INCORPORATING BOTH DISTRIBUTIONAL AND RELATIONAL SEMANTICS IN WORD REPRESENTATIONS

Daniel Fried∗
Department of Computer Science
University of Arizona
Tucson, Arizona, USA
dfried@email.arizona.edu

Kevin Duh
Graduate School of Information Science
Nara Institute of Science and Technology
Ikoma, Nara, JAPAN
kevinduh@is.naist.jp

Abstract
We investigate the hypothesis that word representations ought to incorporate both distributional and relational semantics. To this end, we employ the Alternating Direction Method of Multipliers (ADMM), which flexibly optimizes a distributional objective on raw text and a relational objective on WordNet. Preliminary results on knowledge base completion, analogy tests, and parsing show that word representations trained on both objectives can give improvements in some cases.

1 Introduction

We are interested in algorithms for learning vector representations of words. Recent work has shown that such representations, also known as word embeddings, can successfully capture the semantic and syntactic regularities of words (Mikolov et al., 2013a) and improve the performance of various Natural Language Processing systems, including information extraction (Turian et al., 2010) [Wang & Manning 2013], parsing (Socher et al., 2013a), and semantic role labeling (Collobert et al., 2011).

Although many kinds of representation learning algorithms have been proposed so far, they are all essentially based on the same premise of distributional semantics (Harris, 1954), embodied by J. R. Firth’s dictum: “You shall know a word by the company it keeps.” For example, the models of (Bengio et al., 2003) [Schwenk 2007] Collobert et al. 2011 [Mikolov et al., 2013b] Mnih & Kavukcuoglu 2013 train word representations by exploiting the context window around the word. Intuitively, these algorithms learn to map words with similar context to nearby points in vector space.

However, distributional semantics is by no means the only theory of word meaning. Relational semantics, exemplified by WordNet (Miller, 1995), defines a word by its relation with other words. Relations such as synonymy, hypernymy, and meronymy (Cruse, 1986) create a graph that links words in terms of our world knowledge and psychological predispositions. For example, stating a relation like “dog is-a mammal” gives a precise hierarchy between the two words, in a way that is very different from the distributional similarities observable from corpora. Arguably, the vector representation of “dog” ought to be close to that of “mammal”, regardless of their distributional contexts.

We believe both distributional and relational semantics are valuable for word representations. Our goal is to explore how to combine these complementary approaches into a unified learning algorithm. We thus employ a general representation learning algorithm based on the Alternating Direction Method of Multipliers (ADMM) (Boyd et al., 2011) for jointly optimizing both distributional and relational objectives. Its advantages include (a) flexibility in incorporating arbitrary objectives, and (b) relative ease of implementation.

∗Currently at the University of Cambridge.
In the following, we first discuss objectives for independently learning distributional semantics (§2.1) or relational semantics (§2.2). The ADMM framework that optimizes both objectives is described in §3 and analyzed in §4. To test whether our embeddings are widely applicable, we evaluate three specific ADMM instantiations (each using different ways of incorporating relational semantics) on a wide range of tasks (§5).

2 Objectives for Representation Learning

2.1 Distributional Semantics Objective

A standard way to implement distributional semantics in representation learning is the Neural Language Model (NLM) of Collobert et al. (2011). Each word \( i \) in the vocabulary is associated with a \( d \)-dimensional vector \( w_i \in \mathbb{R}^d \), the word’s embedding. An \( n \)-length sequence of words \( (i_1, i_2, \ldots, i_n) \) is represented as a vector \( x \) by concatenating the vector embeddings for each word, \( x = [w_{i_1} ; w_{i_2} ; \ldots ; w_{i_n}] \). This vector \( x \) is then scored by feeding it through a two-layer neural network with \( h \) hidden nodes:

\[
S_{NLM}(x) = u^T(f(Ax + b))
\]

where \( A \in \mathbb{R}^{h \times (nd)} \) is the weight matrix and \( b \in \mathbb{R}^h \) is the bias vector for the hidden layer, \( u \in \mathbb{R}^h \) is the weight vector for the output layer, and \( f \) is the sigmoid \( f(t) = 1/(1 + e^{-t}) \).

The layer parameters and word embeddings of this model are trained using noise contrastive estimation (Smith & Eisner, 2005; Gutmann & Hyvärinen, 2010; Mnih & Kavukcuoglu, 2013). A sequence of text from the training corpus is corrupted by replacing a word in the sequence with a random word sampled from the vocabulary, providing an implicit negative training example.

To train the network so that correct sequences receive a higher score than corrupted sequences, the hinge loss function is optimized:

\[
L_{NLM}(x, x_c) = \max(0, 1 - S_{NLM}(x) + S_{NLM}(x_c))
\]

The word embeddings \( w \) and network layer parameters \( A, u, b \) are trained with backpropagation, using stochastic gradient descent (SGD) over n-grams in the training corpus. We are concerned with the learned embeddings and disregard the other network parameters after training.

2.2 Relational Semantics Objective

Methods for learning word representations based on relational semantics have only recently been explored. We first present a simple new objective based on WordNet graph distance (§2.2.1), then discuss two recent proposals that directly model relation types (§2.2.2). While our objectives focus on relational semantics in WordNet, they are extensible to other kinds of relational data, including knowledge bases like Freebase.

2.2.1 Graph Distance

In this approach, we aim to train word embeddings such that the distance between word embeddings in the vector space is a function of the distance between corresponding entities in WordNet. The primary entities in WordNet are synonym sets, or synsets. Each synset is a group of words representing one lexical concept. WordNet contains a set of relationships between these synsets, forming a directed graph where vertices are synsets and relationships are edges. The primary relationship is formed by the HYPERNYM (Is-A) relationship.

By treating these HYPERNYM relationships as undirected edges between synsets, we approximate semantic relatedness between synsets as the length of the shortest path between two synsets in the graph. We add a common root node at the base of all hypernym trees so that the synset graph is connected, and adopt the similarity function of Leacock & Chodorow (1998):

\[
\text{SynSim}(s_i, s_j) = -\log \frac{\text{len}(s_i, s_j)}{2 \times \max_{s \in \text{WordNet}} \text{depth}(s)}
\]

where \( \text{len}(s_i, s_j) \) is the length of the shortest undirected hypernym path between synsets \( s_i \) and \( s_j \) in the graph, and \( \text{depth} \) returns the distance from the root of the hypernym hierarchy to a given synset.
Since there is a many-to-many relationship between words and WordNet synsets, and embeddings for words, not synsets, are desired, we define the similarity between two words to be the maximum similarity between their corresponding synsets, if both words have associated synsets, and undefined otherwise:

\[ \text{WordSim}(i, j) = \max_{s_i \in \text{syn}(i), s_j \in \text{syn}(j)} \text{SynSim}(s_i, s_j) \]  

(4)

where syn(i) is the set of synsets corresponding to word i.

To integrate WordNet similarity with word embeddings, we define the following Graph Distance loss, \( L_{GD} \). For a word pair \((i, j)\), we encourage the cosine similarity between their embeddings \( \mathbf{v}_i \) and \( \mathbf{v}_j \) to match that of a scaled version of \( \text{WordSim}(i, j) \):

\[ L_{GD}(i, j) = \left( \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{||\mathbf{v}_i||_2||\mathbf{v}_j||_2} - [a \times \text{WordSim}(i, j) + b] \right)^2 \]  

(5)

where \( a \) and \( b \) are parameters that scale \( \text{WordSim}(i, j) \) to be of the same range as the cosine similarity between embeddings.

SGD is used to train the word embeddings as well as the scalar parameters \( a \) and \( b \). Pairs of words with defined \( \text{WordSim} \) (i.e. if both words have synsets) are sampled from the vocabulary and used as a single training instance. Details of the sampling process are presented in [4].

2.2.2 Existing Relational Objectives

We are aware of two recent approaches from the Knowledge Base literature which, in addition to representing words (entities) with vector embeddings, directly represent a knowledge base’s relations as operations in the vector embedding space. These models both take as input a tuple \((u_l, R, v_r)\) representing a possible relationship of type \( R \) between words \( v_l \) and \( v_r \), and assign a score to the relationship.

The TransE model of [Bordes et al. (2013)] represents relationships as translations in the vector embedding space. For two words \( v_l \) and \( v_r \), if the relationship \( R \) holds, i.e. \((v_l, R, v_r)\) is true, then the corresponding embeddings \( v_l, v_r \in \mathbb{R}^d \) should be close after translation by the relation vector \( R \in \mathbb{R}^d \). The score of a relationship tuple is the similarity between \( v_l + R \) and \( v_r \), measured by the negative of the residual:

\[ S_{\text{TransE}}(v_l, R, v_r) = -||v_l + R - v_r||_2 \]  

(6)

[Socher et al. (2013b)] introduce a Neural Tensor Network (NTN) model that allows modeling of the interaction between embeddings using tensors. The NTN model is a two-layer neural network with \( h \) hidden units and a bilinear tensor layer directly relating embeddings. This provides a more expressive model than TransE, but also requires training a larger number of parameters for each relation. The scoring function for a relation \( R \) is

\[ S_{\text{NTN}}(v_l, R, v_r) = \mathbf{U}^T f \left( \mathbf{v}_l^T \mathbf{W}_R \mathbf{v}_r + \mathbf{V}_R \begin{bmatrix} \mathbf{v}_l \\ \mathbf{v}_r \end{bmatrix} + \mathbf{b}_R \right) \]  

(7)

where \( f \) is the sigmoid non-linearity applied elementwise, \( \mathbf{U} \in \mathbb{R}^h \) is the weight vector of the output layer, and \( \mathbf{W}_R \in \mathbb{R}^{d \times d \times h} \), \( \mathbf{V}_R \in \mathbb{R}^{h \times 2d \times l} \) and \( \mathbf{b}_R \in \mathbb{R}^k \) are a tensor, matrix, and bias vector respectively for relationship \( R \).

As in the Neural Language Model, embeddings and parameters for these relational models are trained using contrastive estimation and SGD, using the hinge loss as defined in [2], where \( S_{\text{NLM}} \) is replaced by either the \( S_{\text{TransE}} \) or \( S_{\text{NTN}} \) scoring function on tuples.

3 Joint Objective Optimization by ADMM

We aim to train a set of word embeddings that, along with the corresponding model parameters, satisfy both the distributional modeling objective (Sec. 2.1) and one of the relational modeling

\[ ^1 \]As in the graph distance objective, we must map from synsets to words. In each SGD iteration, a relationship tuple \((s_l, R, s_r)\) is sampled from WordNet such that synsets \( s_l \) and \( s_r \) contain words in the vocabulary. One word is sampled for each synset from the set of words in the vocabulary contained in the synset, producing a tuple \((w_l, R, w_r)\). This is the correct tuple to be used in training, treating words as entities for the relational model. To produce the corrupted tuple, one of \( w_l, R, \) or \( w_r \) is randomly replaced.
objectives (Sec. 2.2). We adopt the Alternating Direction Method of Multipliers (ADMM) approach (Boyd et al., 2011). Rather than use the same set of embeddings to evaluate the loss functions for both the distributional and relational objectives, the embeddings are split into two sets, one for each objective, and allowed to vary independently. An augmented Lagrangian penalty term is added to constrain the corresponding embeddings for each word to have minimal difference. The advantage of this approach is that existing methods for optimizing each objective independently can be re-used, leading to a flexible and easy-to-implement algorithm.

We describe the ADMM formulation using graph distance as the WordNet modeling objective, but a similar formulation holds when using other relational objectives. Let \( \mathbf{w} \) be the set of word embeddings \( \{ \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{N'} \} \) for the distributional modeling objective, and \( \mathbf{v} \) be the set of word embeddings \( \{ \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{N''} \} \) for the relational modeling objective, where \( N' \) is the number of words in the model vocabulary of the corpus, and \( N'' \) is the number of words in the model vocabularies of WordNet. Let \( I \) be the set of \( N \) words that occur in both the corpus and WordNet, i.e. the intersection. Then we define a set of vectors \( \mathbf{y} = \{ \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N \} \), which correspond to Lagrange multipliers, to penalize the difference \( (\mathbf{w}_i - \mathbf{v}_i) \) between sets of embeddings for each word \( i \) in the joint vocabulary \( I \):

\[
L_P(\mathbf{w}, \mathbf{v}) = \sum_{i \in I} \left( \mathbf{y}_i^\top (\mathbf{w}_i - \mathbf{v}_i) \right) + \frac{\rho}{2} \left( \sum_{i \in I} \| \mathbf{w}_i - \mathbf{v}_i \|_2^2 \right) \tag{8}
\]

In the first term, \( \mathbf{y} \) has same dimensionality as \( \mathbf{w} \) and \( \mathbf{v} \), so a scalar penalty is maintained for each entry in every embedding vector. The second residual penalty term with hyperparameter \( \rho \) is added to avoid saddle points. Later we shall see that \( \rho \) can be viewed as a step-size during the update of \( \mathbf{y} \).

Finally, this augmented Lagrangian term (Eq. 8) is added to the sum of the loss terms for each objective (Eq. 2 and Eq. 5). Let \( \theta = (\mathbf{u}, \mathbf{A}, \mathbf{b}) \) be the parameters of the language modeling objective, and \( \phi = (a, b) \) be the parameters of the WordNet graph distance objective. The final loss function we optimize becomes:

\[
L = L_{NLM}(\mathbf{w}, \theta) + L_{GD}(\mathbf{v}, \phi) + L_P(\mathbf{w}, \mathbf{v}) \tag{9}
\]

The ADMM algorithm proceeds by repeating the following three steps until convergence:

1. Perform stochastic gradient descent on \( \mathbf{w} \) and \( \theta \) to minimize \( L_{NLM} + L_P \), with all other parameters fixed.

2. Perform stochastic gradient descent on \( \mathbf{v} \) and \( \phi \) to minimize \( L_{GD} + L_P \), with all other parameters fixed.

3. For all embeddings \( i \) corresponding to words in both the n-gram and relational training sets, update the constraint vector \( \mathbf{y}_i \):

\[
\mathbf{y}_i := \mathbf{y}_i + \rho (\mathbf{w}_i - \mathbf{v}_i) \tag{10}
\]

Since \( L_{NLM} \) and \( L_{GD} \) share no parameters, the gradient descent step for \( \mathbf{w} \) in Step 1 does not depend on \( \phi \) (the parameters of the relational objective). The derivative of \( L_{NLM}(\mathbf{w}, \theta) + L_P(\mathbf{w}, \mathbf{v}) \) with respect to \( \mathbf{w}_i \) is simply the derivative of \( L_{NLM}(\mathbf{w}, \theta) \) plus \( \mathbf{y}_i + \rho (\mathbf{w}_i - \mathbf{v}_i) \); the second term acts like a bias term to make \( \mathbf{w}_i \) closer to \( \mathbf{v}_i \). Similarly, the gradient descent step for \( \mathbf{v} \) does not depend on \( \theta \), so Step 2 can be optimized easily. In Step 3, a large difference \( (\mathbf{w}_i - \mathbf{v}_i) \) causes \( \mathbf{y}_i \) to become large, and therefore increases the constraint for the two sets of embeddings to be similar in both Steps 1 and 2.

We note that it is possible to introduce a weight parameter \( \alpha \in [0, 1] \) into the joint loss function (Eq. 9) to prioritize either \( L_{NLM} \) or \( L_{GD} \):

\[
L = \alpha L_{NLM}(\mathbf{w}, \theta) + (1 - \alpha) L_{GD}(\mathbf{v}, \phi) + L_P(\mathbf{w}, \mathbf{v})
\]

Empirically we found that while the difference between using joint objective compared to single objective is large, the exact value of \( \alpha \) does not significantly change the results; all experiments here use equal weighting.
Figure 1: Analysis of ADMM behavior as training iteration progresses, for varying values of $\rho$.

4 Data Setup and Analysis

The distributional objective $L_{NLM}$ is trained using 5-grams from the Google Books English corpus, distributed by UC Berkeley in the Web 1T format. This corpus contains over 180 million 5-gram types and 31 billion 5-gram tokens. In our experiments, 5-grams are preprocessed by lowercasing all words. The top 50,000 unigrams by frequency are used as the vocabulary. All less-frequently occurring words are replaced with a token, RARE, which has its own vector space embedding. In each ADMM iteration, a block of 100,000 n-grams is sampled from the corpus. Each n-gram in the block (and a corrupted, noise-contrastive version of the n-gram, see §2.1) is used to perform gradient descent on the distributional loss function $L_{NLM}$ or its ADMM equivalent.

http://tomato.banatao.berkeley.edu:8080/berkeleym_binaries/
Training data for the relational objective varies depending on whether the graph distance objective (GD) or one of the two relational objectives (NTN or TransE) is used. When the graph distance objective is used, word pairs with defined graph distance (e.g. words that are contained in WordNet synsets) are randomly sampled and used as training instances. In each ADMM iteration, 100,000 words are sampled with replacement from the vocabulary. For each sampled word \( w \), up to 5 other words \( v \) with defined graph distance to word \( w \) are sampled from the vocabulary. Each pair \( w, v \) is then used as a training instance for gradient descent on the \( L_{GD} \) loss function (\( \text{2.2.1} \)) or its ADMM equivalent.

When either NTN or TransE is used as the relational objective, WordNet relationship tuples \((s_1, R, s_2)\) where synsets \( s_1 \) and \( s_2 \) contain words in the vocabulary, are used as the training instances for stochastic gradient descent using noise contrastive estimation (\( \text{2.2.2} \)). For comparison with the existing work of Socher et al. (2013b), we use their dataset, which contains training, development, and testing splits for 11 WordNet relationships (Table 1). The entire training set is presented to the network in randomized order, one instance at a time, during each iteration of training.

We next provide an analysis of the behavior of the ADMM joint model \( (L_{NLM} + L_{GD}) \) on the training set in Figure 1. Figure 1(a) plots the learning curve by training iteration using varying values of the \( \rho \) hyperparameter. Although establishing convergence guarantees for non-convex loss functions for ADMM is theoretically still an open question (e.g. \( L_{NLM} \) is a non-convex multi-layer neural net), we empirically observe convergence on our dataset for various values of \( \rho \). Further, in accordance with previous works (Boyd et al., 2011), ADMM attains a reasonable objective value relatively quickly in a few iterations; our loss converges around 100 iterations.\(^1\) Figure 1(b) shows the change in the mean norms of the Lagrange multipliers \( ||y_i||_2 \). The magnitude of these norms indicates the degree to which the \( w_i \) and \( v_i \) vectors are being constrained in the current ADMM iteration. The norm gradually increases, indicating the tightening of constraints in each iteration. As expected, larger values of \( \rho \) lead to faster increases of \( ||y_i||_2 \). Finally, Figure 1(c) shows the normalized difference between the resulting sets of embeddings \( w \) and \( v \), which decreases as desired.\(^2\)

As we perform SGD on ever-more data, the norms of \( w \) and \( v \) generally increase. A conventional solution is to add the L2 norms of \( w \) and \( v \) as additional regularizers in the objective function. We found that, on the knowledge base completion task (\( \text{5} \)), L2 regularization decreased the performance of all ADMM models, but slightly increased the performance of the NTN and TransE single objective models; on the analogy test tasks, it decreased the performance of the GD, NLM, and all ADMM models, but increased the performance of NTN and TransE. Since regularization hurt performance for the majority of models, and the evaluation results converge regardless, we use unregularized models in all experiments reported here.

5 Task-Specific Evaluation

We now compare the embeddings learned with different objectives on three different tasks. For all experiments, we use 50-dimensional embeddings taken from iteration 1000 of training, with \( \rho = 0.05 \) for ADMM.

5.1 Knowledge Base Completion

Models trained using either the NTN or TransE relational modeling objective (\( \text{2.2.2} \)) learn a vector-space representation of relationships in WordNet. We use the methodology and datasets of Socher et al. (2013b) to evaluate the models’ ability to classify relationship triplets as correct or incorrect.\(^3\)

This relation classification is useful for “completing” or “extending” a knowledge base with new facts. The testing set consists of correct relationship tuples that are present in WordNet, and incorrect tuples created by randomly switching entities from correct tuples. A development set is used to determine threshold scores \( T_R \) for each relation that maximize classification accuracy when tuples are present in WordNet.\(^4\)

\(^1\)On a 3.3Hz Xeon CPU, this took about 9 hours for ADMM, not more than the 7 hours for independent \( L_{NLM} \) and 3 hours for independent \( L_{GD} \) objectives combined.

\(^2\)The reason for the peak around iteration 50 in Figure 1(c) is that the embeddings begin with similar random initializations, so initially differences are small; as ADMM starts to see more data, \( w \) and \( v \) diverge, but converge eventually as \( y \) become large.

\(^3\)The training set in Figure 1(a) plots the learning curve by training iteration using varying values of the \( \rho \) hyperparameter.
Table 1: Knowledge Base Completion results: counts by WordNet relation type and test classification accuracies for NTN, TransE, and joint objectives.

| Relationship Type          | Number of Relationships | Classification Accuracies by Model on Test Set |
|---------------------------|-------------------------|-----------------------------------------------|
|                           | Train | Dev | Test | NTN   | NTN + NLM | TransE | TransE + NLM |
| HASINSTANCE                | 36178 | 1632 | 6334 | 80.35 | 76.66     | 79.98  | 80.20        |
| TYPEOF                    | 30556 | 1334 | 5504 | 80.17 | 83.79     | 85.35  | 84.77        |
| MEMBERHOLONYM             | 9146  | 614  | 2346 | 80.95 | 88.61     | 90.57  | 90.36        |
| MEMBERMERONYM             | 9223  | 522  | 2268 | 88.44 | 83.95     | 81.34  | 81.74        |
| PARTOF                    | 6600  | 334  | 1266 | 85.27 | 80.17     | 82.06  | 83.80        |
| HASPART                   | 6139  | 296  | 1348 | 80.72 | 74.40     | 76.33  | 77.52        |
| DOMAINREGION              | 4227  | 168  | 592  | 68.58 | 68.41     | 70.94  | 72.46        |
| SYSETDOMAINTOPIC          | 3976  | 144  | 622  | 91.15 | 90.67     | 89.06  | 90.35        |
| SUBORDINATEINSTANCEOF     | 3778  | 146  | 650  | 94.00 | 92.15     | 91.53  | 92.46        |
| SIMILARTO                 | 1659  | 4    | 42   | 64.28 | 57.14     | 54.76  | 54.76        |
| DOMAINOPIC                | 1099  | 24   | 116  | 67.24 | 62.06     | 68.10  | 72.41        |
| Overall                   | 112581| 5218 | 21088| 80.95 | 81.27     | 82.87  | 83.10        |

\[(v_1, R, v_2) \text{ having } S(v_1, R, v_2) \geq T_R \text{ are classified as correct, and tuples having a score lower than } T_R \text{ are classified as incorrect.}\]

Models trained using a joint objective (NLM with either TransE or NTN) are evaluated by taking the v set of vectors (those learned for the relational objective) and passing these through the relational objective function to score a given tuple. The test accuracies are shown in Table 1. Overall, the NTN baseline achieves 80.95\% accuracy while the joint objective NTN+NLM improves it to 81.27\%. Similarly, TransE+NLM (83.10\%) outperforms the TransE baseline (82.87\%). We conclude that ADMM can give small albeit noticeable improvement to TransE and NTN. Table 1 also shows the accuracies by relation type. We note that TransE+NLM performs at least as well as the TransE single objective across all categories except TYPEOF and MEMBERHOLONYM.

We also experimented with using the average of w and v vectors (rather than v itself after ADMM) for relation classification. This produced lower overall accuracies for the joint model (75.38\% for NTN+NLM and 71.73\% for TransE+NLM), implying that differences between w and v may still be important in actual tasks.

5.2 Analogy Tests for Relational Similarity

SemEval-2012 Task 2 (Jurgens et al., 2012) is a relational similarity task similar to SAT-style analogy questions. The task is to score word pairs by the degree to which they belong in a relation class defined by a set of example word pairs. For example, the relation class REVERSE contains the example pairs (attack, defend) and (buy, sell). There are 69 testing relation categories, each with three or four example word pairs. In the evaluation, the model is shown a number of testing relation pairs in each category and scores each testing pair according to its similarity to the example relation pairs. These similarity scores are then compared to human similarity judgements. This is an useful task to test whether the positioning of learned embeddings in vector space leads to some meaningful semantics.

Following Zhila et al. (2013), we evaluate the embeddings in this task on their ability to represent relations as translations in the vector space. A given relation pair \((word_1, word_2)\) is represented as the vector difference between the two words, \(w_2 - w_1\), where \(w_1\) and \(w_2\) are the embeddings of words \(word_1\) and \(word_2\). Similarities between the example relations and the relations to be scored are computed using cosine distance of these resulting embedding representations. One evaluation metric is the Spearman’s correlation coefficient between the similarity scores output by the model and the scores assigned to pairs by human judges. The second metric is the MaxDiff accuracy, which involves choosing both the most similar and least similar example pairs to a given target pair from a set of four or five pre-defined choices.

\(^3\)GD is not evaluated since it does not model relation types.

\(^6\)Our NTN results differ from those reported in Socher et al. (2013b), likely due to differences in the optimizer (SGD vs L-BFGS), embedding size, and the use or lack of regularizer.
Figure 2: Detailed comparison of correlation and accuracy for various embeddings on SemEval-2012 Task 2, by relationship category.
Table 2: Analogy Test results: Comparison of single and joint objective embeddings. For a random baseline, accuracy=.31 and correlation=.018. ([Jurgens et al., 2012]).

A summary of the results is shown in Table 2. We observe:

1. NLM achieves scores competitive with the recurrent neural language models used in ([Zhila et al., 2013]), which had a maximum accuracy of 0.41 and correlation of 0.23.
2. GD by itself also achieves comparable scores, with 0.42 accuracy and 0.28 correlation. It is interesting to note that independent distributional and relational objectives achieve similar results for this task.
3. The joint objective does not appear to help in this task, however. E.g., GD+NLM does not outperform GD; while NTN+NLM outperforms NTN, it does not outperform NLM.

To analyze this result further, we show the scores by relation category in Figure 2. Despite NLM, GD, GD+NLM, and NTN+NLM achieving similar top scores overall, we observe that the scores by category are considerably varied. We conclude our current objectives, either distribution or relational, are too coarse-grained to reliably address the analogy test. In particular, the relation categories for this task do not correspond to the relation types on WordNet. Since it is infeasible to expect a large WordNet-like resource that is annotated in the particular categories for this task, an objective that includes some form of unsupervised relation clustering may be necessary.

5.3 Dependency Parsing

Dependency parsing experiments are performed on the SANCL2012 “Parsing the Web” data ([Petrov & McDonald, 2012]). The setup is to train a parser on news domain (Wall Street Journal) and evaluate on out-of-domain web text. Our goal is to see whether the performance of a standard parser can be improved by simply using embeddings as additional features. This can be seen as a kind of semi-supervised feature learning ([Koo et al., 2008]).

Following ([Wu et al., 2013]), we first cluster the embeddings using k-means, then incorporate the cluster ids as features. The reasoning is that discrete cluster ids are easier to incorporate into existing parsers as conjunctions of features. We use the standard first-order MST parser. For simplicity, we only attempt to cluster the embeddings into \( k = 64 \) clusters and report results on the development set; a more extensive experiment involving multiple \( k \) and model selection is left as future work.

Table 3 shows the labeled attachment scores (LAS), i.e. the accuracy of predicting both correct syntactic attachment and relation label for each word. We observe:

1. Incorporating embeddings from joint objective training always helps; all of these embeddings improve upon the case of no embeddings (None) in all five domains. This is a nice result considering that our embeddings are trained on Google Books, not SANCL, and have a 9-13% token out-of-vocabulary rate on the data.
2. In contrast, improvements from embeddings trained from a single relational objective are mixed, and are in general poorer than those trained from a single distributional objective (NLM). This suggests distributional information may be more effective for this task.
3. The best results are achieved by the joint models NLM+GD (average LAS of 76.18) and NLM+NTN (76.14). The improvement over NLM (76.03) is not large, but we believe this is
| Embedding | AVERAGE | Answers | Emails | Newsgroups | Reviews | Weblogs |
|-----------|---------|---------|--------|------------|---------|---------|
| None      | 75.83   | 73.40   | 73.72  | 74.88      | 75.46   | 81.70   |
| NTN       | 75.85   | 73.56   | 73.48  | 74.88      | 75.53   | 81.58   |
| TransE    | 75.86   | 73.30   | 73.69  | 75.09      | 75.68   | 81.74   |
| GD        | 75.90   | 73.54   | 73.73  | 75.03      | 75.65   | 81.55   |
| NLM       | 76.03   | 73.65   | 73.77  | 74.96      | 75.81   | 81.94   |
| NLM + NTN | 76.14   | **73.81** | 73.87 | 75.09      | **75.92** | 82.01   |
| NLM + TransE | 76.01 | 73.50 | **73.89** | 75.09 | 75.53 | 82.02 |
| NLM + GD  | **76.18** | 73.78 | 73.69 | **75.39** | 75.76 | **82.28** |

Table 3: Comparison of parsers using different embedding features. Labeled Arc Score (LAS) on five web domains and their average are reported.

...a promising result nevertheless; it implies that our joint objective does indeed complement the strong results of distributional semantics.

6 CONCLUSIONS AND FUTURE WORK

We advocate for a word representation learning algorithm that jointly implements both distributional and relational semantics. Our first contribution is an investigation of the ADMM algorithm, which flexibly combines multiple objectives. We show that ADMM converges quickly and is an effective method for combining multiple sources of information and linguistic intuitions into word representations. Note that other approaches for combining objectives besides ADMM are possible, including direct gradient descent on the joint objective, or concatenation of word representations individually optimized on independent objectives. A comparison of various approaches for multi-objective optimization in learning word representations, where the optimization space is riddled with local optima, is worthwhile as future work.

The second contribution is a preliminary evaluation of three specific instantiations of ADMM, combining the NLM distributional objective with Graph Distance, TransE, or NTN relational objectives. In both the knowledge base completion and dependency parsing tasks, we demonstrate that the combined objective provides promising minor improvements compared to the single objective case. In the analogy task, we show that the combined objective is comparable to the single objective, and learns a very different kind of word representation.

To the best of our knowledge, some recent work (Xu et al. 2014; Yu & Dredze 2014; Faruqui et al. 2014) explored similar motivations as ours. The main differences are in their optimization methods (i.e. gradient descent directly on the joint objective is used in (Xu et al. 2014; Yu & Dredze 2014), while Faruqui et al. (2014) introduces a post-processing graph-based method) as well as alternate relational objectives, e.g. Yu & Dredze (2014) formulate a skip-gram objective where word embeddings are trained to predict other words they share relations with. A detailed comparison on the same datasets would be beneficial to understand the impact of these design choices.

Compared to the large body of existing work in word representations (e.g. LSA, Deerwester et al. 1990), ESA (Gabrilovich & Markovitch 2007), SDS (Mitchell & Lapata 2008), the promise of recent learning-based approaches is that they enable a flexible definition of optimization objectives. As future work, we hope to further explore objectives beyond distributional and relational semantics and understand what objectives work best for each target task or application.

ACKNOWLEDGMENTS

This work is supported by a Microsoft Research CORE Grant and JSPS KAKENHI Grant Number 26730121. D.F. was supported by the Flinn Scholarship during the course of this work. We thank Haixun Wang, Jun’ichi Tsujii, Tim Baldwin, Yuji Matsumoto, and several anonymous reviewers for helpful discussions at various stages of the project.
REFERENCES

Bengio, Yoshua, Ducharme, Réjean, Vincent, Pascal, and Jauvin, Christian. A neural probabilistic language models. *JMLR*, 2003.

Bordes, Antoine, Usunier, Nicolas, Garcia-Duran, Alberto, Weston, Jason, and Yakhnenko, Oksana. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*, pp. 2787–2795, 2013.

Boyd, Stephen, Parikh, Neal, Chu, Eric, Peleato, Borja, and Eckstein, Jonathan. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.

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

Cruse, Alan D. *Lexical Semantics*. Cambridge Univ. Press, 1986.

Deerwester, Scott, Dumais, Susan T., Furnas, George W., Landauer, Thomas K., and Harshman, Richard. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6), 1990.

Faruqui, Manaal, Dodge, Jesse, Jauhar, Sujay Kumar, Dyer, Chris, Hovy, Eduard H., and Smith, Noah A. Retrofitting word vectors to semantic lexicons. *CoRR*, abs/1411.4166, 2014. URL http://arxiv.org/abs/1411.4166

Gabrilovich, E. and Markovitch, S. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJCAI*, 2007.

Gutmann, Michael and Hyvärinen, Aapo. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *International Conference on Artificial Intelligence and Statistics*, pp. 297–304, 2010.

Harris, Zellig. Distributional structure. *Word*, 10(23):146–162, 1954.

Jurgens, David A, Turney, Peter D, Mohammad, Saif M, and Holyoak, Keith J. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pp. 356–364. Association for Computational Linguistics, 2012.

Koo, Terry, Carreras, Xavier, and Collins, Michael. Simple semi-supervised dependency parsing. In *Proceedings of ACL-08: HLT*, pp. 595–603, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/F/F08/F08-1068

Leacock, Claudia and Chodorow, Martin. Combining local context and WordNet similarity for word sense identification. *WordNet: An Electronic Lexical Database*, pp. 265–283, 1998.

Mikolov, Tomas, Yih, Wen-tau, and Zweig, Geoffrey. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 746–751, Atlanta, Georgia, June 2013a. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N13-1090

Mikolov, Tomáš, Sutskever, Ilya, Chen, Kai, Corrado, Greg, and Dean, Jeffrey. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013b.

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

Mitchell, Jeff and Lapata, Mirella. Vector-based models of semantic composition. In *ACL*, pp. 236–244, 2008.

Mnih, Andriy and Kavukcuoglu, Koray. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems 26 (NIPS 2013)*, 2013.
Petrov, Slav and McDonald, Ryan. Overview of the 2012 shared task on parsing the web. In Notes of the First Workshop on Syntactic Analysis of Non-Canonical Language (SANCL), 2012.

Schwenk, Holger. Continuous space language models. Computer Speech and Language, 21(3): 492–518, July 2007. ISSN 0885-2308. doi: 10.1016/j.csl.2006.09.003. URL http://dx.doi.org/10.1016/j.csl.2006.09.003

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

Socher, Richard, Bauer, John, Manning, Christopher D., and Ng, Andrew Y. Parsing with compositional vector grammars. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 455–465, Sofia, Bulgaria, August 2013a. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P13-1045

Socher, Richard, Chen, Danqi, Manning, Christopher D., and Ng, Andrew Y. Reasoning with neural tensor networks for knowledge base completion. In NIPS, 2013b.

Turian, Joseph, Ratinov, Lev-Arie, and Bengio, Yoshua. Word representations: A simple and general method for semi-supervise learning. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P10-1040

Wang, Mengqiu and Manning, Christopher D. Effect of non-linear deep architecture in sequence labeling. In Proceedings of the Sixth International Joint Conference on Natural Language Processing, pp. 1285–1291, Nagoya, Japan, October 2013. Asian Federation of Natural Language Processing. URL http://www.aclweb.org/anthology/I13-1183

Wu, Xianchao, Zhou, Jie, Sun, Yu, Liu, Zhanyi, Yu, Dianhai, Wu, Hua, and Wang, Haifeng. Generalization of words for chinese dependency parsing. In Proceedings of the 13th International Conference on Parsing Technologies (IWPT), 2013.

Xu, Chang, Bai, Yanlong, Bian, Jiang, Gao, Bin, Wang, Gang, Liu, Xiaoguang, and Liu, Tie-Yan. Re-net: A general framework for incorporating knowledge into word representations. In Proceedings of the ACM Conference on Information and Knowledge Management (CIKM), 2014.

Yu, Mo and Dredze, Mark. Improving lexical embeddings with semantic knowledge. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 545–550, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-2089

Zhila, Alisa, Yih, Wen-tau, Meek, Christopher, Zweig, Geoffrey, and Mikolov, Tomas. Combining heterogeneous models for measuring relational similarity. In NAACL-HLT, pp. 1000–1009, 2013.