On Dimensional Linguistic Properties of the Word Embedding Space

Vikas Raunak*
Carnegie Mellon University
vraunak@cs.cmu.edu

Vaibhav Kumar*
Carnegie Mellon University
vaibhav2@cs.cmu.edu

Vivek Gupta
University of Utah
vgupta@cs.utah.edu

Florian Metze
Carnegie Mellon University
fmetze@cs.cmu.edu

Abstract

Word embeddings have become a staple of several natural language processing tasks, yet much remains to be understood about their properties. In this work, we analyze word embeddings in terms of their principal components and arrive at a number of novel and counterintuitive observations. In particular, we characterize the utility of variance explained by the principal components as a proxy for downstream performance. Furthermore, through syntactic probing of the principal embedding space, we show that the syntactic information captured by a principal component does not correlate with the amount of variance it explains. Consequently, we investigate the limitations of variance based embedding post-processing, used in a few algorithms such as (Mu and Viswanath, 2018; Raunak et al., 2019) and demonstrate that such post-processing is counter-productive in sentence classification and machine translation tasks. Finally, we offer a few precautionary guidelines on applying variance based embedding post-processing and explain why non-isotropic geometry might be integral to word embedding performance.

1 Introduction

Word embeddings have revolutionized natural language processing by representing words as dense real-valued vectors in a low dimensional space. Pre-trained word embeddings such as Glove (Pennington et al., 2014), word2vec (Mikolov et al., 2013) and fastText (Bojanowski et al., 2017), trained on large corpora are readily available for use in a variety of tasks. Subsequently, there has been emphasis on post-processing the embeddings to improve their performance on downstream tasks (Mu and Viswanath, 2018) or to induce linguistic properties (Mrkšic et al.; Faruqui et al., 2015). In particular, the Principal Component Analysis (PCA) based post-processing algorithm proposed by (Mu and Viswanath, 2018) has led to significant gains in word and sentence similarity tasks, and has also proved useful in dimensionality reduction (Raunak et al., 2019). Similarly, understanding the geometry of word embeddings is another area of active research (Mimno and Thompson, 2017). In contrast to previous work such as (Yin and Shen, 2018), which focuses on optimal dimensionality selection for word embeddings, we explore the dimensional properties of existing pre-trained word embeddings through their principal components. Specifically, our contributions are as follows:

1. We analyze the word embeddings in terms of their principal components and demonstrate that their performance on both word similarity and sentence classification tasks saturates well before the full dimensionality.

2. We demonstrate that the amount of variance captured by the principal components is a poor representative for the downstream performance of the embeddings constructed using the very same principal components.

3. We investigate the reasons behind the aforementioned result through syntactic information based dimensional linguistic probing tasks (Conneau et al., 2018) and demonstrate that the syntactic information captured by a principal component is independent of the amount of variance it explains.

4. We point out the limitations of variance based post-processing used in a few algorithms (Mu and Viswanath, 2018; Raunak et al., 2019) and demonstrate that it leads to a decrease in performance in sentence classification and machine translation tasks, restricting its efficacy mainly to semantic similarity tasks.

*equal contribution

Proceedings of the 5th Workshop on Representation Learning for NLP (RepL4NLP-2020), pages 156–165
July 9, 2020.

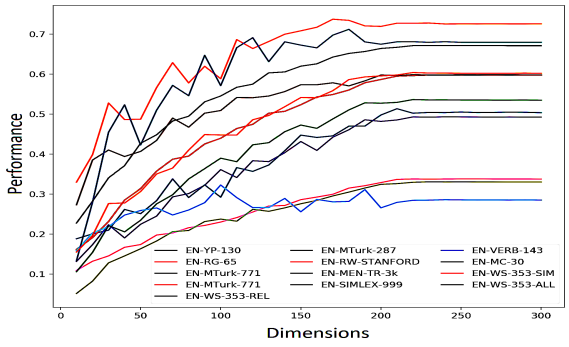

Figure 1: **Rho** x 100 on Word Similarity Tasks

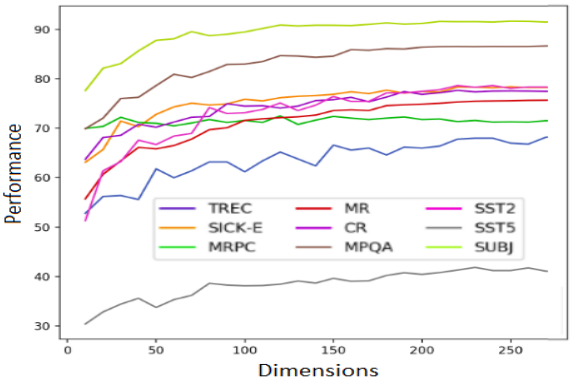

Figure 2: Accuracy on Sentence Classification Tasks

In Section 1, we provide an introduction to the problem statement. In Section 2, we discuss the dimensional properties of word embeddings. In Section 3, we conduct a variance based analysis by evaluating the word embeddings on several downstream tasks. In Section 4, we move on to dimensional linguistic probing tasks followed by Section 5, where we discuss variance based post-processing algorithms, and finally conclude in Section 7. To foster reproducibility, we have released the source code along with paper [1].

2 Dimensional Properties of the Word Embedding Space

Principal components provide a natural basis for studying the properties of an embedding space. In this work, we refer to the properties pertaining to the principal components of the embedding space as dimensional properties and the embedding space obtained by projecting the embeddings on the principal components as the principal embedding space. We study the principal embedding space and the dimensional properties in a number of different contexts such as word similarity, sentence classification. We provide a brief introductions to the both evaluation tasks of our experiment in the sub-sections. For details on the benchmarks, please refer to Conneau and Kiela (2018) for sentence classification and Faruqui and Dyer (2014) for word similarity.

For experiments in this section, we use 300 dimensional a) Glove embeddings (trained on Wikipedia 201 + Gigaword 5 [2]), b) fastText embeddings (trained on Wikipedia, UBMC webbase corpus and statmt.org news dataset [3]) and c) Word2vec embeddings (trained on the GoogleNews dataset [4]. We use Glove embeddings for the word similarity tasks. For the sentence classification tasks, we show results for fasttext and word2vec as well, in addition to Glove embeddings. For the sentence classification tasks we use Logistic Regression as the classifier, since it is the simplest classification model and we are only interested in evaluating performance variation due the changes in representations. Thus, the convex objective used in the classifier avoids any optimizer instability, making our entire evaluation pipeline deterministic and exactly reproducible.

2.1 Word Similarity Tasks

The word similarity benchmarks (Faruqui and Dyer, 2014) have word pairs (WP) that have been assigned similarity rating by humans. While evaluating word embeddings, the similarity between the words is calculated by the cosine similarity of their vector representations. Then, Spearman's rank correlation co-efficient (*Rho*) between the ranks produced using the cosine similarities and the given human rankings is used for the performance evaluation. Hence, for better word similarity, the evaluation metric (*Rho*) will be higher.

Figure 1 shows the performance (*Rho* x 100) of word embeddings (Glove) on 13 word similarity benchmarks w.r.t varying word embedding dimensions. The similarities are computed by projecting the embeddings in the principal component space. Each new evaluation cumulatively adds 10 more principal components to the earlier embeddings, i.e. the units on the X-axis vary in the increments of 10. Thus, we obtain 30 measurements for each dataset, ranging from word embeddings constructed using the first 10 principal components to orignal 300 principal components. From Figure

[1] https://github.com/vyraun/dlp

[2] https://stanford.io/2Gdv8uo

[3] https://bit.ly/2FMTB4N

[4] https://bit.ly/2esteWf

Table 1: Test accuracy of embeddings composed of **Top-100 (T)**, **Middle-100 (M)** and **Bottom-100 (B) principal components** on sentence classification datasets. The highlighted cells correspond to one of the three cases - ***M outperforms T*** (**orange**), ***B outperforms T*** (**red**) and ***B outperforms M*** (**yellow**)

Split	MR	CR	SUBJ	MPQA	SST2	SST5	TREC	SICK-E	MRPC
Random-Embeddings	61.65	71.6	78.9	73.79	60.57	31.09	70.0	77.07	69.91
Glove-Full	75.7	77.48	91.76	86.66	78.03	41.0	68.0	78.49	70.61
Glove-T	70.74	73.67	90.1	81.58	72.49	37.24	61.8	75.71	71.94
Glove-M	**72.98**	**75.04**	87.76	**84.07**	**75.34**	**40.5**	57.6	**76.5**	71.42
Glove-B	67.62	73.01	83.68	**81.61**	69.52	36.11	57.0	72.82	70.96
Word2vec-Full	77.65	79.26	90.76	88.3	79.68	42.44	83.0	78.24	72.58
Word2vec-T	74.34	76.29	89.88	85.07	77.16	40.36	70.0	75.46	71.48
Word2vec-M	72.91	73.43	82.39	82.76	72.65	38.69	66.0	70.53	71.36
Word2vec-B	71.42	**74.25**	**82.47**	81.05	**73.48**	38.46	**72.2**	**74.3**	71.01
fastText-Full	67.85	75.39	85.87	79.85	70.57	35.97	68.0	76.66	70.84
fastText-T	69.42	67.76	87.69	84.64	74.35	36.83	74.8	66.04	70.61
fastText-M	68.88	65.3	81.74	81.45	72.1	35.57	65.2	65.01	68.29
fastText-B	66.45	64.21	79.89	79.83	69.96	31.22	**69.4**	63.77	67.94

1, it is evident that the performance saturates consistently at around 200 dimensions for all of the tasks, after which adding new principal components does not lead to much gain in performance.

2.2 Sentence Classification Tasks

The sentence classification tasks (Conneau and Kiela, 2018) include binary classification tasks (MR, CR, SUBJ, MPQA), multiclass classification tasks (SST-FG, TREC), entailment (SICK-E), semantic relatedness (STS-B) and Paraphrase detection (MRPC) tasks. As usual, the evaluation is done by computing the classification accuracy on the test set.

Figure 2 shows the performance (Test accuracy) on 9 standard downstream sentence classification tasks (Conneau and Kiela, 2018) using the same procedure for constructing word embeddings (Glove) as in 2.1. Further, sentence vectors were constructed using an average of the contained word embeddings, which has been demonstrated to be a very strong baseline for downstream tasks (Arora et al., 2017). From Figure 2, we can observe that, similar to the previous word similarity tasks, the performance saturates consistently at around 200 dimensions for all of the tasks, after which incrementing the embeddings with additional principal components does not lead to much gains in performance. We also report results for original $(300D)$ and post processed PCA reduced $(200D)$ word embeddings for other types (fastText, Glove) in Table 2. In Table 2, we also report results with pretrained $200D$ Glove embedding. [5]

Analysis: To conclude, observations from both word similarity and sentence classification tasks, of saturation in performance around 200, much before the original 300 dimensions implies redundancy among the dimensions (in section 3 we will clarify why it doesn't imply *noise*). Furthermore, this observation is consistent across various embedding types (Glove, fastText and word2vec) for the sentence classification tasks, as demonstrated in Table 2. This also suggests a simple strategy to reduce the embedding size wherein one third of the components could be reliably removed without affecting the performance on word similarity or sentence classification tasks, leading to 33% memory reduction.

3 Variance Based Analysis

In this section, we characterize the redundancy observed in Section 2, in terms of variance of the principal components. Specifically, we measure downstream performance (on the sentence classification tasks of Section 2.2) of word embeddings against the amount of variance captured by the principal components (the variance explained or captured by a principal component is the variance of the embeddings when projected onto that principal component; hereon, we refer to the fraction of variance explained by a principal component simply as variance explained by that component). Similar to the previous sec-

[5] word embeddings for $200D$ for other embedding types (fasttext, word2vec) are not publicly available.

Table 2: Performance on sentence classification tasks of various embeddings (300 dimensional) and their post-processed PCA reduced counterparts of 200 dimensions.

Embedding	MR	CR	SUBJ	MPQA	SST2	SST5	TREC	SICK-E	MRPC
Glove	75.7	77.48	91.76	86.66	78.03	41.0	68.0	78.49	70.61
Glove-PCA	74.62	76.95	91.6	85.97	77.16	40.18	66.6	77.02	72.99
Glove-200	74.69	77.91	91.18	86.52	77.98	40.05	66.4	77.47	72.23
Word2vec	77.65	79.26	90.76	88.30	79.68	42.44	83.0	78.24	72.58
Word2vec-PCA	76.53	78.12	90.50	86.74	79.63	41.49	77.6	76.54	72.17
fastText	67.85	75.39	85.87	79.85	70.57	35.9	68.0	76.66	70.84
fastText-PCA	66.83	74.46	85.26	78.91	69.85	36.11	66.0	76.50	68.75

Table 3: The Variance for each of the T, M, B splits of the embeddings.

	Glove	Word2vec	fastText
T	0.529	0.628	0.745
M	0.371	0.221	0.162
B	0.100	0.151	0.093

tion, we use 300 dimensional Glove embeddings (trained on Wikipedia 201 + Gigaword 5^2) for experiments in this section, along with publically released fastText (trained on Wikipedia, UBMC webbase corpus and statmt.org news dataset[3] and Word2vec (trained on the GoogleNews dataset[4]) embeddings, both of 300 dimensions.

For each of the embedding types, we first construct word embeddings using only top 100 principal components (T), the middle 100 principal components (M) and the bottom 100 principal components (B). Then, we compute the variance for each split by aggregating the variance of the 100 principal components of each split for all three embedding types. Table 3 highlights how the total variance is divided across the three splits. The T embeddings have the first 100 principal components (PCs), so the highest variance explained, while the B embeddings have the bottom 100 components, thereby the least variance explained. Furthermore, the variance explained by the principal components for the same split also differ significantly across the different embedding types. For example, fastText has more variance explained, when compared to Glove and Word2vec, for the split T, while Glove has the most variance explained, among the three embedding types, for the split M. Lastly, Word2vec explains more variance than Glove and fastText for the split B. The differences are expected since, the three embedding types differ considerably in their training algorithms. While Word2vec uses negative sampling, Glove derives semantic relationships from the word-word co-occurrence matrix and fastText uses subword information. So, to summarize we constructed altogether 9 embedding splits (3 from each of the 3 embedding types), which differ significantly in terms of the variance explained by their constituent components.

We use the 100 dimensional embedding obtain from the several splits (T, M, B) and types (Glove, fastText, Word2vec) as features for downstream sentence classification tasks, as in Section 2.2, except that, now, each of the embedding feature has 100 dimensions. The experiments are designed to test whether the variance explained by a split is closely correlated with the downstream performance metric (classification accuracy) for each of the three embedding types. Table 1 shows the results on 9 sentence classification tasks, for each embedding split, for all the three embedding types. In the table, the highlighted cells represent the cases where classification accuracy of the lower variance split exceeds that of the corresponding higher variance split. Each annotated cell corresponds to one of the three cases - **M outperforms T** (**orange**), **B outperforms T** (**red**) and **B outperforms M** (**yellow**). For the comparisons between T and M splits, in 6 out of 27 such comparisons, the M embeddings outperform the T embeddings. Similarly, for comparisons between M and B embeddings, the B embeddings outperform the M embeddings in 7 out of 27 cases and for comparisons between T and B embeddings, in 2 out of 27 cases the B embeddings outperform the T embeddings. Further, in a number of cases (although not highlighted), such as on the MRPC task, the T and M splits differ very little in performance. The same is true for M and B splits on tasks such as MPQA and CR. Such cases are least prominent in fastText, probably due to the extremely large gap in the variance explained between the T, M and T, B splits.

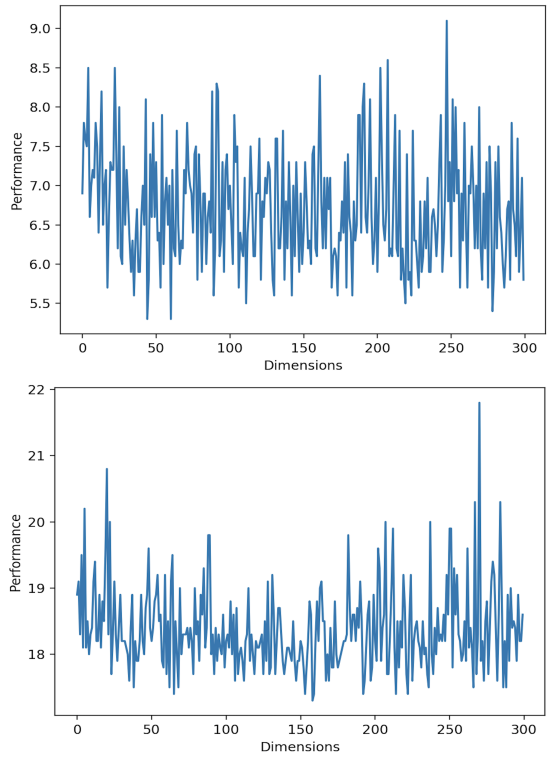

Figure 3: Analysis of individual principal components on the two syntactic information based linguistic probing tasks: TopConst (top) and TreeDepth (bottom). The Y-axis represents the Test accuracy on the two tasks.

Analysis: From Table 1 it is evident that the performance drop between the T, M, B splits is quite low for a number of tasks, which is highly contrary to the expectation, given the large differences in the variance explained (Table 3). Further, there are also many cases where lower variance embeddings (B and M) outperform the emedddings (M and T) with higher variance, for all the three embedding types. These results demonstrate that for word embeddings, *the variance explained by the principal components is not sufficient for explaining their downstream performance. In other words, the variance explained by the principal components is a weak representative of downstream performance.* This is in contrast to the widely used practice of using the variance explained by the principal components as a fundamental tool to assess the quality of the corresponding representations (Jolliffe and Cadima, 2016).

4 Dimensional Linguistic Probing Tasks

A plausible hypothesis to explain the better performance of M and B embeddings (Table 1) in the earlier section is that *'the syntactic information required for downstream sentence classification tasks is distributed independently with respect to the principal components'*. To explore the validity of the proposed hypothesis, we leverage two linguistic probing tasks, namely TreeDepth and TopConst (Conneau et al., 2018). These probing tasks are designed to test whether sentence embeddings are sensitive to the syntactic properties of the encoded sentences. The TreeDepth task (a 8-way classification problem) tests whether the model can predict the depth of the hierarchical syntactic structure of the sentence. For doing well on the TreeDepth task, the embeddings have to group sentences by the depth of the longest path from root to any leaf. In the TopConst task (a 20-way classification problem), a sentence must be classified in terms of the sequence of its constituents occurring immediately below the sentence node of its hierarchical structure. Therefore, for good performance on the TopConst task, the embeddings have to capture latent syntactic structures and cluster them by constituent types. The random baselines for the TreeDepth and TopConst tasks are 12.5 and 5.0 respectively, while full 300-dimensional Glove embeddings obtain accuracies of 37 and 68 percent respectively.

To evaluate the syntactic information contained in each of the principal components, we first construct one-dimensional word embeddings by projecting word vectors onto a single principal component. Then we use these word embeddings to construct sentence vectors, as in Section 2.2, which are used as features for the two classification tasks. For good performance, the single component sentence vector has to distinguish between the probing task's output classes. Therefore, the performance on these tasks can be used to isolate the behavior of individual components with respect to the syntactic information captured. The motivation here is that if the syntactically discriminative components would vary considerably, then we can isolate the behavior of the individual components and see their correspondence with the rank of the principal component. Figure 3 depicts the scores (Test classification accuracy) on TopConst and TreeDepth tasks respectively. The average performance of the one-dimensional representations has mean $\pm$ standard deviation of 18.38 ± 0.64 and 6.71 ± 0.72 for the TreeDepth and TopConst tasks respectively.

Analysis: The average performance of the one-

dimensional representations on both tasks is much lower than full dimension embeddings but well above the random baseline. However, many individual compoenents far exceed the random baseline as well. As mentioned earlier, we wanted to probe whether such discriminativeness is ranked according to variance. However from Figure 3, it is evident that the performance across the dimensions does not have any particular trend (increasing or decreasing) w.r.t to the rank of the principal components. In fact, the peak performance on both the tasks is achieved by a component in the bottom (B) split of the embeddings. This validates the hypothesis that *the syntactic information captured by a principal component is independent of the amount of variance it explains*.

Table 4: Classification Accuracy for Linguistic Probing Tasks using the **T**, **M**, **B** splits of the embeddings. Here also, the highlighted cells correspond to one of the three cases - ***M outperforms T*** (**orange**), ***B outperforms T*** (**red**) and ***B outperforms M*** (**yellow**)

Embedding	**TopConst**	**TreeDepth**
Glove-T	28.1	28.2
Glove-M	26.0	24.8
Glove-B	**27.1**	**26.9**
Word2vec-T	23.9	42.5
Word2vec-M	**24.3**	**43.5**
Word2vec-B	23.7	**44.6**
fastText-T	31.2	50.7
fastText-M	29.3	**51.0**
fastText-B	**30.6**	**56.8**

To further validate the hypothesis, we repeat the experiment described in Section 3 for each of the embedding types, except on the synctatic probing tasks of TopConst and TreeDepth in Table 4. Similar to Table 1, each annotated cell in Table 4 corresponds to one of the three cases - **M outperforms T** (**orange**), **B outperforms T** (**red**) and **B outperforms M** (**yellow**). For the comparisons between T and M splits, in 3 out of 6 such comparisons, the M embeddings outperform the T embeddings. Similarly, for comparisons between M and B embeddings, the B embeddings outperform the M embeddings in 5 out of 6 cases and for comparisons between T and B embeddings, in 2 out of 6 cases the B embeddings outperform the T embeddings. In other words, table 4 shows that for the TreeDepth task, the B embeddings significantly outperform T and M embeddings for word2vec and fastText, whereas for Glove, it outperforms the M embeddings. For the TopConst task as well, the B embeddings outperform M embeddings for Glove and fastText, whereas for Word2vec, it outperforms the T embeddings. Thus, the discrepancy in performance on these syntactic probing tasks is even more severe when compared to the sentence classification tasks evaluated in Section 3. The results also validate our hypothesis that *the variance explained by the embeddings is of little predictive strength in predicting its relative performance*.

5 The Post Processing Algorithm (PPA)

In this section, we briefly describe and then evaluate the post-processing algorithm (PPA) by (Mu and Viswanath, 2018), which achieves high scores on Word and Semantic textual similarity tasks (Agirre et al., 2012). The algorithm (PPA) is listed below as Algorithm 1. PPA removes the projections of top principal components from each of the word vectors, making the individual word vectors more discriminative. The algorithm could be regarded as pushing the word embeddings towards a more isotropic space (Arora et al., 2016), by eliminating the common parts (mean vector and top principal components of the embedding space) from the individual word embeddings. However, it is worth revisiting the assumption whether isotropy (or angular isotropy more specifically) of the embedding space is universally beneficial with respect to downstream tasks. In this section, we stress test this assumption on a range of sentence classification and machine translation tasks. Our fundamental intuition is that since these tasks require the embedding space to capture syntactic properties much more significantly than word-similarity tasks, enforcing isotropy could lead to worse performance.

Algorithm 1: Post Processing Algorithm PPA(X, D)

Data: Embedding Matrix X, Threshold Parameter D
Result: Post-Processed Word Embedding Matrix X
`/* Subtract Mean Embedding */`
X = X - $\overline{X}$;
`/* Compute PCA Components */`
u_i = PCA(X), where i = 1, 2, . . . d;
`/* Remove Top-D Components */`
for ***all*** *v in X* **do**
    $v = v - \sum_{i=1}^{D}(u_i^T \cdot v)u_i$
end

Table 5: Performance on sentence classification tasks of various embeddings and their post-processed (PPA) counterparts. The **red** colored cells denote the cases where the original embeddings outperformed their post-processed (PPA) counterparts.

Embedding	MR	CR	SUBJ	MPQA	SST2	SST5	TREC	SICK-E	MRPC
Glove (300 dim)	75.7	77.48	91.76	86.66	78.03	41.0	68.8	78.49	70.61
PPA on Glove	**75.57**	77.48	**91.01**	86.67	**77.98**	**40.72**	**65.8**	78.53	71.59
Word2vec (300 dim)	77.65	79.23	90.76	88.30	79.68	42.44	82.6	78.24	72.64
PPA on Word2vec	**77.33**	79.5	**90.59**	**88.12**	**79.41**	42.71	83.4	78.26	**72.58**
fastText (300 dim)	74.16	71.63	89.56	87.12	79.24	39.14	79.4	72.34	70.14
PPA on fastText	74.59	71.63	**89.4**	**86.9**	**79.13**	39.64	80.2	72.36	**70.09**

5.1 Sentence Classification Tasks

We compare the performance of PPA (with a constant D=5 across all the embeddings) on the 9 downstream sentence classification tasks, as in Section 3. The results are presented in Table 5. In our work, we adhere to the linear evaluation protocol and use a simple logistic regression classifier in evaluating word representations (Arora et al., 2019; Gupta et al., 2020), whereas (Mu and Viswanath, 2018) use a neural network as their classifier. The **red** colored cells in Table 5 denote the cases where the original embeddings outperformed their Post Processed (PPA) counterparts. Such cases occurred in 14 out of 27 comparisons in Table 5. The results in Table 5 show that post-processing doesn't always lead to accuracy gains and can be counterproductive in a number of tasks.

Analysis: The results in Table 5 are contrary to the expectation that pushing the word embeddings towards isotropy would lead to better downstream performance. This suggests that within the context of downstream sentence classification tasks, projecting word vectors away from the top components leads to a loss of 'useful' information. To explain this loss of 'useful' information, we could use the analysis from Figure 3. From Figure 3, it is evident that the top dimensions also contain syntactic information, the loss of which adversely impacts downstream classification tasks, which by construction, benefit from both semantic and syntactic information. Also, by just removing the mean (no top component nullification as in PPA), we notice almost zero change in performance for most of the sentence classification tasks in Table 5 (the highest change was for TREC, of -0.4, still quite low when compared to -4.4 for PPA), which demonstrably shows that removing the mean must be ruled out as the possible cause for the drop in classification accuracies.

On the same tasks, we also observe a drop in sentence classification accuracy (2.37, 1.99, 3.94 average drop on word2vec, Glove, fastText respectively) using 150 dimensional embeddings obtained from PPA based dimensionality reduction (Raunak et al., 2019). This shows that the variance based post-processing algorithms such as PPA (Mu and Viswanath, 2018) and PPA-PCA (Raunak et al., 2019), when used in *downstream tasks* have significant limitations, which could be attributed to the loss of syntactic information.

5.2 Machine Translation

Recently, (Qi et al., 2018) have shown that pre-trained embeddings lead to significant gains in performance for the translation of three low resource languages namely, Azerbaijani (AZ), Belarusian (BE) and Galician (GL) into English (EN). Here, we demonstrate the impact of the post processing algorithm on machine translation (MT) tasks. We replicate the experimental settings of (Qi et al., 2018) and use a standard 1 layer encoder-decoder model with attention (Bahdanau et al., 2015) and a beam size of 5. Prior to training, we initialize the encoder with fastText word embeddings (no other embeddings are publically available for these languages) trained on Wikipedia[6]. We then use PPA on the pre-trained embeddings and train again. The results of the experiments are presented in Table 6.

Analysis: From the results, it is evident that removing the top principal component(s) leads to a consistent drop in BLEU scores across the three language pairs. The observations are consistent with the previous section, in that removing top components hurts performance in non-similarity based tasks. This can again be explained using the analysis from earlier section i.e. instead of strengthening the embeddings, removing the top components leads to a loss of 'useful' information

[6] https://bit.ly/2WkHQ0Y

for the Machine translation task. Further, similar to the previous section, we can specifically attribute the performance drop to the loss of syntactic information, since the top components are at least as equally important for syntactic information as the other components, thus, nullifying them hurts performance.

Table 6: BLEU scores over three different low-resource language pairs with pretrained emebddings and Top D components removed using PPA. **Green** cells denotes top scores.

	AZ->EN	BE->EN	GL->EN
Pre-Trained	**3.24**	**6.09**	**15.91**
PPA (D = 1)	3.19	6.02	14.81
PPA (D = 2)	3.07	5.50	13.88
PPA (D = 3)	3.04	5.26	13.27
PPA (D = 4)	2.92	4.75	13.24

5.3 Summary and Discussion

To summarize our experiments on variance based post-processing, we conclude the following:

1. We can not rely on principal components for manipulating word embeddings as freely as the current literature suggests. While eliminating the 'common parts' helps improve the discriminativeness between the word embeddings (thereby refining the word similarity scores), pushing the embeddings towards angular isotropy does not lead to performance gain in downstream tasks, e.g. sentence classification and machine translation. Although, we did not assume any generative model for the embeddings in any of the explanations (unlike (Arora et al., 2016), which makes use of the isotropy assumption to explain empirical observations in factorizing the PMI matrix), our work further casts doubt on the isotropy assumption for word embeddings and suggests that non-isotropy may be integral to performance on downstream tasks.

2. Furthermore, worse performance in non-similarity tasks can be attributed to the loss of syntactic information contained in the top components, suggesting that the specific geometry created through the 'common parts' is integral to embeddings capturing syntactic properties. Establishing a link between the syntactic properties of the embedding space and its non-isotropy would be an interesting direction to explore for future work.

6 Related Work

Due to the widespread utility of word embeddings, a number of recent works have explored further improving the embeddings post-hoc, as well as trying to better understand and manipulate the geometry of the embedding space.

Post-Processing Word Embeddings A number of recent works have been proposed to enhance word embedding quality post-hoc (Mrkšic et al.; Faruqui et al., 2014; Mu and Viswanath, 2018). Their applications range from better modeling semantic similarities, improving downstream classification performance to dimensionality reduction of the embeddings (Raunak et al., 2019).

Word Embedding Geometry The linear algebraic structure emergent in word embeddings has received considerable attention (Allen and Hospedales, 2019; Arora et al., 2018), and theoretical links have been established between neural embedding algorithms and factorization based techniques (Levy and Goldberg, 2014). Another prominent line of work has been along the direction of probing tasks (Conneau and Kiela, 2018), which use proxy classification tasks to comparatively measure the presence of certain syntactic/semantic properties in the embedding space.

Our work focuses on the dimensional properties of the embedding space in the principal component basis, and also analyzes a few post-processing algorithms, thus contributing to the existing literature on both the areas of embedding analysis.

7 Conclusion and Future Work

To conclude, besides elucidating redundancy in the word embedding space, we demonstrate that the variance explained by the word embeddings' principal components is not a reliable proxy for the downstream utility of the corresponding representations and that the syntactic information captured by a principal component does not depend on the amount of variance it explains. Further, we show that variance based post-processing algorithms such as PPA is not suitable for tasks which rely more on syntax, such as sentence classification and machine translation. Going further, we wish to explore whether the geometric intuitions developed in our work could be leveraged for contextualized embeddings such as ElMo (Peters et al., 2018), BERT (Devlin et al., 2019), and Roberta (Liu et al., 2019), etc.

References

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 385–393. Association for Computational Linguistics.

Carl Allen and Timothy Hospedales. 2019. Analogies explained: Towards understanding word embeddings. In *International Conference on Machine Learning*, pages 223–231.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016. A latent variable model approach to pmi-based word embeddings. *Transactions of the Association for Computational Linguistics*, 4:385–399.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. *International Conference of Learning Representation*.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2019. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association of Computational Linguistics*, 5(1):135–146.

Alexis Conneau and Douwe Kiela. 2018. Senteval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loic Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2126–2136.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2015. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1606–1615.

Manaal Faruqui and Chris Dyer. 2014. Community evaluation and exchange of word vectors at wordvectors. org. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 19–24.

Vivek Gupta, Ankit Saw, Pegah Nokhiz, Praneeth Netrapalli, Piyush Rai, and Partha Talukdar. 2020. P-sif: Document embeddings using partition averaging. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Ian T Jolliffe and Jorge Cadima. 2016. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. In *Advances in neural information processing systems*, pages 2177–2185.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

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

David Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2873–2878.

Nikola Mrkšic, Diarmuid OSéaghdha, Blaise Thomson, and pages=142–148 year=2016 Gašić, Milica and Rojas-Barahona, Lina and Su, Pei-Hao and Vandyke, David and Wen, Tsung-Hsien and

Young, Steve, booktitle=Proceedings of NAACL-HLT. Counter-fitting word vectors to linguistic constraints.

Jiaqi Mu and Pramod Viswanath. 2018. All-but-the-top: Simple and effective postprocessing for word representations. In *International Conference on Learning Representations*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 529–535.

Vikas Raunak, Vivek Gupta, and Florian Metze. 2019. Effective dimensionality reduction for word embeddings. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 235–243, Florence, Italy. Association for Computational Linguistics.

Zi Yin and Yuanyuan Shen. 2018. On the dimensionality of word embedding. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 895–906. Curran Associates, Inc.