions on Neural Networks*, 1:347–352.

E. H. Huang, R. Socher, C. D. Manning, and A. Y. Ng. 2012. Improving word representations via global context and multiple word prototypes. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *ACL*.

Angeliki Lazaridou, Marco Marelli, Roberto Zamparelli, and Marco Baroni. 2013. Compositional-ly derived representations of morphologically complex words in distributional semantics. In *ACL*.

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. In *SLT*.

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*.

Tomas Mikolov, Stefan Kombrink, Lukas Burget, Jan Cernocký, and Sanjeev Khudanpur. 2011. Extensions of recurrent neural network language model. In *ICASSP*.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *NAACL-HLT*.

George A. Miller and Walter G. Charles. 1991. Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6(1):1–28.

G.A. Miller. 1995. WordNet: A Lexical Database for English. *Communications of the ACM*.

Andriy Mnih and Geoffrey Hinton. 2007. Three new graphical models for statistical language modelling. In *ICML*.

Andriy Mnih and Geoffrey Hinton. 2009. A scalable hierarchical distributed language model. In *NIPS*.

Frederic Morin. 2005. Hierarchical probabilistic neural network language model. AIstats'05. In *AISTATS*.

K. Plunkett and V. Marchman. 1991. U-shaped learning and frequency effects in a multi-layered perceptron: implications for child language acquisition. *Cognition*, 38(1):43–102.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*.

Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. 2007. Online subgradient methods for structured prediction.

Joseph Reisinger and Raymond J. Mooney. 2010. Multi-prototype vector-space models of word meaning. In *NAACL*.

Herbert Rubenstein and John B. Goodenough. 1965. Contextual correlates of synonymy. *Communications of the ACM*, 8(10):627–633.

David E. Rumelhart and James L. McClelland. 1986. On learning the past tenses of English verbs. In J. L. McClelland, D. E. Rumelhart, and PDP Research Group, editors, *Parallel Distributed Processing. Volume 2: Psychological and Biological Models*, pages 216–271. MIT Press.

Cyrus Shaoul and Chris Westbury. 2010. The Westbury lab wikipedia corpus. Edmonton, AB: University of Alberta.

Noah A. Smith and Jason Eisner. 2005. Contrastive estimation: training log-linear models on unlabeled data. In *ACL*.

Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. 2008. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *EMNLP*.

Richard Socher, Christopher Manning, and Andrew Ng. 2010. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *NIPS*2010 Workshop on Deep Learning and Unsupervised Feature Learning*.

R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. 2011a. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *NIPS*.

R. Socher, Cliff C. Lin, A. Y. Ng, and C. D. Manning. 2011b. Parsing natural scenes and natural language with recursive neural networks. In *ICML*.

R. Socher, J. Pennington, E. H. Huang, A. Y. Ng, and C. D. Manning. 2011c. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *EMNLP*.

R. Socher, B. Huval, C. D. Manning, and A. Y. Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *EMNLP*.

J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*.