STRUCTURE AND MEANING IN WORD EMBEDDING SPACES

by

Tucker Berckmann

Department of Computer Science
Brown University

Date:

23 April 2021

Approved:

Prof. Ellie Pavlick, Supervisor

A project submitted in partial fulfillment of the requirements for the degree of Master of Science in the Department of Computer Science in the Graduate School of Brown University

2021
Abstract

Interpreting word embeddings that have been generated by gradient descent is a difficult problem. We explore several methods to discover or create interpretable factors within a word embedding matrix. Finally, we propose a novel neural model that uses a language model to describe the contents of an embedding space, building on the concept of definition modeling. We show that a neural method can isolate connotation in embedding spaces without loss of information, and we demonstrate a new method to generate an English-language summary of arbitrary embedding vectors.
Acknowledgements

We thank Ellie Pavlick for her supervision of this work, Albert Webson for his guidance on connotation discovery, and Tali Bers for her contributions to the BLEU evaluation.
Contents

Abstract ii

Acknowledgements iii

List of Figures vi

List of Tables vii

1 Introduction 1

1.1 Background ......................................................... 1

1.2 Related Work ....................................................... 1

1.3 Motivation .......................................................... 2

2 Embedding Structure 3

2.1 Method .............................................................. 3

2.2 Evaluation ........................................................... 4

2.2.1 Quantitative Measures .......................................... 4

2.2.2 Qualitative Measure .............................................. 5

2.3 Approaches .......................................................... 6

2.3.1 Standard Decompositions ...................................... 6

2.3.2 Neural Methods ................................................... 8

2.4 Combined Results .................................................. 11

3 Embedding Meaning 12

3.1 Method .............................................................. 12

3.2 Evaluation ........................................................... 13

3.2.1 Quantitative Measure .......................................... 13
List of Figures

2.1 Overview of the transform $T$ .............................................. 3
2.2 Identification of party components ........................................ 4
2.3 Spearman’s $\rho$ with labels .................................................. 5
2.4 Threshold for predictions ..................................................... 5
2.5 Toy example of column vector sorting ..................................... 6
2.6 Neural classifier ................................................................. 9
2.7 Ultradense experiment .......................................................... 10
3.1 Word with definition ............................................................ 12
3.2 Definition modeling ............................................................ 13
3.3 Base-LSTM architecture ....................................................... 15
3.4 Definition-LSTM architecture ................................................ 15
3.5 Hybrid architecture ............................................................. 16
## List of Tables

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Qualitative results: PCA</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>Qualitative results: NNMF</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>Qualitative results: densifier</td>
<td>11</td>
</tr>
<tr>
<td>2.4</td>
<td>Experimental results: chapter 2 (embedding structure)</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>Rubric for human evaluation</td>
<td>14</td>
</tr>
<tr>
<td>3.2</td>
<td>Experimental results: chapter 3 (embedding meaning)</td>
<td>17</td>
</tr>
<tr>
<td>3.3</td>
<td>Selected examples of generated definitions from each model</td>
<td>17</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Background

Word embeddings in the context of deep learning models are learned vectors which represent words. This makes them parameters of a model, which are learned via gradient descent from an objective function. Interpreting these parameters is hampered by the same challenge that arises in all deep learning models: it is difficult to assign meaning and structure to large sets of parameters learned by gradient descent.

Specifically, we build on the concepts of 1) style transfer, which generally involves the isolation of specific common attributes in a set of features, and 2) definition modeling, the generation of dictionary definition text through language modeling, given a dictionary headword.

This work is divided into two main parts: 1) interpretable embedding factors and 2) definition modeling. In the first part, chapter 2, we analyze the structure of a pre-trained word embedding space in the context of a binary classification task. In this analysis, we compare several methods to find the optimal strategy for isolating the binary attribute to a minimal set of dimensions (factors). In the second part, chapter 3, we adapt the concept of definition modeling to a pre-trained embedding space in order to extract meaning from vectors within the space.

1.2 Related Work

Finding (or creating) structure in word embeddings falls loosely under the category of style transfer. In their work on style transfer, Shen et. al. [16] projected sentences onto a shared content space, separately isolating the style. One or more components of interest may be isolated; for example Liao et. al. [8] discover embedding factors that correspond to a set of predefined categories. In their work on dictionary learning, Zhang et. al. [20] isolate word factors using dictionary learning after applying sparse coding to the embedding matrix. Similarly, Subramanian et. al. [17] showed a method for transforming the embedding matrix into sparse embeddings using an autoencoder. Without using sparse embeddings, John et. al. [5] propose an effective approach for disentangling style and content latent spaces; here, they explicitly partition the embedding space using an adversarial network. Finally, in a related approach, Webson et. al. [19] partition the embedding space, also using an adversarial network, based on the attribute of word connotation.
As introduced by Noraset et. al. [11] definition modeling is implemented as a language model, which predicts each word of the definition given the headword being defined as well as the previous words in the definition. In their work on unknown phrases, Ishiwatari et. al. [4] extend the method beyond individual words to entire phrases. To improve modeling, Washio et. al. [18] take into account lexical relations between the headword and its definition. In their sequence-to-sequence approach, Mickus et. al. [10] use the contextual embedding of a headword in the context of a sentence in order to generate the word’s definition, as do Reid et. al. [14] in their variational modeler.

1.3 Motivation

In chapter 2, we analyze a set of pre-trained word embeddings based on a simple criterion: how much party connotation is contained in the embedding, as introduced by Webson et. al. [19] Here, party refers to a United States political party, either the Republican Party or the Democratic Party. For example, the term “illegal aliens” may contain a political anti-immigration connotation favored by the Republican Party, whereas the term (with equivalent meaning) “undocumented immigrant” may contain a connotation favored by the Democratic Party. Thus we treat connotation as a binary attribute corresponding to one of the two parties. Conceptually, this classification is similar to sentiment analysis, though we consider individual words instead of complete sentences. In the example above, we treat the compound word “illegal alien” as a single word with a single embedding.

The goal of our work in chapter 2 is to isolate a single factor (dimension) of the embedding space that contains the party connotation. In particular, we seek change-of-basis transformations to avoid information loss. Ideally, the transformation will be a simple matrix decomposition (which does not require a training set). Finally, we would like to ensure that the factors outside of the isolated dimension do not contain any information about the label.

In chapter 3, we introduce several neural models, including one novel model, that perform the definition modeling task. Our goal is to generate an English-language definition entirely from a single embedding vector from an arbitrary embedding space.
Chapter 2

Embedding Structure

2.1 Method

We begin with a set of word embeddings generated using the word2vec method from the Partisan News [6] corpus. We treat the embeddings as an embedding matrix $E_O$ with dimensions $v \times e_O$, where $v$ is the number of words and $e_O$ is the embedding size of each word. We seek a transform $T$ which maps each embedding in $E_O$ onto a new embedding, possibly with a different length $e_N$, in a new embedding matrix $E_N$. The relationship of $E_O$ to $E_N$ is shown in Figure 2.1.

![Figure 2.1: Overview of the transform T](image)

In the new embedding matrix $E_N$, we would like to identify a single factor $C$ (which is a single column vector) that contains the party connotation for each word. Also, we would like to remove information about the connotation from the remaining factors $R$. In addition to $E_N$, we have a vector of binary labels $L$ that describe the party connotation of each word. Note that each entry in $L$ corresponds to a row (representing a word embedding) in $E_N$. Interpreting the word embeddings will thus consist of identifying information in $C$ that predicts the corresponding label. See Figure 2.2 for an overview of $C$ and $R$. 


2.2 Evaluation

2.2.1 Quantitative Measures

Spearman’s Rho

Once we have generated the new embedding matrix $E_N$, we do not know which column vectors correspond to $C$ above. To determine this, we calculate the Spearman rank correlation coefficient (Spearman’s $\rho$) between each of the $e_N$ column vectors and the set of labels, as shown in Figure 2.3.

A Spearman’s $\rho$ that approaches 1.0 or -1.0 indicates a high correlation with the label. We then rank the column vectors in order of their Spearman’s $\rho$ and then examine the highest- and lowest-ranked: these vectors show the most correlation with the label and thus are the best choices for $C$ above.

Prediction Accuracy

Another measure of how much information about the label is contained in $C$ is the ability to predict the label of an unfamiliar word embedding based on the contents of the column vector. To do this, we treat the column vector as having two sample distributions: one for each label. Each of these distributions has a mean, $\mu_0$ and $\mu_1$. We establish a threshold value

$$
\tau_p = \mu_0 + \frac{\mu_1 - \mu_0}{2}
$$

(2.1)
and use $\tau_p$ to predict the unknown label: if its coefficient is below $\tau_p$, we assume that it belongs to distribution 0, and if the coefficient lies above $\tau_p$, we assume that it belongs to distribution 1. An overview is shown in Figure 2.4.

2.2.2 Qualitative Measure

The quantitative measure above treats each column vector as a single factor (dimension), where each word’s position along that dimension expresses its distance from other words and thus its relationship to them. In like manner, to determine the qualitative relationships between words expressed by the column vector, we sort the
words by that column to see how they group together. A toy example (not related to the corpora under consideration) is shown in Figure 2.5. Note that after sorting, the similar words “cat” and “mice” are positioned close to each other.

**Figure 2.5**: Toy example of column vector sorting

### 2.3 Approaches

In the following sections, we compare several methods to generate the desired transformation $T$.

- **Standard Decompositions**
  - Principal Component Analysis
  - Non-negative Matrix Factorization

- **Neural Methods**
  - Neural Classifier
  - Ultradense

#### 2.3.1 Standard Decompositions

**Principal Component Analysis**

**Overview** Principal Component Analysis (PCA) is the process of determining the principal components of a set of vectors and then changing the basis of the vectors to the principal components. The principal components are best-fit vectors which are all orthogonal to each other.

Normally, PCA is calculated using the singular value decomposition (SVD), so

$$E_O = U \Sigma W^T.$$  \hfill (2.2)
Performing the PCA is equivalent to multiplying by the matrix $W$, so

$$E_N = E_O W$$  \hspace{1cm} (2.3)

where $W$ is equivalent to the transform $T$ described above.

**Hypothesis of Experiment** We hypothesize that one or more of the principal components represent the political party most associated with each word, in other words, that they will have a high Spearman’s $\rho$ against the labels.

**Experiment Description** We perform SVD on $E_O$ to get $E_N$ and then find the column vector in $E_N$ with the highest Spearman’s $\rho$.

**Experiment Results** The column vector with the highest Spearman’s $\rho$ was the second principal component, which suggests that party connotation has high variance in the embedding space. However, $\rho$ was lower in this case than for raw embeddings, which means that $E_N$ is less useful than $E_O$ for making predictions. Thus the hypothesis was not supported. The top and bottom five words in the component with the highest $\rho$ are shown in Table 2.1. See Table 2.4 for a comparison between this method and the raw embeddings.

### Table 2.1: Qualitative results: PCA

<table>
<thead>
<tr>
<th>Top (R)</th>
<th>Coeff.</th>
<th>Bottom (D)</th>
<th>Coeff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>capital_flows</td>
<td>+2.04</td>
<td>msnbc</td>
<td>-1.46</td>
</tr>
<tr>
<td>account_surplus</td>
<td>+1.86</td>
<td>o’reilly</td>
<td>-1.48</td>
</tr>
<tr>
<td>emerging_markets</td>
<td>+1.84</td>
<td>sean_hannity</td>
<td>-1.53</td>
</tr>
<tr>
<td>inflows</td>
<td>+1.80</td>
<td>hannity</td>
<td>-1.60</td>
</tr>
<tr>
<td>outflows</td>
<td>+1.80</td>
<td>daily_show</td>
<td>-1.61</td>
</tr>
</tbody>
</table>

**Non-negative Matrix Factorization**

Standard non-negative matrix factorization (NNMF) seeks to decompose a matrix into the product of two other matrices, where all three matrices contain only non-negative elements. One of these matrices is $E_N$, the new embedding matrix we seek, and the other matrix is $D$, the dictionary, which we discard after calculating $E_N$. So

$$E_O^T = DE_N^T.$$  \hspace{1cm} (2.4)

Normally, an iterative method, such as from Lee and Seung [7] is used to solve Equation 2.4. Thus instead of learning $T$, as described above, we calculate $E_N$ indirectly. The utility of this factorization in the context of word embeddings is this:
since the matrix factors are non-negative, each word in the vocabulary is represented as an additive combination of some set of attributes. However, standard NNMF requires that all three matrices be nonnegative, and $E_0$ is not, in general, nonnegative. As demonstrated by Faruqui et. al. [3] using an approach specifically designed for word embeddings, which have negative values, we may learn $E_N$ by solving

$$\text{arg min}_{D,E_N} ||E_O^T - DE_N^T||_2^2 + \lambda \Omega(E_N) + \tau_r ||D||_2^2$$

using adaptive gradient descent, where $\lambda$ is a regularization hyperparameter, $\Omega$ is the regularizer, and $\tau_r$ is the regularization penalty.

**Hypothesis of Experiment**  By setting $\epsilon_N$ to a large value (ten times $\epsilon_O$), we decompose each word into a large number of factors. Since we expect that NNMF will discover all or most of the relevant factors, we hypothesize that one of these factors corresponds to the word’s connotation.

**Experiment Description**  We perform NNMF on $E_O$ to calculate $E_N$ and then find the column vector in $E_N$ with the highest Spearman’s $\rho$, as in PCA.

**Experiment Results**  As in PCA, the highest $\rho$ was lower in this case than for raw embeddings. This implies that there was no single column vector (i.e. factor) that corresponds to the party label. Therefore, the hypothesis was not supported. The qualitative results are shown in Table 2.2.

### 2.3.2 Neural Methods

**Neural Classifier**

**Overview**  A neural classifier is a neural network that makes predictions based on a set of training examples. In this work, the training examples are the word embeddings and the prediction is binary: party connotation. Unlike the examples above, when using a neural classifier, there is no transformation $T$ and no new embedding.
matrix $E_N$; the classifier predicts labels directly from $E_N$. Consequently, there is no Spearman’s $\rho$ or word ordering measurement for the neural classifier.

**Hypothesis of Experiment** We assume that the classifier has maximal ability, compared with all other methods, to determine the label based on $E_N$ and consequently represents an upper bound on the accuracy for all experiments.

**Experiment Description** We train a standard neural classifier on the embedding matrix $E_O$ to predict the connotation label, as shown in Figure 2.6.

![Figure 2.6: Neural classifier](image)

**Experiment Results** The results support the hypothesis; the neural classifier had the highest accuracy. Thus it serves as a basis for comparison for the other methods. See Table 2.4 for a comparison between the neural classifier and the other methods.

**Ultradense**

**Overview**

As described by Rothe et. al. [15] the densifier is a method that learns $T$ above in order to focus “the information relevant for a task in an *ultradense subspace*.” In our case, the task is distinguishing the label, and the ultradense subspace is a single column of $E_N$.

Since the densifier method’s restriction that $T$ be an orthogonal matrix is not, in general, enforced by gradient descent, the densifier method finds the nearest orthogonal matrix at every update step.
In the densifier method, the parameter matrix of the ultradense model $T_\Phi$, which is learned by gradient descent, is an approximation to $T$. In the same manner as PCA, we may decompose $T_\Phi$ using SVD:

$$T_\Phi = U\Sigma W^T.$$  \hspace{1cm} (2.6)

Now according to Fan and Hoffman [2], the closest orthogonal matrix to $T_\Phi$ is:

$$T = UW^T.$$ \hspace{1cm} (2.7)

We substitute $T$ for $T_\Phi$ at each gradient update step to maintain both the orthogonality of $T$ as well as the progress of the model updates.

**Hypothesis of Experiment**  When employing the densifier, we choose one column vector in $E_N$ to focus the ultradense coefficients. We hypothesize that this column vector will have a high Spearman’s $\rho$ in relation to its party labels. We further hypothesize that the remaining embedding space outside of target column vector will not contain party connotation information, since this information has been focused into the ultradense subspace.

**Experiment Steps**  We train a densifier model (which learns $T$) to generate $E_N$. We measure Spearman’s $\rho$ and prediction accuracy on its ultradense subspace. We then train a standard neural classifier on the remaining components to predict the connotation label. The experiment components are shown in Figure 2.7.

![Figure 2.7: Ultradense experiment](image-url)
<table>
<thead>
<tr>
<th>Top (R) Coeff.</th>
<th>Bottom (D) Coeff.</th>
</tr>
</thead>
<tbody>
<tr>
<td>daily_show +1.34</td>
<td>regional_organization -1.79</td>
</tr>
<tr>
<td>dad +1.26</td>
<td>africa_politics -1.82</td>
</tr>
<tr>
<td>donate +1.23</td>
<td>international_organizations -1.87</td>
</tr>
<tr>
<td>tmv +1.20</td>
<td>africa_nigeria -1.90</td>
</tr>
<tr>
<td>kid +1.19</td>
<td>counterterrorism_war -1.93</td>
</tr>
</tbody>
</table>

Table 2.3: Qualitative results: densifier

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy %</th>
<th>Spearman’s ρ</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neural Classifier</td>
<td>76.4</td>
<td>-</td>
<td>2.3.2</td>
</tr>
<tr>
<td>Classifier on Complement</td>
<td>76.3</td>
<td>-</td>
<td>2.3.2</td>
</tr>
<tr>
<td>Ultradense Classifier</td>
<td>74.3</td>
<td>0.52</td>
<td>2.3.2</td>
</tr>
<tr>
<td>Raw Embeddings</td>
<td>68.8</td>
<td>0.41</td>
<td>-</td>
</tr>
<tr>
<td>Principal Component Analysis</td>
<td>66.7</td>
<td>0.37</td>
<td>2.3.1</td>
</tr>
<tr>
<td>Nonnegative Matrix Factorization</td>
<td>56.3</td>
<td>0.21</td>
<td>2.3.1</td>
</tr>
</tbody>
</table>

Table 2.4: Experimental results: chapter 2 (embedding structure)

**Experiment Results** The densifier model was able to produce an ultradense subspace that had a high correlation with the labels, as shown in Table 2.4. Thus the first part of the hypothesis was supported by the experimental results. However, a classifier was able to predict the labels with high accuracy from components \( R \); this means that \( R \) duplicated the information from \( C \), so the second part of the hypothesis was not supported by the experiment. The qualitative output of the experiment is shown in Table 2.3.

### 2.4 Combined Results

In chapter 2, we experimentally test several methods for isolating the desired dimension. The results are shown in Table 2.4. The best performing methods have the highest accuracy, although as described in the goals section, “the classifier on complement” ideally will have an accuracy of 50%, which corresponds to random choice. The neural classifier is a reference, which we take as an upper bound on the accuracy.

The results in Table 2.4 show that we succeeded in isolating one factor (one dimension) of the embedding space that contains the party connotation (ultradense classifier), but we failed to remove this information from the remaining factors (classifier on complement).
Chapter 3

Embedding Meaning

3.1 Method

Definition modeling is, in general, using a language model to predict each word of a dictionary definition given the headword being defined as well as the previous words in the definition. This is accomplished by the model on headword, definition pairs from a standard English dictionary. One example of this language modeling task is described mathematically as

$$p(\vec{d}|w^*) = \prod_{t=1}^{T} p(w_t|w^*, w_2, ..., w_{t-1}).$$  \hfill (3.1)

In Equation 3.1, the probability of the definition $\vec{d}$ given the headword $w^*$ is the product of the probability of each word given the previous definition words. For example, given the headword ($w^*$) “speak,” correct output from the model would be the definition ($\vec{d}$) “To utter words.” In Figure 3.1, we show the headword and its definition.

![Figure 3.1: Word with definition](image)

We propose using an arbitrary vector in a static embedding space (i.e. one not learned by the model) to replace $w^*$ as a basis for definition generation. We will let $\mathbb{Z}$ be the set of nonnegative integers and $\mathbb{R}$ be the set of real numbers. Although $w_t \in \mathbb{Z}$ is in general a discrete word (or subword) index in a numbered vocabulary, we generalize the headword (but not the other words in the definition) as an embedding itself, $\vec{e}$, with length $e_O$. So $\vec{e}$ is a row in the original embedding matrix $E_O$ and $\vec{d}$ is a row in the definition matrix $D$. Instead of the matrix $T$ as described in section 2.1, we implement a neural language model—the learned function $T_\Phi: \mathbb{R}^{v \times e_O} \rightarrow \mathbb{Z}^{v \times d_N}$—so

$$D = T_\Phi(E_O).$$  \hfill (3.2)
In Figure 3.2, we show how embeddings are transformed into the corresponding definitions according to Eqn. 3.2. Note that $d_N$ is the output window size, which is the largest allowable definition length.

We define $E_O$ to be a subset of the pre-trained Glove [13] embedding matrix. Similarly to the approach by Noraset et. al. [11] we begin with the GNU Collaborative International Dictionary of English (GCIDE) and select the 20,000 most frequent words from the trillion-word corpus. [1]

Thus $v = 2 \cdot 10^4$. Departing from their method, we choose a single definition for words which have multiple definitions.

3.2 Evaluation

3.2.1 Quantitative Measure

To evaluate our results automatically, we use BLEU as described by Papineni et. al. [12] Unlike the paper, we take $N = 3$ and $w = \{\frac{2}{3}, \frac{2}{3}, \frac{1}{3}\}$. Given an input word or embedding, the model generates a prediction in inference mode using beam search with a width of five. We evaluate the BLEU score on this output against the set of definitions for this headword in GCIDE as references.

3.2.2 Qualitative Measure

We employ a custom grading scheme for human evaluation, using a numerical score from 0 to 6 based on the rubric as described in Table 3.1.
<table>
<thead>
<tr>
<th>Score</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Scholarly definition</td>
<td>purview: the limit or scope of a statute</td>
</tr>
<tr>
<td>5</td>
<td>Precise definition</td>
<td>jump: To leap</td>
</tr>
<tr>
<td>4</td>
<td>Correct definition, with flaws</td>
<td>broom: A long, slender piece of wood</td>
</tr>
<tr>
<td>3</td>
<td>Oversimplified / circular definition</td>
<td>break: To break into pieces</td>
</tr>
<tr>
<td>2</td>
<td>On topic, but not correct</td>
<td>hear: To utter words indistinctly</td>
</tr>
<tr>
<td>1</td>
<td>Proper English, but off topic</td>
<td>fed: A genus of plants</td>
</tr>
<tr>
<td>0</td>
<td>Not intelligible English</td>
<td>target: A persistent of traitor</td>
</tr>
</tbody>
</table>

Table 3.1: Rubric for human evaluation

3.3 Models

We consider three neural models that implement $T_d$, the first two of which are inspired by Noraset et. al., [11] and the third of which is a novel architecture:

- Base-LSTM
- Definition-LSTM
- Hybrid

Each of these models is trained as a language model, using the definition of the headword as the generated text. During inference, only the headword is supplied, and the model predicts the remainder of the definition using beam search.

3.3.1 Base-LSTM

The Base-LSTM model is a baseline that takes as input word indices instead of embeddings, as envisioned by Noraset et. al. [11] This model implements Equation 3.1 directly. In Figure 3.3, we show the model architecture of the Base-LSTM model. Here, the embedding matrix is a subset of Glove and is frozen during training. Also note that this model does not conform to Equation 3.2.

3.3.2 Definition-LSTM

The Definition-LSTM model injects $\tilde{e}$ into the LSTM at every time step. Thus the Definition-LSTM is described by

$$p(\tilde{d}|\tilde{e}) = \prod_{t=1}^{T} p(w_t|w_1, w_2, ..., w_{t-1}, \tilde{e}). \quad (3.3)$$

In Figure 3.4, we show the model architecture of the Definition-LSTM model.
Figure 3.3: Base-LSTM architecture

Figure 3.4: Definition-LSTM architecture
3.3.3 Hybrid

The Hybrid model uses a pre-trained GPT-2 model as a subcomponent to first generate contextual embeddings of the definition. Using a custom language modeling head, a transformer, we inject $\vec{e}$ at the first time step, replacing the headword. Thus the model is described by:

$$p(\vec{d}|\vec{e}) = \prod_{t=1}^{T} p(w_t|\vec{e}, w_2, ..., w_{t-1}).$$

(3.4)

As shown in Figure 3.5, the transformer learns to apply $\vec{e}$ appropriately to the subsequent time steps of the output through its attention mechanism. The final layers of GPT-2 are unfrozen and trained to project the contextual embeddings onto a shared space with $\vec{e}$.

![Figure 3.5: Hybrid architecture](image)

3.4 Experiment Description

We train all three models on the training set: 90% of the 20,000 most common definitions in the GCIDE data set, split at random. We use the remaining 10% as a test set for calculating the BLEU score. The human evaluation is performed on a set of thirty common words which are hand-selected and included in the test set, but not the training set.

As a reference, we fine-tune GPT-2 on the definitions from the training set as a language modeling task: the headwords are concatenated with their definitions. In inference mode, we perform beam search by providing GPT-2 with the headword only and allowing it to predict the most likely definition.
In chapter 3, we evaluate several original models on the definition modeling task. The evaluation metrics include a human evaluation, with a score from 0 to 6, as well as a BLEU [12] score against a set of dictionary entries. We use GPT-2, fine-tuned on the language-modeling task, as a reference.

As demonstrated by Table 3.2, our novel model “Hybrid,” achieved promising results on a definition modeling task. The BLEU score is not informative, since it does not distinguish between large differences in output quality. We consider the results of the human evaluation to be definitive. Since this model takes real-valued embedding vectors as its input, it successfully summarizes the content of arbitrary vectors in this embedding space.

While the Hybrid model produced simpler definitions than the reference, they were generally more creative, although this model’s performance was less consistent than the reference. Although the BLEU score demonstrated a difference between search methods, it did not accurately reflect the quality of the models. The LSTM-based models were likely too simple to capture the depth of the relationships required to generate good definitions. Example output for each model is illustrated in Table 3.3.

### Table 3.2: Experimental results: chapter 3 (embedding meaning)

<table>
<thead>
<tr>
<th>Method</th>
<th>Human Evaluation</th>
<th>BLEU</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\mu$</td>
<td>$\sigma$</td>
</tr>
<tr>
<td>GPT-2</td>
<td>3.21</td>
<td>0.52</td>
</tr>
<tr>
<td>Hybrid</td>
<td>2.35</td>
<td>1.11</td>
</tr>
<tr>
<td>Definition-LSTM</td>
<td>0.50</td>
<td>0.64</td>
</tr>
<tr>
<td>Base-LSTM</td>
<td>0.75</td>
<td>0.75</td>
</tr>
</tbody>
</table>

### Table 3.3: Selected examples of generated definitions from each model

<table>
<thead>
<tr>
<th>Model</th>
<th>truck</th>
<th>rack</th>
<th>boy</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPT-2</td>
<td>One who, or that which, truckes</td>
<td>A piece of wood, metal, or other material</td>
<td>One who, or that which, is boy</td>
</tr>
<tr>
<td>Hybrid</td>
<td>One who, or that which, loads</td>
<td>An instrument for measuring temperature</td>
<td>A young person</td>
</tr>
<tr>
<td>Definition-LSTM</td>
<td>A week of hurst genus States</td>
<td>To eclectic a passer of hurst</td>
<td>A genus of hurst genus</td>
</tr>
<tr>
<td>Base-LSTM</td>
<td>The justice or hearsay of meterology</td>
<td>The respecting of dictator</td>
<td>One who, or dispense which, defends</td>
</tr>
</tbody>
</table>

### 3.5 Experiment Results

In chapter 3, we evaluate several original models on the definition modeling task. The evaluation metrics include a human evaluation, with a score from 0 to 6, as well as a BLEU [12] score against a set of dictionary entries. We use GPT-2, fine-tuned on the language-modeling task, as a reference.

As demonstrated by Table 3.2, our novel model “Hybrid,” achieved promising results on a definition modeling task. The BLEU score is not informative, since it does not distinguish between large differences in output quality. We consider the results of the human evaluation to be definitive. Since this model takes real-valued embedding vectors as its input, it successfully summarizes the content of arbitrary vectors in this embedding space.

While the Hybrid model produced simpler definitions than the reference, they were generally more creative, although this model’s performance was less consistent than the reference. Although the BLEU score demonstrated a difference between search methods, it did not accurately reflect the quality of the models. The LSTM-based models were likely too simple to capture the depth of the relationships required to generate good definitions. Example output for each model is illustrated in Table 3.3.
Chapter 4

Conclusion

In chapter 2, our primary goals were:

- to isolate the connotation label in a single dimension of an embedding matrix, while preventing information loss, using a simple matrix decomposition;
- in the absence of a working simple decomposition, to accomplish the same objective using a supervised method; and
- to remove information about the connotation label in the remainder of the embedding matrix.

We did not achieve the first goal, and while did we not conclusively show that it cannot be achieved, finding a simple matrix decomposition did not appear promising, since none of the simple decompositions revealed a vector component that correlated with the label. We achieved the second goal, since the densifier method isolated a column vector that could predict almost the same accuracy in labels as the neural classifier, our hypothesized upper bound. We did not achieve the third goal, since the densifier method did not eliminate the party connotation from the remaining components of the embedding space. Whether this goal is achievable requires further investigation.

In chapter 3, our goal was to demonstrate a method to transform an arbitrary embedding space into a set of English-language descriptions. The Hybrid model approached the quality of our reference GPT-2 implementation and was able to produce correct definitions. Although the reference implementation scored higher on the human evaluation, many of these results were circular definitions, which while correct, are not rich definitions of the headword. Moreover, since the reference implementation was passed a word index as input, it had access to the word’s position in its own embedding matrix, which it could simply copy to the output. Our Hybrid model produced high quality definitions without a priori knowledge of the original embedding matrix. Thus, this type of model represents a promising direction for future research.
Chapter 5

Future Work

In part I, we might want to isolate factors of the embedding spaces for different label categories in general. It’s also possible that adding training objectives for additional factors would naturally lead to a reduction in information duplication. For this, the densifier method or a similar method may yield positive results. The work of Liao et. al. [8] also partitions the embedding space into multiple factors; although it uses the presence of an attribute as a training label instead of a regular classification label, it may still be adaptable to our goals.

Finally, it may be possible to influence the embedding space more effectively at training time, using a method similar to that of Luo et. al. [9]

In part II, we have generated definitions for an unknown embedding vector in a given embedding space. For our evaluation, this vector was a held-out word with a known definition. However, our method may be extended to other scenarios, such as to generate definitions for foreign words in a cross-lingual embedding space. It might also be used to describe relationships between words in a given embedding space using phrase generation.
Bibliography


[19] Albert Webson, Zhizhong Chen, Carsten Eickhoff, and Ellie Pavlick. Do “undocumented immigrants” == “illegal aliens”? differentiating denotation and con-