Project Overview

Title

Entity Extraction using Condensed Words-to-Entity Vector Transformation

Members

Henry Chen, Johnny Tan

Abstract

Extracting entities in text and deriving their relationships from a knowledge graph is a principle task in applications to understanding context. Existing approaches often run a Named Entity Recognition (NER) model to extract the entities first, then link the extracted name using a knowledge graph such as FreeBase. These entity recognition models follow a statistical approach and fail to address ambiguity or unseen entities. To address this issue we first devise a projection method that projects a sequence of word vectors to its condensed vector form while retain its semantics. Then we propose a semi-supervised approach in finding the transformation functions that map word vectors generated from word2vec to their corresponding chunk-ed entity vector. By aggregating these mappings we aim to have a generalized function for each entity label and to apply it to unseen words to improve entity extraction. We evaluate these new entity vectors on the named entity recognition task and their word representation quality on analogy tasks.

Intellectual Merits

In order to better encapsulate the semantics of sequences of word vectors that might be potential entities, this project will be exploring and experimenting different ways to consolidate contextual embeddings of individual words into an augmented vector of the same dimension. We aim to devise a novel method to accumulate word semantics from sequences of embeddings with minimal lost in word representation. This project would then construct an estimated transformation matrix to map vectors, built from condensing sequence of word vectors, onto their closest corresponding entity vectors. We will be the first in attempt to address the problem of unseen entity using entity specific disambiguation. Determining the gold-labeled entity vector and evaluating our approach are the main challenges of this project. The first is addressed in a semi-supervised fashion by building our transformation matrix using phrase vectors and chunk-ed entity vectors as our initial gold entity vectors. For extrinsic evaluation we will test our approach on the named entity recognition task for performance. To evaluate the quality of our generated vectors we will use analogy and word similarity tasks.

Contributions

This project has several potential impacts in other Natural Language Processing tasks, such as Named Entity Recognition, Entity Disambiguation and Sub-categorization, Relation Extraction, and Attribute Extraction. By introducing a simpler way for entity extraction and constructing entity embedded vectors, the computational complexity of identifying relations between entities and attributes of target entities can be easily reduced. In addition the semantic information generated can be used as additional features for existing statistical approaches. Our novel projection method to condense a word vector from a sequence of individual word vectors will provide valuable semantic information for unseen named entity recognition. Not only will the results of this project be beneficial to unseen entity identification, but we also introduce an alternative approach of tackling the NER task with minimal supervised learning while taking complete advantage of distributional semantics

Entity Extraction using Condensed Words-to-Entity Vector Transformation

Henry Chen, Johnny Tan
{yche463, jtan25}@emory.edu
Department of Mathematics and Computer Sciecne at Emory University

1 Objectives

The main objectives of this research include:

- Devise a projection method to encapsulate and consolidate a sequence of word vector with minimal lost in its collective semantics.
- Determine a transformation matrix that maps a sequence of word vectors to its respective entity vector.
- Determine an efficient way to extract semantic entity information from entity vectors.

The novelty of this project lies in its approaches and applications of words-to-entity vector transformation. The three main objectives above tie closely with each other. In order to construct high-quality entity vector, we would have to first suggest a projection method that projects a sequence of word vectors to its condensed vector form while retaining most of its word representations. Rather than treating sequences of words as singleton tokens or creating phrase vectors by extracting their surrounding contextual distribution, our projection method attempts to accumulate the word representation and distributional semantic of individual word vectors given by word2vec. This is a crucial preliminary step for applying our words-to-entity vector transformation.

This project then proposes a semi-supervised approach in finding the transformation matrices that map word vectors generated from word2vec to their corresponding chunk-ed entity vector. We compare the approach of decomposing the words-to-entity vector transformation matrix through inverse matrix estimation to learning it via neural network. The initial set of training entity vectors will be generated from phrase vectors that exist only in the DBpedia ontology. These vectors that exist in the ontology should provide enough information to learn the rest of the phrase vectors with self-supervision. By learning a words-to-entity vector transformation matrix, this project introduces a more efficient way to use distributional semantic information for entity extraction. The combination of the projection method and the words-to-entity vector transformation gives flexibility and nondiscrimination for its input. This implies the possibility for semi-supervised or even completely unsupervised Named Entity Recognition (NER).

2 Background and Significance

2.1 Motivation

In recent years the increase of available information and computing power has made natural language processing to be one of the most active and growing research fields. To enable computers to comprehend and derive meaning from human input, a system must be able to tackle syntax, semantic, and knowledge extraction challenges. To face these challenges it is essential to improve core-level NLP tasks such as named-entity recognition or part-of-speech tagging. The standard paradigm of these components follow a statistical inference approach to weigh input features and express relative certainty of different predictions.

Although the use of statistical approaches yield reliable results, these systems will still fail to handle ambiguity or unseen words. This is largely due to the fact that the way human expression is ever-changing. To be more specific, entity recognition systems that are enriched by knowledge bases like Wikipedia will not be able to recognize an unseen entity name such as Golden Buddha (local restaurant) or an ambiguous name such as Lincoln (motor company). This research strives to address these concerns with a novel alternative to entity recognition using entity-based distributional semantics.

2.2 Distributional Semantics

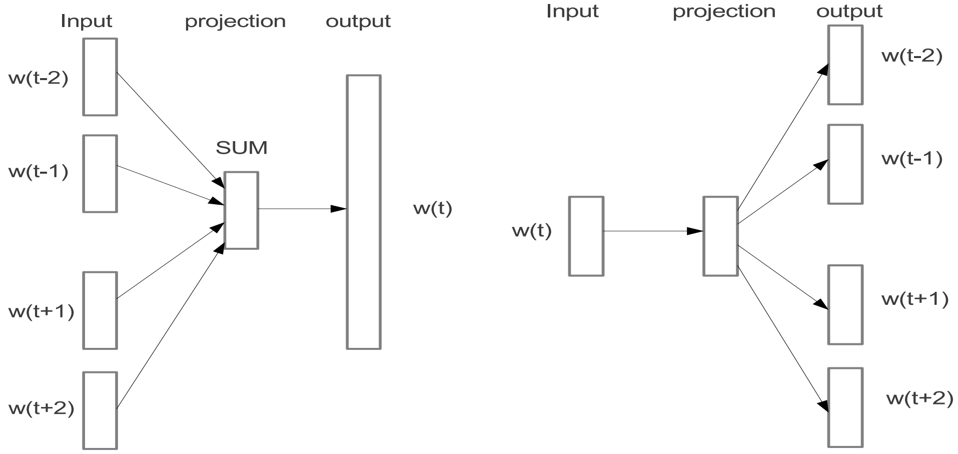
The field of distributional semantics which have recently gained a lot of attention have shown to be helpful for handling unseen words in several NLP tasks (Turian et al., 2010). Word vectors are created from quantifying and categorizing semantic similarities of words based on their distributional properties in large corpus. These vectors provide semantic information that can be used as features or can be clustered to form classes similar to Brown Clustering. For example the word representation of the entity ”New York Times” should be rather similar to that of ”Chicago Tribune”. Our research strives to take advantage of this similarity to a lower level. We seek to understand how the individual word vectors of ”New York Times” can be transformed to contain the semantic information that it is an entity (Organization). Then by extracting this transformation information for each entity type we hope to have a generalized mapping function. As a result when an unseen entity appears the generalized mapping functions are applied to it and the output should be most similar to one of the existing entity labels.

2.3 Previous Work

2.3.1 Word2Vec

Word2vec is a shallow word embedding model that learns to map each word in the vocabulary into a low-dimension continuous vector from their distributional properties observed in raw text. The model has two architectures, continuous bag-of-words (CBOW) and skip-gram. The CBOW architecture is trained to learn a target word t given the surrounding context words c with the goal of maximizing $p(t|c)$. The SGNS architecture differs in which it tries to predict the context words

$c_0..c_n$ given the target word t. The idea of negative sampling has been introduced to provide skip gram to be more efficient in training (SGNS) Mikolov et al. (2013b). In addition more finer-grained vectors are produced from training skip-gram on a large corpus as opposed to CBOW. This is because the CBOW architecture averages all the context words giving it a ”regularizing” effect and allowing it to do well when there is little data. The embeddings used in this project are generated from both architectures with the suggested parameters in Mikolov et al. (2013b). Below illustrates the two architectures of Word2vec.



(a) CBOW Architecture

(b) Skip Gram Architecture

Figure 1: Word2vec Models

2.3.2 Vector Projection

Unsupervised data-driven learning of word-embedded vector space using large quantity of unannotated data has been well-received and has become one of the most important technique in Natural Language Processing for constructing distributional semantics for individual vocabularies (Turney et al., 2010). Since word vectors generated from word2vec are in linear relationship with its input, they implicitly reflect the distribution around its context words. Thus these vectors have the property of additive composition, and it is possible to perform analogical reasoning through simple vector arithmetic (Mikolov et al., 2013b).

Word-embeddings and Vector Space Model (VSM) have been applied in tasks like Sentiment Analysis, Word Sense Disambiguation, etc. However, each word vector only capture the contextual distribution of one vocabulary, and thus the problem of representing collective semantics in phrase- or document-level arises. In order to utilize existing word-embeddings structure without designing new embeddings, recent researches have tried to project multiple word vectors onto one with the same vocabulary dimension. Mikolov et al. (2013a) takes the average word vector of window with specified size when constructing the input word vector using continuous bag of words. Vector projection has also been used in cross-lingual vocabulary mapping that projects one word vector onto another of the same word in a different language based off similarity of their distribution. Faruqui and Dyer (2014) creates projection vectors for different language pairs through Canonical Correlation Analysis in order to incorporate word-embedding information of different languages as features. This project will explore the use of average vector projection and canonical correlation analysis projection. In addition it will aim to create a new projection method to preserve as much semantic as possible from the context.

2.3.3 Inverse Matrix Estimation

Levy and Goldberg (2014) has shown that skip-gram with negative sampling is implicitly factorizing a word-context matrix, whose cells are the point wise mutual information (PMI) of the respective word and context pairs, shifted by a global constant. He then compares SGNS to his factorization using different global constants and alternative methods such as singular value decomposition. Instead of applying these matrix factorization methods to generating word embeddings, our project will apply these methods toward finding the transformation matrix that produces the entity vector. In addition to finding the inverse matrix by decomposition we will also attempt to learn the transformation matrix using an artificial neural network.

There has considerable amount of research in sparse inverse covariance matrix estimation. Most of these approaches apply the concept of Gaussian Maximum Likelihood Estimation. Various graphical model and linear programming methods have been used to approximate the optimal covariance or inverse sparse matrix. Hsieh et al. (2011) proposed a log-determinant algorithm that utilizes quadratic approximation to compute the matrix. Such approach is super-linear and guarantees quadratic convergence for the estimation. Furthermore, the paper shows that with improvement data arrangement and caching, the computational complexity of this process can be reduced from $O(n^2)$ to $O(n)$.

The main distinction of the inverse matrix estimation in this project is to compare and apply different methods to factorize the transformation matrix to our neural net approach of learning it.

3 Proposed Work

3.1 Methodology

The figure below illustrates the complete flow of the steps this project takes to semi-supervised generate the words-to-entity transformation vector while learning potential unseen entity vectors.

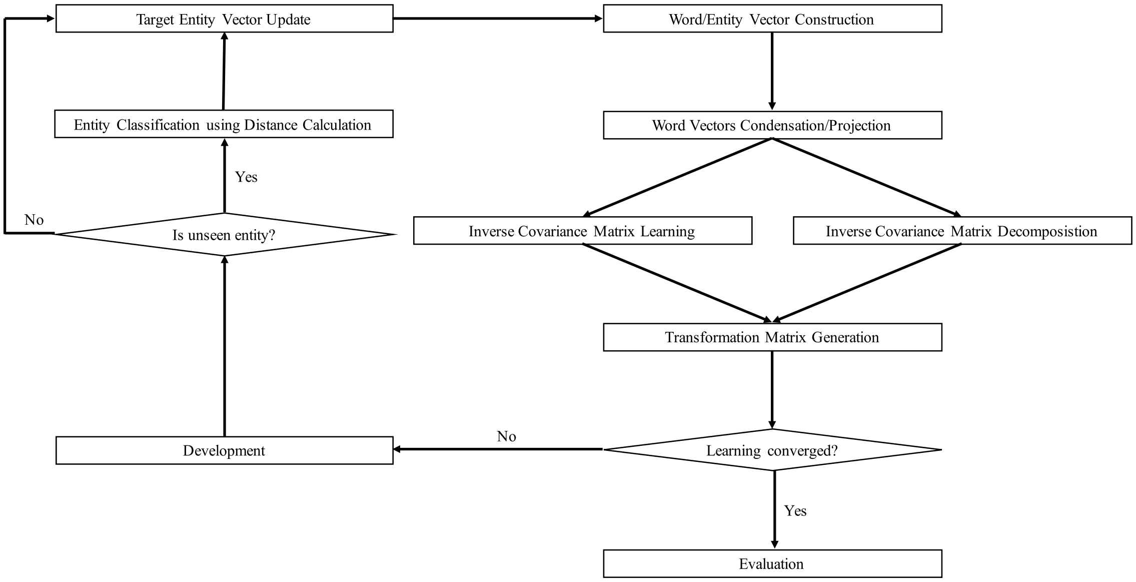

Target Entity Vector Update
Word/Entity Vector Construction
Entity Classification using Distance Calculation
Word Vectors Condensation/Projection
Yes
No
Is unseen entity?
Inverse Covariance Matrix Learning
Inverse Covariance Matrix Decomposistion
Transformation Matrix Generation
No
Development
Learning converged?
Yes
Evaluation

Figure 2: Overview of Methodology Flow

Let $w_0 ... w_n$ represent the individual word vectors and $p_0 ... p_n$ represents the phrase vectors which have the concatenated word form of all words in the phrases. D is a set of pre-identified entities of $d_0 ... d_n$ (gold entity-vectors) in DBpedia ontology. We denote E to be a set of entity vectors of $e_0 ... e_n$ where $E = D \cap p$. The elements $e \in E$ are obtained from either from initial entity vector generation or later learning of unseen entities. $proj(\{w_i, ..., w_j\})$ is our projection function which takes in a sequence of word vector and output a semantically condense vector with the same dimension of the input word vectors.We then seek to find a transformation matrix Σ^{-1} where

$$proj(\{w_i, ..., w_j\}) \cdot \Sigma^{-1} = d$$

For all entity vectors in DBpedia $d_0 ... d_n$ we compute their transformation matrices $\Sigma_0^{-1} .... \Sigma_n^{-1}$ and cluster them by each entity label. We can now compute a principle vector c to represent each group. Then for all entity vectors $e_0 ... e_n$ where we find the closest entity vector c and learn from it.

3.1.1 Pre-trained Entity Vector Construction

Our words-to-entity transformation matrix requires pre-train sets of word vectors and phrase entity vectors. We concatenate three different corpus, New York Times, Wall Street Journal, and Wikipedia, into one raw corpus. Word vectors are then trained using the word2vec skip-gram model with negative sampling and continuous bag-of-words (Mikolov et al., 2013b). Two sets of phrased entity vectors are constructed here. One will use the built in phrase vector generated by word2vec, and the other one with chunker we built that extracts both proper noun and named entity chunks based on the information given by the ClearNLP [1].

The initial set of gold entity vectors will be constructed from grouping the pre-trained phrase vectors that exist in the DBpedia ontology. If more than one entity vectors are found in both sets of phrase vectors, the average vector of the target entity vectors will be added to the entity vector set. The algorithm is described as below.

Algorithm 1 Generate initial entity vectors

```
Require: $pW$ (Set contains concatenated word form of $p_0, ..., p_n$)
Ensure: $|e| = |D \cap pW|$
  for $\forall d$ in $D$ do
    $count \leftarrow |d \cap pW|$
    if $c > 0$ then
      $v \leftarrow \frac{1}{c} \sum p_i ||$ concatenated word form of $p_i \in pW$ add $(v)$ to $e$
    end if
  end for
```

Each $e_0 ... e_n$ of identified entity vector are further categorized into groups with its corresponding label dictated by the DBpedia ontology. Principle entity vectors or the centroids of each groupings are then calculated from taking the average of all entity vector within the same group.

3.1.2 Word Vector Projection and Condensation

In order to address the potential input as sequences of word vectors, we will be experimenting with different projection method that would condense the semantics of each individual word representations. To ensure the dimension of the input vector with be consistence to the dimension of the transformation matrix we will attempt the following:

- Average Word Vector
- Stop Word Elimination
- Substitution of the entity vector segments

[1] https://github.com/clir/clearnlp

Word2vec introduce a projection layer to their neural network for the continuous bag of words model. The layer calculates the projected vector from taking the average of all word vectors within in a specified window (Mikolov et al., 2013a). The following equation describes the approach.

$$p(\{w_i, ..., w_j\}) = \frac{1}{j-i} \sum_{x=i}^{j} w_x$$

Figure 3: Projection function averaging word vectors

Stop word elimination would potentially help reduce the potential noise in distributional semantics. Stop words are a set of word tokens that have high frequency of occurrence within the input corpus. Due to their frequent count in contextual distribution, they usually do not add to the semantics of a word sequence. We will generate a list of stop word candidate by introducing a threshold hyper-parameter for occurrence-count of each word token. While projecting the a sequence word vectors, the vectors that represent a stop word will be ignored and thus not accounted for project.

This project would also try to utilize existing entity vector to provide additional semantic information at the entity level. It is doubtful that the average vector projection method above would produce a compact vector representation for the word sequence. It seems reasonable that syntactically phrases are composed of individual words, but they might not directly correlated to the accumulation of their contextual distribution information. Therefore, we propose to include segmented entity vector information to replace parts of the word sequence that can be recognized as entities. The following algorithm describes such approach.

Algorithm 2 Replace segments of potential entities in a sequence of word vectors

```
Require: $w_i, ..., w_j$ (Sequence of consecutive words in their vector form)
Require: $WordForm(x)$ returns concatenated word form of the input set of vectors
Ensure: $v = proj(\{w_i, ..., w_j\}) \neq null$
  $c \leftarrow i$
  $s \leftarrow \{\emptyset\}$
  for $i = i;\ i < j;\ i = i + 1$ do
    if $WordForm(\{x_c, ..., x_i\}) \in \{WordForm(e_0), ..., WordForm(e_n)\}$ then
      add $e_k$ to $s$ where $WordForm(\{x_c, ..., x_i\}) = WordForm(\{e_k\})$
      $c \leftarrow i + 1$
    else
      add $w_i$ to $s$
    end if
  end for
  $v \leftarrow \frac{1}{|s|} \sum_{x=1}^{|s|} s_x$
  return $v$
```

3.1.3 Words-to-Entity Transformation Matrix Decomposition

Matrix decomposition is a well researched area and many of its methodology is publicly available. Due to the sparseness and high dimension nature the transformation matrix will have to be estimated. We will use and evaluate the following factorization methods:

- Trivial Singular Value Decomposition
- Implicit Factorization using PPMI
- Quadratic Approximation (Hsieh et al., 2011)
- Linear Programming (Yuan, 2010)

Existing implementation of the inverse matrix approximation approaches will be modified and used in this project to construct $\Sigma_0^{-1}....\Sigma_n^{-1}$ transformation matrices. This step will serves as a prior analysis on the feasibility of the concept of words-to-entity vector transformation. The performance algebraically obtained inverse matrices will later be compared with the learbned matrices.

3.1.4 Semi-supervised Entity Vector Update

A cluster analysis to validate the clusters of transformation matrices will be taken before the semi-supervised learning. In addition the calculation of the centroid of each cluster will be explored since taking the average might result in loss of information. After this analysis similarity measures are calculated between rest of the phrase vectors $e_0...e_n$ and the centroids $c_0...c_n$. Each phrase vector is added to the group who's centroid is closest. Then the centroids are recalculated and this leads to minimal supervision for learning. The similarity measures and model updates requires further exploration and raises the question of having finer-grained entity types as opposed to general entity types.

3.1.5 Words-to-Entity Transformation Matrix Learning

Instead of estimating the transformation matrices by factorization we suggest the use of a feed forward neural net to learn each matrix. The architecture of this net consists of an input layer, projection layer, a single hidden layer, and an output layer. We input $w_0...w_n$ into our input layer. The projection function $p(w_i,...,w_j)$ is applied in the projection layer. Initial random weights are assigned from the projection layer to the hidden layer and the hidden layer to the output layer. Our output layer will be fixed and will hold d that corresponds to input w. The net will then learn the hidden layer which is our transformation matrix Σ^{-1} through back propagation. The input layer can remain fixed throughout training or can be relearned to produce better word vectors. The exploration and use of these new word vectors is not the scope of the project but will be explored separately.

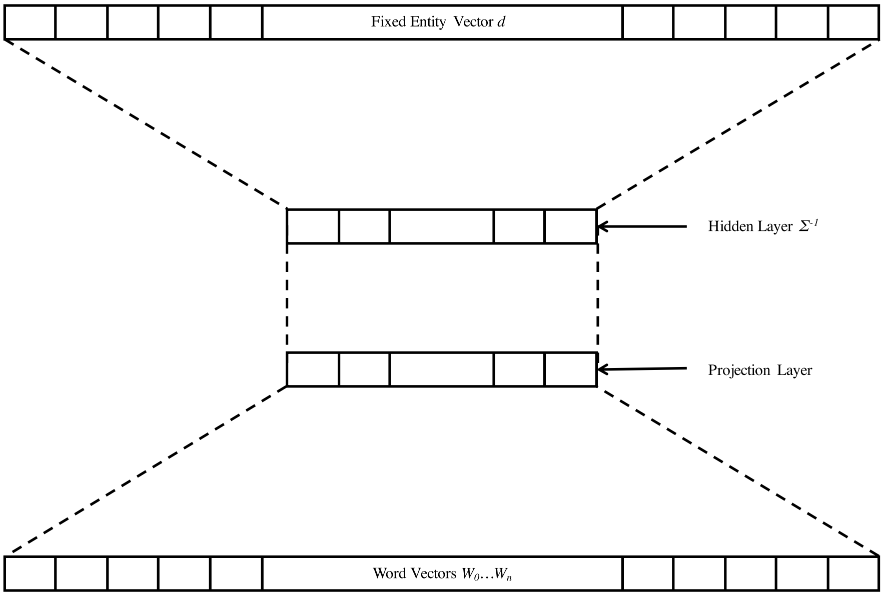


Figure 4: Matrix Learning Architecture

3.2 Evaluation

3.2.1 Accuracy Evaluation

To evaluate the accuracy of our words-to-entity vector transformation, we will use the output of the mapping for the Named Entity Recognition (NER) task. The output will be tested on three different data sets:

- CoNLL '03 testing data for NER
- ACE 2003 and 2007 data
- MUC 7 Formal Run

The F1 harmonic score between precision and recall will be used to evaluate the accuracy. The equations are shown below.

$$Precision = \frac{gold \cap prediction}{prediction}, \quad Recall = \frac{gold \cap prediction}{gold}, \quad F1Score = 2 \times \frac{precision \times recall}{precision + recall}$$

Figure 5: Equations for Named Entity Prediction Evaluation

In terms of the labels of the output entity vectors, since most of the initial entity vectors are labeled based on the ontology of DBpedia, entities already seen or in DBpedia will not have different labels. The labeling of learned unseen entity will be determined by calculating the closest distance between the output entity vector and the centroids of each entity groupings.

3.2.2 Entity Vector Quality Evaluation

To evaluate the quality of the resultant entity vectors, several analogy tests will be perform on the output vectors to determine the embedding quality. The analogy tests focus on the analogical reasoning between pairs of words or concepts. There are two main categories for the test, and they are syntactic and semantic questions. Syntactic questions typically test on analogies of verb tenses and forms of adjectives, and semantic question mainly examine analogies of relationships between entities.

Run is to running as walk to _____? (Sample syntactic question)
Washington DC is to United States as Taipei to _____? (Sample semantic question)

Figure 6: Examples for Analogy Tests

Since this project mainly concerns on the similarity between entities (or sequences of words), all syntactic questions and the semantic questions that involve non-entity words or phrases will be ignored.

The analogy task in Mikolov et al. (2013a) will be used to evaluate the quality of the output entity vectors. Additional the following word similarity tasks will also be used:

- WordSim-353 (Finkelstein et al., 2001)
- SCWS (Huang et al., 2012)
- RW (Luong et al., 2013)

4 Timeline

The following table is a proposed bi-weekly timeline for this project. It separates and assigns foreseeable tasks to each member of this project. The completion dates mark the desired dates for concluding the assigned tasks.

Completion Date	Henry Chen	Johnny Tan
10/18/2015	Related work research Project proposal drafting	Related work research Project proposal drafting
11/01/2015	Update existing chunkers DBpedia-based entity grouping Projection method exploration	Initial entity vector generation Word2Vec phrase vector construction Projection method exploration
11/15/2015	Component structure implementation Transformation matrix decomposition via inverse matrix estimation	Neural network adaptation Transformation matrix decomposition via neural network
11/29/2015	Unseen entity addition analysis Evaluation data collection Project paper/presentation drafting	Cluster Analysis Evaluation data collection Project paper/presentation drafting
12/07/2015	Documentation and demo creation Project paper/presentation revision	Documentation and demo creation Project paper/presentation revision
12/21/2015	Project integration with NER system NACCL 2016 short paper drafting	Project integration with NER system NACCL 2016 short paper drafting
01/04/2015	NACCL 2016 short paper revision	
01/06/2015	NACCL 2016 short paper submission	

Table 1: Projected Bi-Weekly Project Timeline

References

Manaal Faruqui and Chris Dyer. Improving vector space word representations using multilingual correlation. Association for Computational Linguistics, 2014.

Lev Finkelstein, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman, and Eytan Ruppin. Placing search in context: The concept revisited. In *Proceedings of the 10th international conference on World Wide Web*, pages 406–414. ACM, 2001.

Cho-Jui Hsieh, Inderjit S Dhillon, Pradeep K Ravikumar, and Mátyás A Sustik. Sparse inverse covariance matrix estimation using quadratic approximation. In *Advances in Neural Information Processing Systems*, pages 2330–2338, 2011.

Eric H Huang, Richard Socher, Christopher D Manning, and Andrew Y Ng. Improving word representations via global context and multiple word prototypes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 873–882. Association for Computational Linguistics, 2012.

Omer Levy and Yoav Goldberg. Neural word embedding as implicit matrix factorization. In Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2177–2185. Curran Associates, Inc., 2014. URL `http://papers.nips.cc/paper/5477-neural-word-embedding-as-implicit-matrix-factorization.pdf`.

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

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013b.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word Representations: A Simple and General Method for Semi-supervised Learning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, ACL'10, pages 384–394, 2010.

Peter D Turney, Patrick Pantel, et al. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research*, 37(1):141–188, 2010.

Ming Yuan. High dimensional inverse covariance matrix estimation via linear programming. *The Journal of Machine Learning Research*, 11:2261–2286, 2010.