OPTIMIZING THE PERFORMANCE OF TEXT CLASSIFICATION MODELS BY IMPROVING THE ISOTROPY OF THE EMBEDDINGS USING A JOINT LOSS FUNCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent studies show that the spatial distribution of the sentence representations generated from pre-trained language models is highly anisotropic, meaning that the representations are not uniformly distributed among the directions of the embedding space. Thus, the expressiveness of the embedding space is limited, as the embeddings are less distinguishable and less diverse. This results in a degradation in the performance of the models on the downstream task. Most methods that define the state-of-the-art in this area proceed by improving the isotropy of the sentence embeddings by refining the corresponding contextual word representations, then deriving the sentence embeddings from these refined representations. In this study, we propose to improve the quality and distribution of the sentence embeddings extracted from the [CLS] token of the pre-trained language models by improving the isotropy of the embeddings. We add one feed-forward layer, referred to as the Isotropy Layer, between the model and the downstream task layers. We train this layer using a novel joint loss function that optimizes an isotropy quality measure and the downstream task loss. This joint loss pushes the embeddings outputted by the Isotropy Layer to be more isotropic, and it also retains the semantics needed to perform the downstream task.
The proposed approach results in transformed embeddings with better isotropy, that generalize better on the downstream task. Furthermore, the approach requires training one feed-forward layer, instead of retraining the whole network. We quantify and evaluate the isotropy through multiple metrics, mainly the Explained Variance and the IsoScore. Experimental results on 3 GLUE datasets with classification as the downstream task show that our proposed method is on par with the state-of-the-art, as it achieves performance gains of around 2-3% on the downstream tasks compared to the baseline.

1 INTRODUCTION

Recent Transformer-based models have achieved significant success on various natural language processing tasks (Kalyan et al., 2021). However, Ethayarajh (2019) observes that some language models including Bidirectional Encoder Representations from Transformers (BERT) produce contextualized word representations that are not isotropic. In other words, the information in the embeddings is not uniformly distributed in all directions in the space. This is not desirable as these representations vary the most in top directions, which limits the expressiveness of the space. Gao et al. (2019) referred to this problem as the representation degeneration problem. Even though researchers did not agree on the source of anisotropy, having an isotropic space is very desirable as the more isotropic the space is, the more diverse the embeddings are. Furthermore, having an isotropic space affects the optimization of the model (i.e., convergence and accuracy), and leads to improvement in the performance of the model(Wang et al., 2020).
As mentioned previously, BERT-based models suffer from the problem of having an anisotropic space. This affects the representation capacity of the embedding space and affects the accuracy of the downstream task. More specifically, Rajaee & Pilehvar (2021b) highlighted that the Classification ([CLS]) token representations are much more anisotropic than all representations in the fine-tuned space. The authors highlighted that this problem becomes even more dramatic after fine-tuning, as

this process tends to concentrate information about the target task in the dominant directions (i.e., the top principal components). Furthermore, Rajaee & Pilehvar (2021b) showed that improving the isotropy, in general, does not immediately result in a better performance for the model. Therefore, we propose to learn an embedding transformation that renders the [CLS] embeddings more isotropic without losing the information of the target task; once improved, the embedding space should exhibit better statistical properties, which should result in a better performance on the downstream task (i.e., increase in the model performance). Thus, we proceed by freezing the parameters of the fine-tuned model and the downstream task layer, and adding an Isotropy Layer between these two models. This Isotropy Layer is one feed-forward layer, which goal is to output embeddings of better isotropy. The parameters of this layer are learned using a joint loss function that combines an isotropic loss function and the downstream task loss function.

We apply empirical methods to quantitatively measure the improvement of the models in terms of isotropy and performance on the downstream task. The improvement in isotropy is evaluated by computing two measures of isotropy, mainly the isoscore and the explained variance, while the improvement in the model performance is evaluated by computing a dataset-specific metric. Two main experiments are carried out. The first experiment compares the proposed method to the baseline (i.e., finetuned model with no Isotropy Layer), while the second experiment compares the method to the Isotropic Batch Normalization (IsoBN) method (Zhou et al., 2021). To the best of our knowledge, our work is the second study besides IsoBN Zhou et al. (2021) that aims to improve the isotropy of the [CLS] token representations. The main contributions of this work are as follows:

1. We provide a method to improve the isotropy of the embeddings by adding an Isotropy Layer at the output of the finetuned language model, and only training this layer using a joint loss function.
2. As shown by Rajaee & Pilehvar (2021b), it is not sufficient to only improve the isotropy of the embeddings, as the embeddings need to maintain the semantics required for the downstream task. Therefore, we propose a novel joint loss function that optimizes both an isotropic loss measure as well as a downstream task loss, and results in embeddings that are more isotropic and that perform better on the downstream task. The isotropic loss is based on the IsoScore. This joint loss function should encourage to include unsupervised quality measures inside the loss function to enforce some statistical properties on the model.
3. We evaluate our method of improving the isotropy of embeddings on multiple datasets from the General Language Understanding Evaluation (GLUE) benchmark for several downstream tasks and compare its performance with the IsoBN method. Experimental results on 3 GLUE datasets demonstrate that our method can improve isotropy significantly, as well as improve the model performance.

This paper is structured as follows: Section 2 introduces the concept of Isotropy as well as the related work that is relevant to the study. Section 3 presents the proposed method and describes the implementation of this method. Section 4 describes the experimental evaluation as well as the results obtained from these experiments. Finally, Section 5 provides the summary of the findings and conclusions as well as the future scope of this study.

2 Isotropy

Isotropy is a geometric property that assesses the distribution of the points in space Biś et al. (2021). In an ideally isotropic space, the embeddings are uniformly distributed in all directions of the space, i.e., the embeddings are not biased in a specific direction.

2.1 Properties

Isotropy has been linked to multiple properties in space. For instance, in an anisotropic space, randomly sampled words tend to be highly similar to one another when measured by cosine similarity (Ethayarajh, 2019). Furthermore, the representations exhibit word-frequency bias, as the high-frequency words concentrate densely in the embedding space while low-frequency words disperse sparsely in the space (Li et al., 2020).

Multiple studies tried to explain the source of the anisotropy. Timkey & van Schijndel (2021) showed that the anisotropy in contextual models is a product of rogue dimensions of the entire embedding space. These rogue dimensions drive the similarity metrics, explaining the high similarity property between random embeddings of these spaces. However, the authors mentioned that these models still perform well as their behavior is not greatly affected by these rogue dimensions. Their analysis showed that these dimensions handle a small subset of the model's linguistic abilities (i.e., punctuation and positional information). Furthermore, Biś et al. (2021) showed that embeddings do not occupy a narrow cone, but rather only appear as a cone when projected to a lower-dimensional space. The authors showed that during training, word embeddings share the same direction gradients, therefore are shifted in one dominant direction in the vector space.

2.2 Measures

There is a need to approximate the degree of isotropy of the space, i.e., the spatial utilization of the embeddings. As mentioned previously, an isotropic space has embeddings that are distributed uniformly in all dimensions. In other words, the embeddings have elongations that are similar across different directions of the space. Therefore, methods that are based on the Principle Component Analysis are the most appropriate to find and study the most elongated directions of the space. We present the two most robust PCA-based methods to quantify the isotropy, mainly the explained variance ratio and the IsoScore as highlighted by Rudman et al. (2022).

Explained Variance Ratio: The explained variance ratio, which we refer to as EV_k Score, measures how much total variance is explained by the first k principal components of the data. This metric measures the difference in variance in different directions of the space. However, computing it requires the specification of a certain number of Principle Components (PCs). Therefore, in this study, we will be numerically examining this score for the first three PCs, and graphically for the top components. Given that λ_i is the i^{th} largest singular value of the embeddings matrix E, the variance explained ratio is computed as follows:

$$EV_k(E) = \frac{\sum_{i=1}^{k} \lambda_i^2}{\sum_{i=1}^{D} \lambda_i^2} \tag{1}$$

IsoScore: The IsoScore (Rudman et al., 2022) of an embedding space can be interpreted as the fraction of dimensions uniformly used by the embedding space. This score is derived from an isotropy defect that is calculated by computing the distance between the identity matrix and the normalized covariance matrix of the PCA-reoriented data. The IsoScore scales linearly with the number of dimensions used and is stable when distributions contain highly isotropic subspaces. A high IsoScore (i.e., close to 1.0), indicates that the principal components are uniformly distributed across all dimensions of space, implying that the space is isotropic. However, a small IsoScore (i.e., close to 0.0) indicates that the first components explain almost all the variance of the data, implying a highly anisotropic space.

2.3 Related Work

In this section, we present some related work aiming to solve the representation degeneration problem and improve the isotropy of the space. We can split the studies into two: (1) studies that regularize the embeddings during the training stage and (2) studies that post-process the embeddings after the training phase.

Regularizing the embeddings: Multiple studies applied regularization to improve the isotropy of the learned embeddings. Firstly, Gao et al. (2019) employed cosine regularization to decrease the similarity between the embeddings and increase the representation power of the space. However, Zhang et al. (2020) proposed the Laplacian regularization as a better alternative to cosine regularization, as it minimizes the similarities between the embeddings with similar contexts (instead of applying it to all embeddings). Moreover, Wang et al. (2020) mitigated the fast singular value decay phenomenon of anisotropic space using spectrum control. Finally, Gong et al. (2018) learned embeddings of better isotropy by alleviating the word frequency bias of anisotropic spaces using adversarial training.

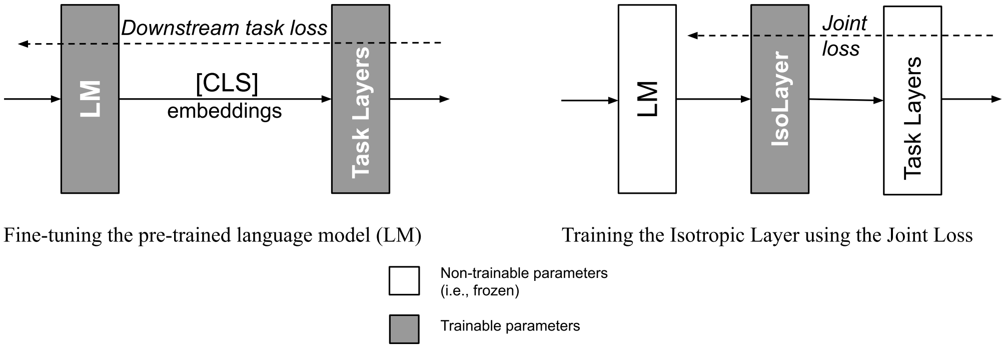
Postprocessing the embeddings: Other studies proposed to post-processes the learned embeddings to improve the isotropy of the space. Firstly, Mu & Viswanath (2018) introduced the All-but-the-top method that removes the common vector and dominant directions from the embeddings, rendering them more isotropic. Secondly, Rajaee & Pilehvar (2021a) increased the isotropy by clustering the embeddings and nulling the principal components of each cluster. Third, Liang et al. (2021) applied a weighted removal of a selected number of dominant directions from the embedding. The weights were learned through a word similarity task applied to the embeddings.

Discussion: Our method of improving the isotropy of embeddings is similar to the regularization-based approaches as it improves the embeddings through a penalty term. However, the measure used in our method is unsupervised and more robust, and it directly estimates isotropy instead of computing an indicator of isotropy. Furthermore, our method is not as computationally expensive as the regularization methods, as it only trains one feed-forward layer instead of the whole language model. Our method is also a form of postprocessing of the embeddings, as the layer introduced transforms the embeddings and makes them more isotropic, without compromising the modeling power of the space.

3 Improving the isotropy of embeddings using an isotropic layer and a joint loss

As mentioned previously, we limit the scope of our work to the embedding space of the [CLS] representations. To our knowledge, the only study besides ours which improves the [CLS] embeddings is the IsoBN Zhou et al. (2021). In their study, Zhou et al. (2021) first highlighted the anisotropic nature of the [CLS] embeddings. Then, they proposed to improve the isotropy of these embeddings using an isotropic variant of the batch normalization method. Furthermore, the downstream task considered in our study is classification. However, we could generalize the method to different downstream tasks.

The proposed approach can be visualized in Figure 1. In summary, the approach consists of adding a layer, referred to as the Isotropy Layer, between the pre-trained language model and the downstream task layers. This Isotropy Layer is responsible for transforming the [CLS] embeddings of BERT into embeddings with a better isotropic property. Since the goal of the study is to improve the isotropy of the space to perform better on the downstream task, we condition the learning process by freezing the parameters of the downstream task layers as well as the language model (preserving the knowledge acquired to solve the downstream task). Since only this layer is updated during training, we are learning a clear transformation of the space; this transformation post-processes the embeddings to improve the distribution of the embeddings in the space while keeping the semantics needed to perform the downstream task. The process can be summarized in Figure 1. We present the details of the approach in the following subsections.



Fine-tuning the pre-trained language model (LM)

Training the Isotropic Layer using the Joint Loss

Figure 1: Diagram summarizing the approach described in the study.

3.1 Fine-tuning the language model using the downstream task loss

Our proposed approach assumes that the pre-trained language model is first fine-tuned on the downstream task. This is done by extracting the [CLS]embeddings from the language model and feeding these embeddings to a predefined set of downstream task layers. Then, both the pre-trained language model and the downstream task layers will be trained to perform the downstream task. Fine-tuning the model on a downstream task allows us to leverage the knowledge encoded in this model, and transfer it to perform the task at hand.

3.2 Freezing the network and adding Isotropy Layer

We insert one feed-forward layer, called the Isotropy Layer, between the fine-tuned model and the downstream task layers. The goal of the Isotropy Layer is to transform the [CLS] embeddings outputted by the fine-tuned model to a new space that is more isotropic. It should be noted that this layer could be replaced by a more complex neural network, with the only limitation that the output of this neural network should be of the same size as its input (i.e., the embedding vector outputted by the pre-trained model), to ensure compatibility of the output of the network with the downstream task layers.
As mentioned previously, improving the isotropy by itself is not sufficient (Rajaee & Pilehvar, 2021b). Therefore, the Isotropy Layer needs to maintain the semantics needed to perform the downstream task at hand. To do so, we perform the following:

- We freeze the parameters of the fine-tuned model. As we know, the fine-tuned model has learned some part of the knowledge required to perform the downstream task. Freezing this fine-tuned model preserves the knowledge learned.
- We freeze the parameters of the downstream task layers. As mentioned previously, the output of the Isotropy Layer (i.e., the transformed embeddings) needs to maintain the semantics needed to perform the downstream task. Therefore, we freeze these layers to condition the output of the Isotropy Layer to adjust to the knowledge of this layer during the training.

3.3 Training the Isotropy Layer using a Joint Loss

Joint Loss: Now that the fine-tuned model and the downstream task layers are both frozen, we train the Isotropy Layer using the proposed joint loss function:

$$\mathcal{L} = \alpha \times \mathcal{L}_1 + (1 - \alpha) \times \mathcal{L}_2 \tag{2}$$

We define the variables in the equation as the following:

1. $\mathcal{L}_1$ is the loss used to fine-tune the pre-trained model. It varies with the downstream task (i.e., CrossEntropy for the classification, Mean Squared Error for regression tasks). This measure is computed over the output of the new network, i.e., the embeddings are fed to the pre-trained model, transformed through the Isotropy Layer, then go through the downstream task layers to generate an output. This term ensures that the transformed embeddings maintain the semantic information required for the downstream task.
2. $\mathcal{L}_2$ quantifies the degree of the isotropy of the space of embeddings at the output of the Isotropy Layer. It is an unsupervised measure computed over a mini-batch of embeddings at a time. Intuitively, the bigger the batch of embeddings, the more accurate the isotropy measure is. This term acts as a regularizer for the new embeddings, and it pushes the weights of the Isotropy Layer to produce more isotropic embeddings.

   We propose to use the IsoScore as the measure of quality used to compute $\mathcal{L}_2$. We usually desire to optimize a decreasing function. Knowing that the IsoScore increases for a better isotropic space, we propose to use the logarithmic of the inverse of the IsoScore. This decision was taken due to its sensitivity to the change in the measure used (i.e., IsoScore).

Balancing both losses: Ideally, setting α to 0.5 should result in equally acceptable isotropic property and downstream task performance. However, both losses in the joint loss operate around different scales. Therefore, the learning process might be biased toward one of the losses and might optimize one of the losses at the expense of the other one. Inspired by the work done by Zabihzadeh (2021), we overcome this issue by normalizing the losses by computing a Moving Average as follows:

1. We keep track of the set of mean values of each loss $\overline{\mathcal{L}}_1$ and $\overline{\mathcal{L}}_2$, as well as their average $\overline{\mathcal{L}}$.
2. We estimate the set of normalized losses as $\hat{\mathcal{L}}_i = \mathcal{L}_i \frac{\overline{\mathcal{L}}}{\overline{\mathcal{L}}_i}$ where $i \in \{1, 2\}$.
3. At iteration k of the learning process, we update the set of mean values using exponential moving average, where s is the smoothing factor (set as 0.15, as in Zabihzadeh (2021)).

$$\overline{\mathcal{L}_i^{new}} = \mathcal{L}_i \times \frac{s}{1+k} + \overline{\mathcal{L}}_i \times (1 - \frac{s}{1+k}) \tag{3}$$

4. The loss used to train the Isotropy Layer is $\hat{\mathcal{L}} = \frac{1}{2}(\hat{\mathcal{L}}_1 + \hat{\mathcal{L}}_2)$.

4 EXPERIMENTS

To evaluate the proposed method, we conducted two main experiments. The first experiment evaluates the proposed method and compares it to our baseline (i.e., the text classification system without the Isotropy Layer) in terms of IsoScore, Explained Variance, and performance measurement. The second experiment compares the proposed method with the IsoBN method (Zhou et al., 2021).

4.1 EXPERIMENTAL SETUP

Datasets: To evaluate our approach, we used multiple datasets from the GLUE benchmark (Wang et al., 2018). The GLUE benchmark is a collection of Natural Language Understanding (NLU) tasks including question answering, sentiment analysis, and textual entailment. GLUE datasets favor models that learned to represent linguistic knowledge for sample-efficient learning and knowledge transfer across tasks (Wang et al., 2018). Each dataset has its metric to evaluate the model performance. We selected the three specific datasets described in Table 1 because they represent the three main tasks of the GLUE benchmark, which are Inference Tasks (RTE), Single-Sentence Tasks (CoLA), and Similarity and Paraphrase Tasks (MRPC). We evaluate the classification on the dev sets that were provided by these datasets [1].

Table 1: Details of the datasets used in the experimental evaluation.

Dataset ID	Dataset Name	Dataset Description	Performance Metric
RTE	Recognizing Textual Entailment	Determines whether each sentence entails a given hypothesis or not	Accuracy
CoLA	Corpus of Linguistic Acceptability	Determines whether each sentence is grammatically correct or not	Mathew's correlation coefficient
MRPC	Microsoft Research Paraphrasing Corpus	Consists of a pair of sentences and determines whether the sentences are paraphrases from one another	Accuracy

Models[2]: The models were implemented using the transformers library provided by HuggingFace (Wolf et al., 2019) using PyTorch. The optimizer used is AdamW(Loshchilov & Hutter, 2017). Early stopping was applied according to task-specific metrics on a validation set (train/validation split of

[1] The labels of the test sets were not provided (they are only evaluated through the leaderboard at https://gluebenchmark.com/leaderboard). This setup also follows the work done by IsoBN.

[2] More information regarding hyper-parameter tuning is available in a technical report (Not disclosed due to the double-blind review)

70/30). The approach is evaluated on two pre-trained language models, mainly BERT-base-cased and RoBERTa-large, as these models perform well for the English language. Since the datasets used are binary classification datasets, the downstream task layers consist of only one classification layer of 2 neurons. The activation function used is the softmax function and the loss used is the binary cross-entropy loss. The Isotropy Layer is a one-layer feed-forward neural network, with a number of neurons of the same size as the [CLS] embedding extracted from the pre-trained language model (768 in the case of BERT and 1024 in the case of RoBERTa). We could have opted out for a more complex neural network. We leave this direction for future studies.

4.2 Comparing the proposed approach to the baseline

4.2.1 Model performance and IsoScore

We proceed by applying our approach and evaluating the performance of the model per dataset performance metric and the isotropy of the transformed embedding space using the IsoScore. These measures have been computed over the dev set. Results are displayed in Table 2.

		bert-base-case		**roberta-large**	
Dataset	**Method**	**Performance**	**IsoScore**	**Performance**	**IsoScore**
RTE	Finetuned Language Model (LM)	67.87	0.0051	85.56	0.0049
	Finetuned LM + Isotropy Layer	71.84	0.025	87.36	0.0145
	Improvement of the approach	**+5.8%**	**+390.2%**	**+2.1%**	**+195.9%**
CoLA	Finetuned Language Model (LM)	61.61	0.004	67.23	0.0012
	Finetuned LM + Isotropy Layer	63.57	0.0255	68.77	0.0023
	Improvement of the approach	**+3.2%**	**+537.5%**	**+2.3%**	**+91.67%**
MRPC	Finetuned Language Model (LM)	85.29	0.0033	90.93	0.0016
	Finetuned LM + Isotropy Layer	87.99	0.0103	91.17	0.00245
	Improvement of the approach	**+221.2%**	**+221.2%**	**+0.26%**	**+53.13%**

Table 2: Internal evaluation of the approach on 3 GLUE benchmarks. We can observe a significant increase in the isoscore and a notable increase in task performance.

4.2.2 Explained variance

We examine the explained variance curve of the models trained by computing the metric over the top K=20 principal components. The results are displayed in Figure 2. We can see that using the Isotropy Layer resulted in embeddings with a smaller explained variance compared to the baseline. This means that the singular values distribute more uniformly in the transformed space, inferring that the information is spread across more principal components uniformly (the space is more isotropic). We also notice that the proposed approach had limited improvement in explained variance for RoBERTa, compared to BERT. We provide multiple explanations in the discussion section for such behavior.

4.3 Comparing the proposed approach with IsoBN

We compare the proposed approach to IsoBN, the only approach in the literature besides ours that aims to improve the isotropy of the [CLS] embeddings. To prove that the improvements incurred by our approach are consistent, we run the approach on each dataset with 5 random seeds. Furthermore, we measure the isotropy by examining the explained variance of the top 3 principal components. As we can see, our approach provides consistent improvements in both performance and isotropy. We notice that our approach is on par with IsoBN in terms of model performance. As for the isotropy, we can see that our approach results in better explained variance for all BERT models, while IsoBN results in better explained variance for the RoBERTa models. This is interpreted in the discussion section.

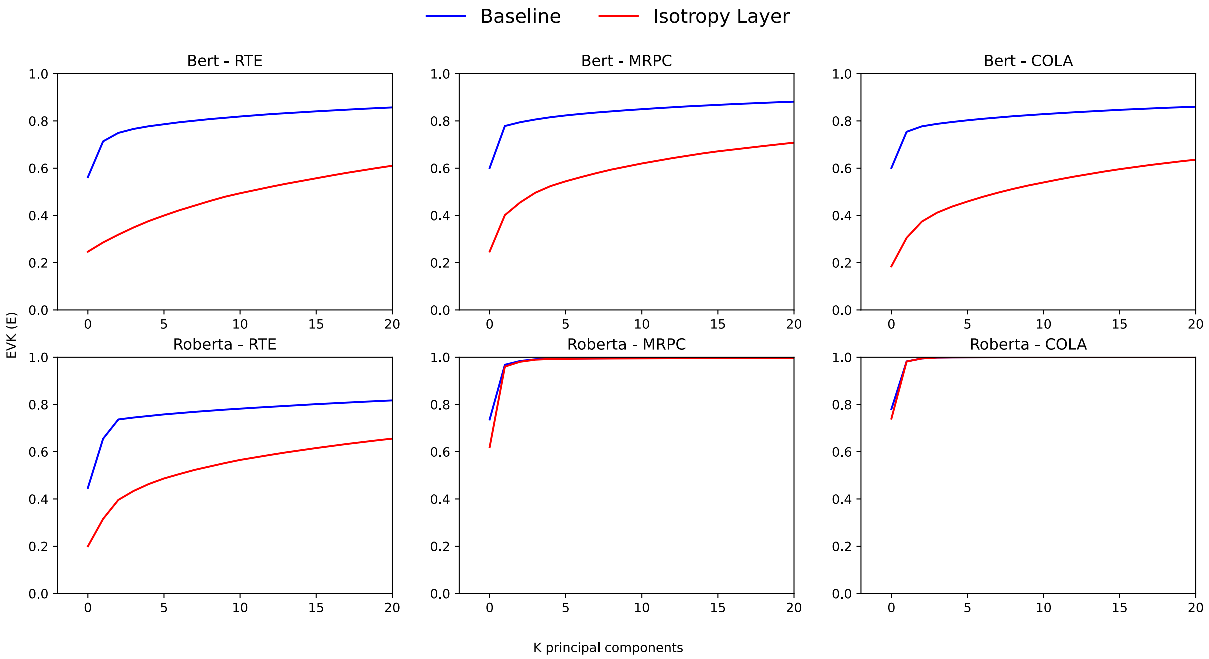


Figure 2: The plots present the explained variance of the top principal components on 3 selected datasets. These plots show that the proposed method results in a smaller explained variance compared to the baseline, which indicates that the variations of the embeddings tend to distribute equally in all directions (i.e., more isotropic space).

Dataset	**Method**	**Performance Measure**		**Isotropic Measure**	
		IsoBN	**IsoLayer**	**IsoBN**	**IsoLayer**
RTE	BERT-base	67.87 (1.1)	67.72 (0.83)	0.88/0.93/0.95	0.61/0.75/0.78
	BERT-base+Isotropy	70.75 (1.6)	70.40 (1.05)	0.49/0.72/0.85	0.27/0.35/0.38
	RoBERTa-L	84.47 (1.0)	85.56 (0.8)	0.53/0.66/0.70	0.44/0.65/0.73
	RoBERTa-L+Isotropy	87.00 (1.3)	87.36 (0.6)	0.15/0.29/0.37	0.19/0.31/0.39
CoLA	BERT-base	60.72 (1.4)	60.89 (0.81)	0.49/0.58/0.64	0.60/0.75/0.77
	BERT-base+Isotropy	61.59 (1.6)	62.82 (0.85)	0.25/0.37/0.48	0.18/0.30/0.32
	RoBERTa-L	68.25 (1.1)	67.33 (0.9)	0.83/0.88/0.90	0.66/0.87/0.91
	RoBERTa-L+Isotropy	69.70 (0.8)	68.77 (0.8)	0.21/0.38/0.49	0.41/0.63/0.76
MRPC	BERT-base	85.29 (0.9)	86.64 (1.09)	0.76/0.87/0.89	0.63/0.80/0.81
	BERT-base+Isotropy	87.5 (0.6)	87.5 (0.42)	0.37/0.68/0.77	0.43/0.49/0.52
	RoBERTa-L	90.68 (0.9)	90.93 (0.2)	0.86/0.90/0.91	0.73/0.96/0.98
	RoBERTa-L+Isotropy	91.42 (0.8)	91.17 (0.3)	0.18/0.36/0.43	0.61/0.96/0.98

Table 3: Results on the dev sets of selected GLUE tasks after running 5 times with different random seeds. For the performance measures, we report the median and standard deviation over the 5 models. As for the isotropy measure, we report the explained variance of the model that exhibits median EV1. Results from IsoBN have been extracted from the work done by Zhou et al. (2021).

4.4 DISCUSSION

BERT vs RoBERTa: We notice that the improvements on BERT were higher than the improvements in RoBERTa in terms of performance and isotropy. Investigating this phenomenon is left for future work. However, we provide some hypotheses that could explain the observed behavior. One source of this discrepancy can be attributed to the highly anisotropic nature of the RoBERTa embedding space; Figure 2 shows that most of the variance is concentrated in the top 5 principal components (unlike BERT, where the variance is more distributed among the PCs). Furthermore, RoBERTa is more complex than BERT (larger architecture). Therefore, a solution worth exploring is to increase the complexity of the Isotropy Layer (i.e., replacing the one feed-forward neural network with a deeper network that can learn a more complex transformation resulting in better embeddings). Another source of the discrepancy observed could be due to the strict constraint imposed by the downstream task layers (that are frozen). In other terms, the downstream-task layers might rely heavily on the information encoded in the top principal components. A solution worth exploring in future work is to retrain the Isotropy Layer on the improved embeddings, fine-tune the downstream task layers on these transformed embeddings, and repeat both steps until convergence. Another potential solution is to jointly train both the Isotropy Layer and the downstream-task layers with the joint loss function.

IsoLayer vs IsoBN: We pinpoint an interesting analogy between both approaches; the IsoBN approach employs an isotropic batch normalization to regularize the embeddings, while our method learns a transformation that adds an isotropic penalty term to regularize the embeddings. We should note that the IsoLayer method trains only one feed-forward neural network, while the IsoBN method performs the training for the whole network. A disadvantage of our method is that the performance is highly constrained by the downstream task layers. Perhaps, a more isotropic embedding space with better semantic properties can be reached with a different downstream task network.

5 CONCLUSIONS AND FUTURE WORK

As mentioned in the previous sections, BERT-based models suffer from the problem of having an anisotropic embedding space. This affects the representation capacity of the embedding space and affects the accuracy of the downstream tasks. In our work, we proposed to learn an embedding transformation that improves the isotropy of the [CLS] embeddings by adding an Isotropy Layer at the output of the fine-tuned language model and only training this layer using a joint loss function. Once trained, the layer will output transformed embeddings of better statistical properties that result in a better performance on the downstream task. We applied empirical methods to quantitatively measure the improvement of the models in terms of isotropy and performance on the downstream task. The experimental results on 3 GLUE datasets showed that our proposed method is on par with the state-of-the-art, as it achieves performance gains of around 2-3% on the downstream tasks compared to the baseline. A promising direction would be to understand the impact of our solution on the semantics of the model. To do so, we propose to employ tools that allow us to navigate the embedding space, giving us insights into the distribution of the concepts in the embedding space (i.e., reach more interpretable results). Since this property is not supported by default, we leave this direction for future work.

REPRODUCIBILITY STATEMENT

As mentioned in the previous sections, all datasets used in this study are part of the GLUE benchmark (https://gluebenchmark.com/). Furthermore, all seeds have been fixed to ensure that the results are reproducible. More information regarding the details of the training process (including hyperparameter tuning) is available in a separate technical report (reference to which is not disclosed due to the double-blind review). The source code of the model is available on GitHub at (URL is not disclosed due to the double-blind review).

REFERENCES

Daniel Biś, Maksim Podkorytov, and Xiuwen Liu. Too much in common: Shifting of embeddings in transformer language models and its implications. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 5117–5130, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.403. URL https://aclanthology.org/2021.naacl-main.403.

Kawin Ethayarajh. How contextual are contextualized word representations? comparing the geometry of bert, elmo, and GPT-2 embeddings. volume abs/1909.00512, 2019. URL http://arxiv.org/abs/1909.00512.

Jun Gao, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tieyan Liu. Representation degeneration problem in training natural language generation models. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SkEYojRqtm.

Chengyue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. Frage: Frequency-agnostic word representation. *ArXiv*, abs/1809.06858, 2018.

Katikapalli Subramanyam Kalyan, Ajit Rajasekharan, and Sivanesan Sangeetha. Ammus : A survey of transformer-based pretrained models in natural language processing, 2021. URL https://arxiv.org/abs/2108.05542.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9119–9130, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.733. URL https://aclanthology.org/2020.emnlp-main.733.

Yuxin Liang, Rui Cao, Jie Zheng, Jie Ren, and Ling Gao. Learning to remove: Towards isotropic pre-trained bert embedding. In *Artificial Neural Networks and Machine Learning – ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part V*, pp. 448–459, Berlin, Heidelberg, 2021. Springer-Verlag. ISBN 978-3-030-86382-1. doi: 10.1007/978-3-030-86383-8_36. URL https://doi.org/10.1007/978-3-030-86383-8_36.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2017. URL https://arxiv.org/abs/1711.05101.

Jiaqi Mu and Pramod Viswanath. All-but-the-top: Simple and effective post-processing for word representations. 2018. Publisher Copyright: © Learning Representations, ICLR 2018 - Conference Track Proceedings.All right reserved.; 6th International Conference on Learning Representations, ICLR 2018 ; Conference date: 30-04-2018 Through 03-05-2018.

S. Rajaee and Mohammad Taher Pilehvar. A cluster-based approach for improving isotropy in contextual embedding space. In *ACL*, 2021a.

S. Rajaee and Mohammad Taher Pilehvar. How does fine-tuning affect the geometry of embedding space: A case study on isotropy. In *EMNLP*, 2021b.

William Rudman, Nate Gillman, Taylor Rayne, and Carsten Eickhoff. IsoScore: Measuring the uniformity of embedding space utilization. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 3325–3339, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.262. URL https://aclanthology.org/2022.findings-acl.262.

William Timkey and Marten van Schijndel. All bark and no bite: Rogue dimensions in transformer language models obscure representational quality. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 4527–4546, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.372. URL https://aclanthology.org/2021.emnlp-main.372.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.

Lingxiao Wang, Jing Huang, Kevin Huang, Ziniu Hu, Guangtao Wang, and Quanquan Gu. Improving neural language generation with spectrum control. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=ByxY8CNtvr.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2019. URL https://arxiv.org/abs/1910.03771.

Davood Zabihzadeh. Ensemble of loss functions to improve generalizability of deep metric learning methods. *ArXiv*, abs/2107.01130, 2021.

Zhong Zhang, Chongming Gao, Cong Xu, Rui Miao, Qinli Yang, and Junming Shao. Revisiting representation degeneration problem in language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pp. 518–527, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.46. URL https://aclanthology.org/2020.findings-emnlp.46.

Wenxuan Zhou, Bill Yuchen Lin, and Xiang Ren. Isobn: Fine-tuning bert with isotropic batch normalization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(16):14621–14629, May 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17718.