Centrum für Informations- und Sprachverarbeitung

Ludwig-Maximilians-Universität München

Masterarbeit

im Studiengang Computerlinguistik
Fakultät für Sprach- und Literaturwissenschaften

Rogue Dimensions in Multilingual Embeddings

The Influence of Outliers on Similarity Search

Alina Fastowski

Prüfer:	Prof. Dr. Alexander Fraser
Betreuer:	Katharina Hämmerl
Bearbeitungszeitraum:	21. März - 08. August 2022

Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig angefertigt, alle Zitate als solche kenntlich gemacht sowie alle benutzten Quellen und Hilfsmittel angegeben habe.

München, den 08. August 2022

.....
Alina Fastowski

Contents

Chapter 1

Introduction

Transformer-based language models have gained large popularity for solving numerous NLP tasks over the last years. Despite reaching high performances, such models are reported to exhibit *rogue dimensions* in the embeddings produced by them, i.e. dimensions whose values significantly deviate from the norm (Kovaleva et al. [2021], Timkey and van Schijndel [2021]).

The available research on this topic is mainly concerned with monolingual models, testing the effect of such outliers on tasks mostly requiring fine-tuning (see chapter 3). There is however a lack of discussed multilingual models, as well as the effects on tasks directly using such rogue textual representations. This thesis aims to close this gap and perform research into this direction.

1.1 Aims of this work

This thesis investigates the multilingual XLM-R model [Conneau et al., 2020] on the topic of rogue embedding dimensions. In order to specifically study the effects of such dimensions on model performance, the general task of textual similarity search was chosen. Because in such problems, the embeddings a model produces are directly used for the task, similarity search was considered especially suitable for studying the immediate effects of embedding outliers.

The following work aims to first identify the dimensions which are rogue for the XLM-R model, after which they are systematically removed from the sentence embeddings used for the chosen Tatoeba similarity search task [Artetxe and Schwenk, 2019] to study their effects on performance. Observing an increase in accuracy when discarding such dimensions, it is then searched for other outliers able to improve similarity search performance. During this process, new outlier dimensions are discovered, whose removal is likewise beneficial for the task, but are not defined by magnitude as their main criterion.

1.2 Outline

After defining central topics essential for the further understanding of this thesis in chapter 2, chapter 3 presents the previously done research on the topic of rogue dimensions in language models. Chapter 4 then begins the investigations of outliers in the XLM-R model, focusing on the task of similarity search. After gaining first insights, chapter 5 analyzes the findings in more depth. Chapter 6 then returns to the investigations, viewing rogue dimensions from a different perspective and asking the question, what truly characterizes an outlier. The discussion of the thesis is presented in chapter 7, which collects the main findings of this work as well as possible future research directions.

Chapter 2

Multilingual Language Models

As this thesis is concerned with multilingual embeddings, this chapter lays out the key points associated with the topic for better further understanding. In the following, the subject of word embeddings is reviewed through its most central aspects, after which the Transformer architecture, which is the building block for the models discussed throughout this work, is outlined. Combining the presented information, embeddings are put into the context of neural network architectures.

2.1 Embeddings

Embeddings are representations of words in the form of mathematical vectors, each vector having a certain number of dimensions. Each such dimension can be thought of as encoding a certain e.g. semantic or syntactic feature, allowing words to be represented in a shared vector space, placing words of similar meanings close to each other.

2.1.1 Static vs. Contextualized

On the topic of word embeddings it needs to first be clearly distinguished between static and contextualized representations.
Static embeddings assign one single meaning, i.e. one single static vector, to each word. It therefore does not matter which context it occurs in and whether that may give it an inherently different meaning. Examples for such are Word2Vec [Mikolov et al., 2013] and GloVe [Pennington et al., 2014].
Contextualized embeddings on the other hand assign context-sensitive representations: depending on other words that surround a given word in e.g. an input sentence, a certain embedding is generated. This is the case in state-of-the-art language models such as e.g. BERT [Devlin et al., 2019], ELMo [Peters et al., 2018], the GPT models [Radford and Narasimhan, 2018], and XLM-R [Conneau et al., 2020] among others.

2.1.2 Mono- vs. Multilingual

As current language models largely employ contextualized representations over static ones, there is another difference that exists among them: they can be mono- or multilingual.
The difference between them majorly lies in the pre-training data they are exposed to: while e.g. the original BERT is pre-trained using only English textual data[1], the

[1] 800M words from the BookCorpus [Zhu et al., 2015] and 2,500M words from the English Wikipedia. [Devlin et al., 2019, p. 5]

multilingual BERT instead sees 104 different languages in this process. The same applies for the model investigated in this work, XLM-R[2].

Because word embeddings capture the *meaning* of a word, multilingual representations become comparable regardless of the language, since e.g. the meaning of the English *cat* is the same as the German *Katze*. In the shared vector space, they therefore end up close to each other.

Moreover it is important to note that, when passing textual input to a multilingual model, it does not explicitly distinguish the language of the text by a marker of some kind. Likewise there neither is any special mechanism *enforcing* translations of words in different languages to have similar embeddings [Pires et al., 2019, p. 4997].

2.2 Model architecture

Having defined the embeddings investigated in this work, which are contextualized and multilingual, this section outlines the underlying architecture of the language models which produce them.

2.2.1 The Transformer

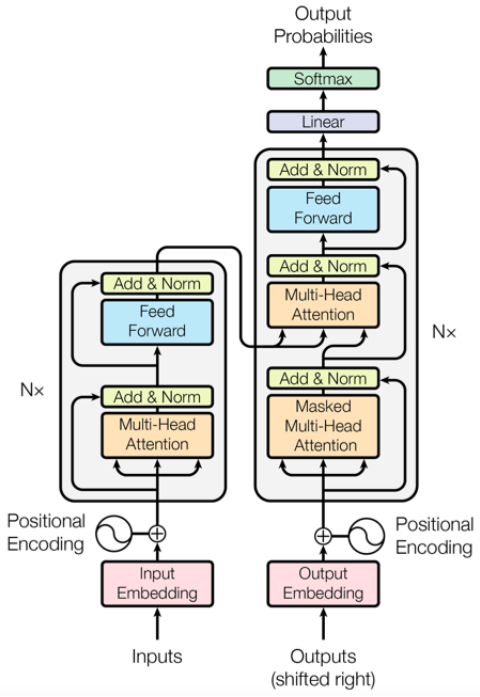


Figure 2.1: The Transformer architecture [Vaswani et al., 2017].

The Transformer, as first introduced by Vaswani et al. [2017], is a neural network architecture following an encoder-decoder scheme, as depicted on the left- and right-hand side of Figure 2.1, respectively. The key functionality of this architecture is the *attention mechanism*, which "draws global dependencies between input and output" [Vaswani et al., 2017, p. 2] and by that makes it possible to omit convolutions and recurrence. The Transformer is also auto-regressive, meaning that for each current timestep, the output of the respective previous one is additionally consumed.

[2]Further model and pre-training description see section 4.1.

Looking at the architecture in more detail, both encoder and decoder consist of N=6 identical layers. In the encoder, each such layer includes a multi-head self-attention mechanism as well as a simple feed-forward layer. Both of these components are followed by layer normalization, which is also called the *LayerNorm*.

The decoder is different in the way that there is an additional attention component processing the previous output embeddings first, before its result is passed on further where the current output of the encoder is infused. Both embeddings passed into the encoder as well as to the decoder are also coupled to positional information, which is done by adding *positional encodings*. [Vaswani et al., 2017, p. 2f]

In recent years, most well-known language models were based on the Transformer. Important to note however is that not the entire architecture needs to be used: for example, GPT uses only the decoder part, while BERT and XLM-R use the encoder.

2.2.2 Neural nets and embeddings

Combining the above information, the big picture of generating word embeddings, as is being done for the entirety of experiments for this work, is presented.

Since the aforementioned models are deep neural networks, they follow the scheme of exhibiting a certain amount of model layers, each layer producing certain *hidden states*.

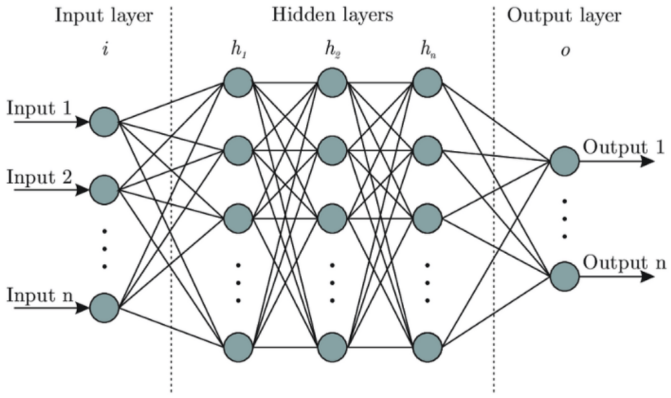


Figure 2.2: A simple neural network. [Riggs and Williams, 2020]

After a language model is pre-trained on (multi-)lingual data, using e.g. the Transformer encoder in the case of XLM-R, it is able to encode given words as contextualized embeddings. The process of obtaining them can be described as follows: passing the model textual input, the words are processed by each hidden layer. The hidden states in the hidden layers encode *features* of the words. Hence, in the following, when referring to a model's *embeddings*, it is referred to the *hidden states* of the model layer in question.

Chapter 3

Related Work

It is a well established fact that Language Models based on the Transformer architecture [Vaswani et al., 2017] tend to develop one or multiple outlier dimensions (Kovaleva et al. [2021], Timkey and van Schijndel [2021]). The following discusses for which models exactly such a phenomenon has been detected and how, which side effects such outliers may additionally cause, as well as possible ways to deal with these rogue dimensions. Lastly, some of the current conjectures as to what may be causing outliers in the first place are laid out.

3.1 Outliers in Language Models

Instead of directly examining the embeddings generated by a model, the search for outliers can be started earlier in the architecture. For instance, Kovaleva et al. [2021] have identified such phenomenon in the output LayerNorm components of Transformer-encoder models (see also section 2.2.1). In the following, findings of both monolingual and multilingual language models are outlined.

3.1.1 Monolingual Language Models

Considering mainly models from the BERT family (its base, medium and large versions, as well as RoBERTa [Liu et al., 2019]) [Devlin et al., 2019], Kovaleva et al. [2021] establish abnormally high values in the respective LayerNorms' scaling factors and biases.
To locate outlier dimensions, the authors compute the mean and the standard deviation σ of these weights. Dimensions, in which both of the weights deviate 3σ or more from the respective mean[1] are considered outliers.

As an example, this method detects the dimensions `308` and `381` as outliers for BERT-base. Since the embeddings generated by the model are directly influenced by the output LayerNorm, the effects of its outliers are likewise directly visible in the embeddings.
As can be seen in Figure 3.1, only dimension `308` is a consistent outlier across all model layers, whereas `381` is most prominent in layers 7-10. The relationship between the magnitude of LayerNorm weights and the resulting embeddings shall be further discussed in section 3.4.

Additionally to the models mentioned above, Kovaleva et al. [2021] also consider other Transformer-based architectures like BART [Lewis et al., 2020], ELECTRA [Clark et al., 2020], XLNet [Yang et al., 2019b] and GPT-2 [Radford et al., 2019].

[1] In the case of RoBERTa, 2σ is chosen as a boundary.

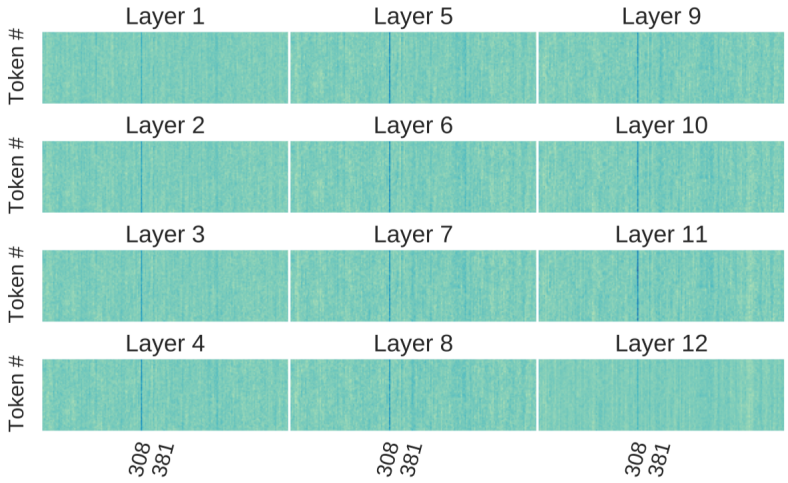


Figure 3.1: Heatmap visualization of outliers 308 and 381 in the embeddings of BERT-base [Kovaleva et al., 2021]. Generated with randomly sampled text from WikiText [Merity et al., 2016].

Employing the same method of locating outliers in the output LayerNorm[2], the authors report the same outlier phenomenon being observed in these models, but the outliers themselves aren't specified further.

The results of Kovaleva et al. [2021]'s investigations concerning BERT and RoBERTa are likewise supported by Puccetti et al. [2022], who replicate identical results using the same technique for locating rogue dimensions as described above. For BERT-base, again the dimensions 308 and 381 are reported as outliers, while for RoBERTa-base they establish the dimensions 77 and 588.

3.1.2 Multilingual Language Models

While the aforementioned models were exclusively monolingual (English) ones, Rajaee and Pilehvar [2022] turn their focus to the comparison of BERT with its multilingual version, mBERT [Devlin et al., 2019]. To identify outliers in these models' embeddings, the same procedure as proposed by Kovaleva et al. [2021] by calculating the mean and standard deviation is followed, however in this case the authors do not apply this method to features of the LayerNorms, but instead directly to the embeddings, i.e. the mean and standard deviation are calculated over the dimensions of the embeddings, making each dimension differing from the mean by at least 3σ a rogue one.

Using articles from Wikipedia as textual input, the authors choose the languages English, Spanish, Arabic, Turkish, Sundanese and Swahili to assess mBERT. Naturally, English input is chosen for monolingual BERT. The representations are obtained from the respective last layer of the models. Figure 3.2 depicts the average representations reported by the authors.

While not providing exact dimensions, the outlier in the monolingual BERT embeddings appears to be dimension 308, which would coincide with the findings of Kovaleva et al. [2021] and specifically Figure 3.1, which also shows that the embeddings of the last layer, which are used here, really only feature the outlier 308.

[2]For GPT-2, the dense output layer is considered instead, since in its architecture this is the last component before the final output, instead of the LayerNorm [Kovaleva et al., 2021].

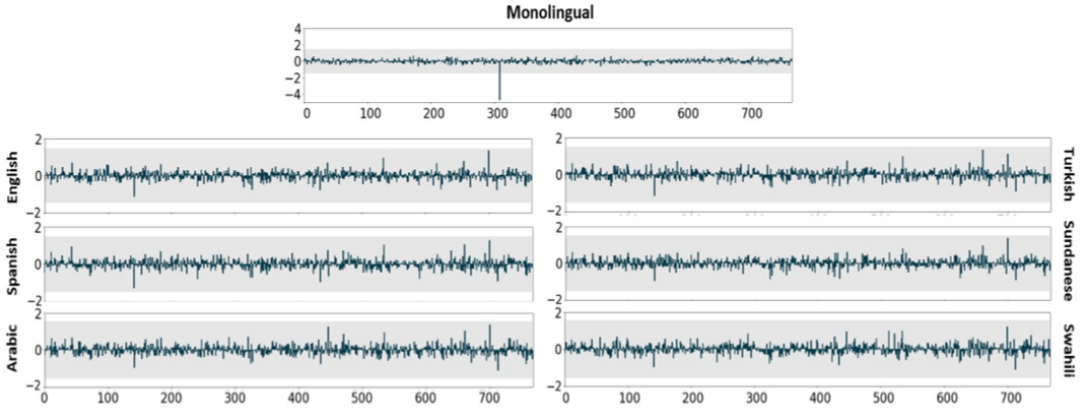


Figure 3.2: Average embeddings of BERT and mBERT. The grey area marks the space of 3σ [Rajaee and Pilehvar, 2022].

Most interestingly however, the multilingual counterpart mBERT does not produce any rogue dimensions at all, which is in contrast to any other previously considered models.
It may appear that the phenomenon of rogue dimensions is exclusive to monolingual models, however this is not true as further proven by Rajaee and Pilehvar [2022], who additionally perform the same experiments with another multilingual model, XLM-R [Conneau et al., 2020]. While this model is not discussed in detail, the key finding that it, in contrast to mBERT, *does* exhibit outlier dimensions in its embeddings, is still made.

This section illustrated the observations of outliers in Transformer-based Language Models that have been discovered up until now. However, the presence of outliers in these models' embedding spaces is often discussed alongside the phenomenon of *anisotropy*, which additionally needs to be addressed in this context.

3.2 (An)Isotropy

3.2.1 Definition

A central feature of the embedding spaces, which is strongly associated with the presence of outlier dimensions, is anisotropy, which is the opposite of isotropy. This concept is built around the definition of cosine similarity, which measures the angle between two vectors. For two vectors of dimension d, the angle between them, or their similarity, is measured as follows:

$$\cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|} = \frac{\sum_{i=1}^{d} \mathbf{u}_i \mathbf{v}_i}{\sqrt{\sum_{i=1}^{d} (\mathbf{u}_i)^2} \sqrt{\sum_{i=1}^{d} (\mathbf{v}_i)^2}} \tag{3.1}$$

Equation (3.1) produces a value between 0 and 1, where a higher value indicates higher similarity between the two vectors.

Considering an embedding space, where words from a text corpus are encoded as such, the concept of isotropy describes how similar these embeddings are to each other on average:

> "If word representations [...] were isotropic (i.e., directionally uniform), then the average cosine similarity between uniformly randomly sampled words would be 0. The closer this average is to 1, the more anisotropic the representations" [Ethayarajh, 2019, p. 58].

Following this definition, an anisotropic space would indicate that randomly selected word representations are highly similar to each other, which is naturally an undesired property since it defies a major purpose of word embeddings themselves, making them indistinctive from one another – regardless of whether they describe similar concepts or completely opposite ones.

3.2.2 Anisotropy in Contextual Embeddings

Probing Language Models on the subject of anisotropy in their embeddings, Ethayarajh [2019] consider BERT, ELMo [Peters et al., 2018] and GPT-2 in their experiments, in which they analyze the embeddings produced by each layer. Apart from the hidden layers (2 in ELMo, 12 in BERT and GPT-2, respectively) the authors additionally include the 0^{th} layers, which are the input layers. These are not yet contextualized. [Ethayarajh, 2019, p. 57]

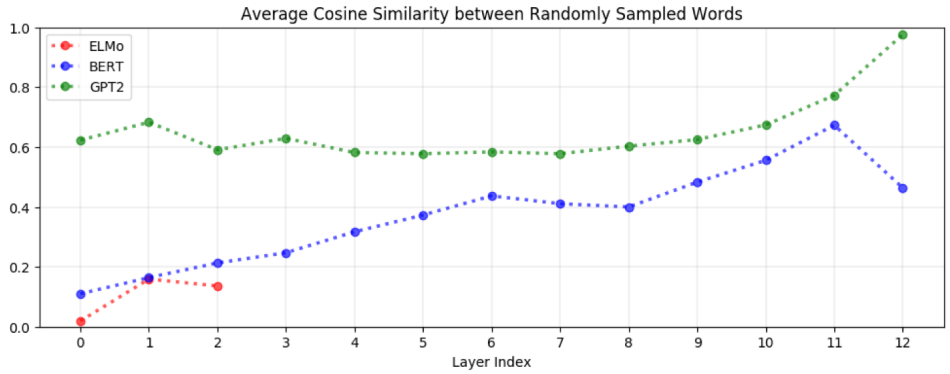


Figure 3.3: Average cosine similarity between randomly sampled words for all layers of BERT, ELMo and GPT-2 [Ethayarajh, 2019].

As can be concluded from Figure 3.3, all three models experience increasing anisotropy the further the number of layers progresses. With the 0^{th} layers being non-contextualized, it can be assumed that "high anisotropy is inherent to, or least a by-product of, the process of contextualization" [Ethayarajh, 2019, p. 59].
Clearly, the observed development of anisotropy is not exactly a linear progression: for GPT-2, there are almost no changes until layer 9, and for BERT the second-to-last layer demonstrates significantly higher anisotropy than the last layer. However, the trend is still observable.
In either of the cases, the embeddings produced by the last layers are remarkably more anisotropic than in the first ones, reaching around 0.65 at their peaks for BERT, and even 1.0 in the last layer of GPT-2, making these embeddings "so anisotropic that any two words have on average an almost perfect cosine similarity" [Ethayarajh, 2019, p. 58].

3.2.3 The Role of Rogue Dimensions

After it has been shown by Ethayarajh [2019] that the embedding spaces produced by contextualized language models like BERT and GPT-2 feature the phenomenon of anisotropy in the first place, Timkey and van Schijndel [2021] perform an analysis of what the cause of this may be.
They come to the conclusion that "the cosine similarity of any two tokens is *dominated* by a small subset of *rogue dimensions*" [Timkey and van Schijndel, 2021, p. 4528], which is the reason why this topic is discussed in this work.

Employing BERT, RoBERTa, GPT-2 and XLNet as contextual models, the authors select Word2Vec [Mikolov et al., 2013] and GloVe [Pennington et al., 2014] as static ones for comparison. As for textual input, they construct a corpus of 85k random tokens from the English Wikipedia.
Timkey and van Schijndel [2021]'s experiments have the purpose of identifying specific embedding dimensions that contribute to the overall anisotropy in a model's embedding space. In the process, they additionally calculate the proportion that each of those dimensions contributed to the overall anisotropy. This is achieved as follows:

Let the anisotropy in layer ℓ of model f be defined as the expected cosine similarity between any two word representations. In the following, this is denoted as $\hat{A}(f_\ell)$. This value can be estimated from a sample S of random pairs of tokens $\{\{x_1, y_1\}, ..., \{x_n, y_n\}\}$ of the chosen corpus. [Timkey and van Schijndel, 2021, p. 4528]
Let the cosine similarity between two embedding vectors u and v be defined as in Equation (3.1). The function $CC_i(u, v)$ then expresses the contribution of a dimension i to the overall cosine similarity between u and v.

$$CC_i(u, v) = \frac{u_i v_i}{\|u\| \|v\|} \tag{3.2}$$

$CC(f_\ell^i)$ then defines the contribution of dimension i to the expected anisotropy in layer ℓ.

$$CC(f_\ell^i) = \frac{1}{n} \cdot \sum_{\{x_\alpha, y_\alpha\} \in S} CC_i(f_\ell(x_\alpha), f_\ell(y_\alpha)) \tag{3.3}$$

It then holds that $\hat{A}(f_\ell) = \sum_i^d CC(f_\ell^i)$, d being the total number of dimensions.

[Timkey and van Schijndel, 2021, p. 4529]

Sampling 500k random token pairs from their constructed Wikipedia corpus, Timkey and van Schijndel [2021] perform the described analysis on the six chosen models. Reported are the top-3 dimensions contributing to anisotropy, each contribution $CC(f_\ell^i)$ normalized by $\hat{A}(f_\ell)$, the total expected anisotropy.

Model	Layer	1	2	3	$\hat{A}(f_\ell)$
GPT-2	11	0.275	0.269	0.265	0.640
	12	0.763	0.131	0.078	0.885
BERT	10	0.817	0.004	0.003	0.396
	11	0.884	0.003	0.002	0.506
RoBERTa	7	0.726	0.193	0.032	0.705
	12	0.663	0.262	0.020	0.745
XLNet	10	0.990	0.000	0.000	0.887
	11	0.996	0.001	0.000	0.981
Word2Vec		0.031	0.023	0.023	0.130
GloVe		0.105	0.096	0.095	0.104

Figure 3.4: Proportion of total expected cosine similarity, $CC(f_\ell^i)/\hat{A}(f_\ell)$, contributed by each of the top 3 dimensions in the two most anisotropic layers of each model [Timkey and van Schijndel, 2021, p. 4529].

It can for one be observed that the static models Word2Vec and GloVe produce rather small values for $\hat{A}(f_\ell)$, making them more isotropic. E.g. for Word2Vec the result would mean that any two randomly picked tokens have a cosine similarity of around 0.13. The rest of the models however, which are contextual, all have (at least!) two layers that produce highly anisotropic word representations. As the authors themselves circle out, XLNet produces the most extreme result, in which for layers 10 and 11 the expected cosine similarity is around 0.89 and 0.98 respectively, and for each of these layers, a single dimension contributes over 99% to the anisotropy [Timkey and van Schijndel, 2021, p. 4529].
Looking into the identified contributing dimensions further, the authors find that these are indeed outlier dimensions present in the embedding spaces: "**The dimensions which drive anisotropy are centered far from the origin relative to other dimensions**" [Timkey and van Schijndel, 2021, p. 4529].

This finding is likewise supported by parallel research: for instance, in their investigations of the rogue dimensions in BERT and RoBERTa, Luo et al. [2021] find that by clipping the outliers, i.e. zeroing out the respective dimensions in the embedding vectors, the embedding spaces become significantly less anisotropic: if the expected cosine similarity in RoBERTa's last layer was originally around 0.8, after removing the outliers it drops to about 0.15 [Luo et al., 2021, p. 5317, see Fig. 6].

3.3 Removing Rogue Dimensions

3.3.1 Methods

The following describes the two most straightforward ways that can be used to make adjustments to rogue dimensions.

Clipping As mentioned above, Luo et al. [2021] use "clipping", i.e. zeroing out the respective embedding dimensions, as a method to deal with outliers. The same method is likewise applied by Kovaleva et al. [2021], who however apply the clipping not directly to the word representations, but the identified outlier weights in the LayerNorm component (see also section 3.1.1).
The approach of clipping targets the rogue dimensions only, without modifying the rest of the respective word representation.

Standardization Ethayarajh [2019] propose the method of standardization as a post-processing step to account for rogue dimensions. For a corpus $\mathcal{O}$ with its word representations $x \in \mathbb{R}^d$, the mean vector $\mu \in \mathbb{R}^d$ and the standard deviation for each dimension $\sigma \in \mathbb{R}^d$ would be computed as follows:

$$\mu = \frac{1}{|\mathcal{O}|} \cdot \sum_{x \in \mathcal{O}} x \tag{3.4}$$

$$\sigma = \sqrt{\frac{1}{|\mathcal{O}|} \cdot \sum_{x \in \mathcal{O}} (x - \mu)^2} \tag{3.5}$$

[Ethayarajh, 2019, p. 4533]

Applying standardization, the mean would be subtracted from each vector, and the result divided by the standard deviation, i.e. each vector changes to $\frac{x-\mu}{\sigma}$ [Ethayarajh, 2019].
This method however modifies all dimensions of the word representations, not only the rogue ones.

3.3.2 Effects on Model Performance

As established in section 3.2.3, one of the effects of rogue dimensions is the increased anisotropy in the embedding spaces, and consequently, their removal makes the spaces more isotropic, which is desirable. It is however also interesting to consider what happens to model performance on downstream tasks, when these dimensions are removed.

Kovaleva et al. [2021] investigate this by evaluating the BERT-base model, in which they have identified the outlier dimensions `308` and `381` (see section 3.1.1), on tasks from the GLUE benchmark [Wang et al., 2018]. The nine tasks focus on single-sentence tasks (for sentence acceptability and sentiment), detection of semantic similarity and Natural Language Inference [Wang et al., 2018, p. 353].

	MRPC	STS-B	MNLI	MNLI-mm	COLA	SST-2	QQP	QNLI	RTE
Baseline (full model)	87.2	88.8	84.1	84.2	56.8	92.5	89.8	90.6	61.7
Non-outlier†	+0.3	-0.1	-0.2	-0.1	+0.2	0	-0.1	0	-0.4
Outlier-`308`	-10.5	-23.4	-2.2	-1.8	-2.16	-0.6	-1.0	-1.9	-7.2
Outlier-`381`	-4.6	-4.4	-13.7	-13.0	-22.2	-3.4	-10.8	-7.3	-5.0
Random non-outlier pair‡	-1.1	0.0	+0.3	+0.2	-0.5	+0.1	+0.1	0	+0.5
Outliers `308` + `381`	**-8.6**	**-44.1**	**-27.9**	**-27.2**	**-32.3**	**-20.8**	**-13.0**	**-12.2**	**-10.0**

Figure 3.5: Performance of BERT-base on GLUE tasks with disabled outliers. †For each of the non-outlier dimensions, their parameters are disabled one at a time. It is then averaged over them. ‡For pairs of non-outlier dimensions, averages over 1000 runs are reported. [Kovaleva et al., 2021, p. 3397]

Kovaleva et al. [2021] evaluate the model performance using the full model (i.e. without modifying anything, here: "Baseline"), and compare it with its performances when different outliers are disabled. The two outliers `308` and `381` are disabled both one at a time, as well as together. For additional comparison, non-outlier dimensions are also tested.
As can be seen in Figure 3.5, the biggest drop in performance occurs when both outliers are clipped, while disabling non-outlier dimensions does not have much of

an effect altogether. Which single outlier causes the most damage is however not universal across all tasks: for example, disabling `308` causes a big drop for STS-B (-23.4), but a comparatively small one for CoLA (-2.16). For the outlier `381` it is instead the opposite: its removal damages the performance on STS-B by only -4.4, but CoLA is more significantly affected with a drop of -22.2 [Kovaleva et al., 2021, p. 3397].
A similar drop in performance for GLUE tasks when the model's outliers are removed is reported for RoBERTa by [Puccetti et al., 2022, Appendix A].

Concluding this section, it can be established that disabling rogue dimensions can have both positive and negative effects. On the one hand, their removal improves the problem of anisotropy in the embedding spaces of language models, making them more isotropic. On the other hand however, disabling them seems to impair model performance on downstream tasks. Therefore one will have to decide based on the use case whether to disable rogue dimensions or not.
Another possibility would be to find a way to pre-train the mentioned models such that outlier dimensions don't develop in the first place, however therefore the root cause for their emergence needs to be known. The current theories on this matter are discussed in the next section.

3.4 Possible causes for outliers

Outlier dimensions becoming apparent in the embeddings of language models is merely a consequence of some other cause that is either rooted somewhere earlier in the model, or even outside of it, e.g. in the data it has seen during pre-training. Therefore, in order to find the reason for the emergence of rogue embedding dimensions, the search should be started in earlier stages.

3.4.1 LayerNorm weights

Being the immediate component to output the embeddings in Transformer-encoder models, it is logical to start the search there. As already discussed in section 3.1.1, Kovaleva et al. [2021] find outliers in models' LayerNorm scaling factors and biases. Because the output-LayerNorm is directly connected to the resulting embeddings, its outliers are consequently projected onto the word representations (see also Figure 3.1).
In a further experiment, Kovaleva et al. [2021] pre-train BERT-medium themselves on the BookCorpus [Zhu et al., 2015] and monitor the development of the LayerNorm scaling factors and biases. They find that "both [...] begin to diverge from their initialization values quite early [...] in the training process. At roughly the same point, both training loss and evaluation perplexity begin to fall off" [Kovaleva et al., 2021, p. 3399]. This is illustrated by the results in Figure 3.6, which feature the outlier dimension `417` identified for BERT-medium.

An additional insight this result provides is that the development of the outlier seems to help the model, since training loss drastically decreases at around the same time the outlier emerges. This is also in line with the results discussed in section 3.3.2, which show how model performance drops when the outliers are clipped.

Therefore the outliers in the LayerNorm may mathematically be the cause for outliers in the resulting embeddings, however in this case something must in turn cause the LayerNorm outliers to emerge in the first place, as well. This consequently may not be the original root cause of the phenomenon.

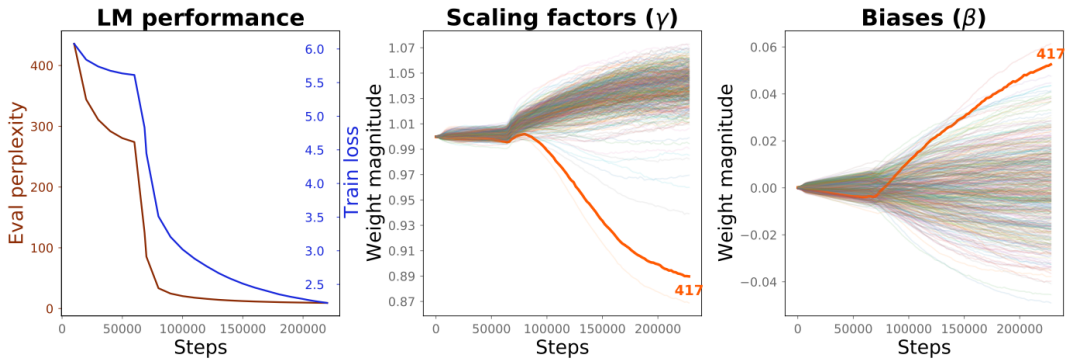


Figure 3.6: Monitoring of BERT-medium pre-training. Graphics show the evaluation perplexity, train loss, as well as both output-LayerNorm weights. The (orange) highlighted dimension `417` is an identified outlier outside the 3σ range for BERT-medium. [Kovaleva et al., 2021, p. 3400]

3.4.2 Positional Embeddings

Instead of the LayerNorm, Luo et al. [2021] have put their focus on the input layer of BERT and RoBERTa, which consists of token and segment embeddings, as well as positional embeddings [Luo et al., 2021, p. 5314]. Specifically the latter were found to feature outliers in the same dimensions as the resulting word representations.

For instance, the authors replicate the outlier dimensions `77` and `588` in RoBERTa's embeddings. Coincidentally, dimension `588` was found to be nearly always the maximum element of each positional embedding[3] [Luo et al., 2021, p. 5314].
In order to verify a causal relationship between outliers in the positional information and the final word embeddings, the authors pre-train RoBERTa themselves on the English Wikipedia Corpus, once including positional embeddings, and once discarding them.
They find that while the model pre-trained with positional encodings still produces the outlier phenomenon, the counterpart that doesn't include positional information does *not* exhibit rogue embedding dimensions. From this finding, Luo et al. [2021] conclude a direct relationship of such outliers to positional embeddings [Luo et al., 2021, p. 5316].

While this hypothesis is interesting and may be valid, there are a few points for critique. For one, Luo et al. [2021] prove this behaviour only for the RoBERTa-base model. It is hard to therefore conclude such causal relationship between positional information and outlier dimensions for other Transformer-based models, which were also proven to exhibit outliers (see e.g. section 3.1.1).
Secondly, this hypothesis is also questioned by Rajaee and Pilehvar [2022], who have shown how BERT exhibits outlier dimensions, while mBERT does not (see section 3.1.2). They therefore argue that Luo et al. [2021]'s hypothesis can not be valid, "given that both multi- and monolingual spaces are constructed using the same training procedure involving positional encodings" [Rajaee and Pilehvar, 2022, p. 4].

Although this theory may not be entirely proven to be true, it reports interesting findings and was therefore considered important mentioning here.

[3] Specifically, this is the case from the 4th positional embedding through the final one.

3.4.3 Token Frequency in Pre-Training Data

Taking a look outside model architectures, Puccetti et al. [2022] hypothesize pre-training data to be related to the emerging problem: "Since [Language Models] rely on statistical patterns of token co-occurrence, token frequency could be expected to affect the learned representations" [Puccetti et al., 2022, p. 4].

As a first check, the authors examine the perfomance of the Masked Language Modeling (MLM) task using BERT-base and RoBERTa-base. Using 10^4 sentences from Wikipedia and randomly masking 15% of tokens, they analyze whether the frequency distribution of tokens generated by the MLM changes, when outliers[4] are removed. In order to measure what a high- or low-frequency token is, they estimate[5] the models' pre-training data from a corpus similar to BERT's pre-training corpus.

They find that the models with removed outliers generate *more high-frequency tokens* than the non-modified models. This is the case for both BERT and RoBERTa. For BERT specifically they find that removing its outliers makes the model produce more punctuation, nouns, symbols and adpositions [Puccetti et al., 2022, p. 4].

Apart from examining generated tokens from the MLM, the authors additionally consider whether there may be a correlation between the frequency of *encoded* tokens and the outliers. In an experiment they establish "the correlation between [the outliers'] magnitude and encoded token frequency [to be] much higher than for random dimensions" [Puccetti et al., 2022, p. 4].

After finding a connection between token frequency and outliers, Puccetti et al. [2022] aim to establish a causal relationship. For this, the authors pre-train three BERT-medium versions, each with a different method of tokenization. Each is focused on a different application of the *[SEP]* token, which is employed for sentence separation in BERT's training data. While method (1) inserts the *[SEP]* token simply after each sentence, the other two disturb the linguistic structure by either (2) applying the special token after each 256 tokens rather than each sentence, reducing the overall amount of *[SEP]*s, or (3) replacing half of the high-frequency tokens with low-frequency ones.

While all three models develop outliers, they behave slightly differently when evaluated on the MNLI task [Williams et al., 2018a]. For the analysis, Puccetti et al. [2022] evaluate using the full models, as well as with removing their respective outliers. The most striking insight would have to be that for method (2), which features less special tokens, the performance drops significantly when one of its outliers is disabled. The authors conclude that "since high frequency is one the factors that characterizes special tokens, it must also contribute to their role in outliers" [Puccetti et al., 2022, p. 7].

As a possible solution to the problem of emerging outliers, which Puccetti et al. [2022] connect to the token frequency in the pre-training data, they propose finding a pre-training scheme which better accounts for token frequency [Puccetti et al., 2022, p. 8f].

[4] Analogous to previous research established to be `308` and `381` for BERT-base and `77` and `588` for RoBERTa. [Puccetti et al., 2022, p. 2]

[5] Their estimated corpus consists of the BookCorpus [Zhu et al., 2015] and a 2021 dump from Wikipedia. [Puccetti et al., 2022, p. 4]

3.4.4 Conclusion

The previously described insights discuss hypotheses that are both rooted inside the model architecture, i.e. connecting the outlier problem to LayerNorm weights and positional embeddings, and outside of it, finding the cause in the pre-training data. In fact, both types of factors may apply. Because Puccetti et al. [2022], who propose the pre-training data hypothesis, likewise find outliers in a non-Language Model, the Vision Transformer [Dosovitskiy et al., 2020], this may be a clue that the root of the problem may at least partially lie in the Transformer architecture itself.

Therefore, until further research is conducted to establish the definite cause of outliers, the current way to think about it would be to accept multiple factors as possible contributors to the problem.

Chapter 4

Exploring XLM-R

This chapter investigates the multilingual model XLM-R, as well as two modified versions of it, on the topic of rogue embedding dimensions. After defining their formalities, the Tatoeba similarity search task is introduced, which itself includes the dataset used for all following experiments.
Defining the methodology for obtaining and analyzing the embeddings, the received results on outlier dimensions in the models are presented. Employing the Tatoeba task, the models are then tested on their performance with regard to their outlier dimensions.

4.1 Models

4.1.1 XLM-R

XLM-RoBERTa (XLM-R) is a Transformer-based multilingual language model introduced by Conneau et al. [2020]. The model is influenced by both XLM [Lample and Conneau, 2019] and RoBERTa [Liu et al., 2019] in different ways.

XLM The acronym of *XLMs* denotes the concept of *cross-lingual language models*. In their paper, Lample and Conneau [2019] explore methods to pre-train XLMs by using three different pre-training objectives:

- Causal Language Modeling (CLM) trains a Transformer to model the probability of a word w_t in a sentence as $P(w_t|w_1, ..., w_{t-1}, \theta)$.
- Masked Language Modeling (MLM) is identical to the method introduced by Devlin et al. [2019], which in its essence replaces a fraction of tokens with an artificial [MASK] token, making the model predict the original token. The only difference is that Lample and Conneau [2019] use text streams of 256 tokens instead of sentences specifically.
- Translation Language Modeling (TLM), being an extension of MLM, applies the masking to concatenated parallel sentences instead of monolingual text, where words of both the source and target language sentence are masked. For prediction, the model can then attend to words of both languages' sentences.
  [Lample and Conneau, 2019, p. 3]

The goal of Lample and Conneau [2019] is specifically to pre-train models with the above objectives in a *cross-lingual* way. With CLM and MLM, this is achieved with monolingual data, making the methods unsupervised, whereas the supervised TLM requires parallel data. [Lample and Conneau, 2019, p. 2]

For their experiments, the authors consider models pretrained on CLM only, MLM only as well as MLM together with TLM, each relying on a 1024 hidden unit Transformer architecture with 8 heads. [Lample and Conneau, 2019, p. 5]

Evaluating the resulting models on cross-lingual classification, supervised and unsupervised machine translation, the authors find that their "fully unsupervised MLM method sets a new state of the art on zero-shot cross-lingual classification" [Lample and Conneau, 2019, p. 6], where using MLM together with TLM improves the result even more. Testing machine translation performance with the MLM- and CLM-only models, they are both observed to consistently outperform the previous states of the art on both supervised and unsupervised MT tasks, where the MLM produces the overall best results. [Lample and Conneau, 2019, p. 7]
The authors conclude CLM and MLM pre-training for cross-lingual language models to produce solid cross-lingual features. Additionally using parallel data, TLM is likewise concluded to be highly beneficial. [Lample and Conneau, 2019, p. 8]

Resulting from this research, there are currently multiple different versions of XLM available[1]: XLM-15, XLM-17 and XLM-100, the number indicating the number of languages the models include. It is noted that XLM-17 and XLM-100 are pre-trained on the MLM objective only. **XLM-R can be seen as a refinement of XLM-100**, but with a significant change, for which the inspiration is drawn from RoBERTa.

RoBERTa The "Robustly optimized BERT approach" RoBERTa is, as the name suggests, a version of BERT that is improved by a series of adjustments in the way the model is pre-trained. While these measures include removing the next sentence prediction objective and training the model longer, another significant change is the *increased amount of training data*. [Liu et al., 2019]

This entry point for model improvement is leveraged by XLM-R: in comparison to XLM-100, which it is in general based off of, XLM-R uses more and different pre-training data. While XLM-100 is trained on text from Wikipedia, Conneau et al. [2020] construct a new corpus of 2.5 TB of data for XLM-R, which is based on the CommonCrawl[2]. By relying on 12 dumps from the CommonCrawl for all languages except for English, the authors manage to majorly increase the size of the dataset in comparison to XLM-100. This change is particularly beneficial for low-resource languages. [Conneau et al., 2020, p. 3]

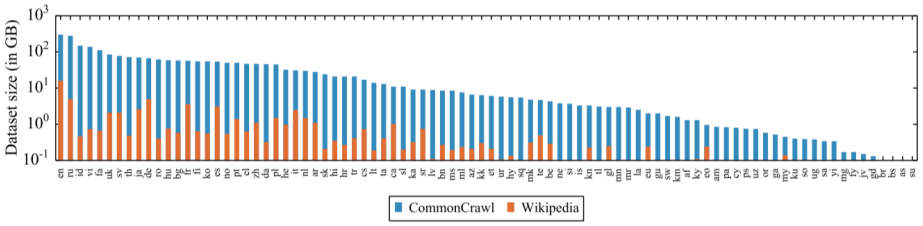


Figure 4.1: Amount of data in GB (log-scale) for the Wikipedia corpus used for XLM-100 vs. the CommonCrawl corpus of XLM-R. [Conneau et al., 2020, p. 3]

Concluding the above definitions, XLM-R is therefore a mixture of XLM and RoBERTa: following the architecture of XLM-100, Conneau et al. [2020] train a Transformer with MLM using monolingual data only, but use the RoBERTa approach for its improvement by infusing the model with a magnitude of more pre-training data.

[1] https://github.com/facebookresearch/XLM
[2] https://commoncrawl.org

The success of XLM-R is measured by its perfomance on various tasks, showing it to exceed on the multilingual field and even being competitive with monolingual models. Specifically, it outperforms mBERT on cross-lingual benchmarks like XNLI (+14.6% accuracy), MLQA (+13% F1 score) and NER (+2.4% F1 score). The biggest improvements are seen on low-resource languages, which is a significant achievement. [Conneau et al., 2020, p. 1]
Comparing XLM-R with monolingual models on the GLUE benchmark [Wang et al., 2018], the authors find that while it is slightly worse than RoBERTa and $\text{XLNet}_{\text{large}}$, it manages to outperform $\text{BERT}_{\text{large}}$ by an average of 1.6% [Conneau et al., 2020, p. 7].

The authors release two versions of the model, $\text{XLM-R}_{\text{base}}$ and XLM-R, which have the following features:

	layers	hidden units	attention heads	parameters
$\text{XLM-R}_{\text{base}}$	12	768	12	270M
XLM-R	24	1024	16	550M

In all experiments of this work, the smaller $\text{XLM-R}_{\text{base}}$ is used. For convenience, it is in the following referred to as XLM-R.

4.1.2 X2S-MSE and X2S-CCA

Additionally to XLM-R itself, two closely related models constructed by Hämmerl et al. [2022] are also included in the investigations on rogue embedding dimensions.

In order to improve on the quality of multilingual embeddings, the authors propose the possibility of combining the advantages of both static and contextual representations. For this, they first construct static embeddings that are multilingually aligned: using Gupta and Jaggi [2021]'s *X2Static*, they extract static embeddings from XLM-R for 40 languages. These are then multilingually aligned with the help of unsupervised dictionaries and VecMap [Artetxe et al., 2018], resulting in the embeddings *X2S-MA* (X2Static-Multilingually-Aligned) [Hämmerl et al., 2022, p. 3].

Through continued pre-training, two modified XLM-R models are created. For this, two loss alignment terms are considered, which are both employed by comparing the embeddings the original XLM-R produces with the ones constructed in X2S-MA.
One way to do this is is to apply mean squared error (MSE) between these representations. The second way is a correlation loss called "deep canonical correlation analysis" (DCCA) [Andrew et al., 2013], which is based off of standard CCA [Hotelling, 1936]. This method aims to linearly transform two representations such as to maximize their correlation. DCCA replaces the linear transformations with deep networks: Hämmerl et al. [2022] treat as the two deep networks the contextualized XLM-R model and the static X2S-MA embeddings respectively. [Hämmerl et al., 2022, p. 4]

This procedure results in two modified XLM-R models, each extended with $+\text{X2S-MA}_{\text{MSE}}$ or $+\text{X2S-MA}_{\text{DCCA}}$ of continued pre-training. For reasons of convenience, the resulting extended models are shortened to **X2S-MSE** and **X2S-CCA** here.

4.2 Tatoeba

For an analysis of the models' embeddings with a focus on rogue dimensions, the Tatoeba similarity search task [Artetxe and Schwenk, 2019] was considered especially suitable for experiments, since it is based directly on the embeddings a model produces for the data. Tatoeba is composed of both a task and a dataset, which are both introduced in the following.

4.2.1 Dataset

The Tatoeba dataset is derived from an open-source collection[3] of high quality translated sentence pairs from English into more than 300 other languages, where the number of available sentences varies per language.
As a pre-processing step for constructing their dataset from the collection, Artetxe and Schwenk [2019] remove symbols that aren't specific to any language, as well as sentences shorter than three words or duplicates [Artetxe and Schwenk, 2019, p. 13].
The resulting dataset consists of up to 1000 English-aligned sentences per language, which is available for 72 languages. The range can be increased to 86 languages by limiting the available sentences to 500, and to 112 languages by allowing up to 100 translations only. [Artetxe and Schwenk, 2019, p. 13]

4.2.2 Task

Tatoeba denotes a multilingual similarity search task, for which its own dataset can be used. The goal of the task evaluation is to pick the correct translations between the sentences in each `xx-en` data pair. For the similarity search, the parallel sentences per language pair can simply be shuffled. The original parallel data then naturally contains the correct, desired pairings.
Similarity search is performed by computing the cosine similarity (as defined in Eq. 3.1) between the sentences, where the sentence embeddings are derived from the model evaluated. For each sentence in language `xx`, its translation is considered to be the English sentence that produces the highest value. [Artetxe and Schwenk, 2019, p. 7]

4.2.3 XTREME

For this work, the Tatoeba task is experimented with in the context of the *Cross-lingual Transfer Evaluation of Multilingual Encoders (XTREME)*[4] benchmark [Hu et al., 2020], which was established for "evaluating the cross-lingual generalization capabilities of multilingual representations" [Hu et al., 2020, p. 1].
XTREME evaluates models on their Tatoeba performance using 36 language pairs (`xx-en`) from the original Tatoeba dataset, each pair containing up to 1000 parallel sentences. The actual data is downloaded from the LASER (Language-Agnostic Sentence Representations) repository[5]. Its details can be found in Table 4.1.

For evaluating specifically XLM-R on the task, the hidden states of its 8th layer are considered for the embeddings. The evaluation metric is the accuracy score, i.e. the percentage of correctly established translations.

[3] https://tatoeba.org/en/
[4] https://github.com/google-research/xtreme
[5] https://github.com/facebookresearch/LASER/tree/main/data/tatoeba/v1

4.3 Outlier Identification

This section investigates the models XLM-R, X2S-CCA and X2S-MSE on the topic of outlier embedding dimensions. Each model produces 768 embedding dimensions in each of the 12 model layers[6]. The investigations are carried out in the context of the Tatoeba dataset and task.
For all following experiments, the XLM-R model is instantiated from the Huggingface Transformers library [Wolf et al., 2020], whereas the X2S-MSE and X2S-CCA checkpoints are obtained directly from the authors [Hämmerl et al., 2022].

4.3.1 Method

In order to extract the embeddings the models produce, they need to be presented with textual input to encode. For this, the Tatoeba dataset as described in section 4.2 is employed. It encompasses 36 languages paired with their English translations, each set of languages containing up to 1000 sentence pairs. The following languages are included:

ISO	language	sentences	ISO	language	sentences
af	Afrikaans	1000	jv	Javanese	205
ar	Arabic	1000	ka	Georgian	746
bg	Bulgarian	1000	kk	Kazakh	575
bn	Bengali	1000	ko	Korean	1000
de	German	1000	ml	Malayalam	687
el	Greek	1000	mr	Marathi	1000
es	Spanish	1000	nl	Dutch	1000
et	Estonian	1000	pt	Portuguese	1000
eu	Basque	1000	ru	Russian	1000
fa	Farsi	1000	sw	Swahili	390
fi	Finnish	1000	ta	Tamil	307
fr	French	1000	te	Telugu	234
he	Hebrew	1000	th	Thai	548
hi	Hindi	1000	tl	Tagalog	1000
hu	Hungarian	1000	tr	Turkish	1000
id	Indonesian	1000	ur	Urdu	1000
it	Italian	1000	vi	Vietnamese	1000
ja	Japanese	1000	zh	Mandarin	1000

Table 4.1: Languages with their ISO-codes and number of available parallel sentences used for Tatoeba evaluation in XTREME.

The general procedure of acquiring a model's embeddings is to pass it textual input, and extract the hidden states of a specific model layer. The hidden states denote token embeddings of the passed input. In the context of the Tatoeba task, these embeddings are further processed into sentence embeddings, which are then examined for dimensions with exceptionally high magnitudes: this is given if a dimension's value exceeds the embeddings' mean by at least 3 standard deviations in either direction.

4.3.2 Sentence Embeddings

In order to obtain representations that describe entire sentences rather than single tokens, the mean pooling operation is applied. As also used in Sentence-BERT [Reimers and Gurevych, 2019], which focuses on deriving specifically sentence embeddings, mean-pooling simply computes the mean over a sentence's token representations. The result is a representation describing the entire sentence.

[6] See table with XLM-R_{base} architecture in 4.1.1

4.3.3 Outliers in XLM-R

The goal of the primary examination is to establish whether specifically XLM-R produces rogue embedding dimensions in the first place, and if so, which dimensions are affected. In order to establish this, the sentences of all 36 languages of the Tatoeba dataset are encoded by the model, after which one average embedding over all sentences is calculated.

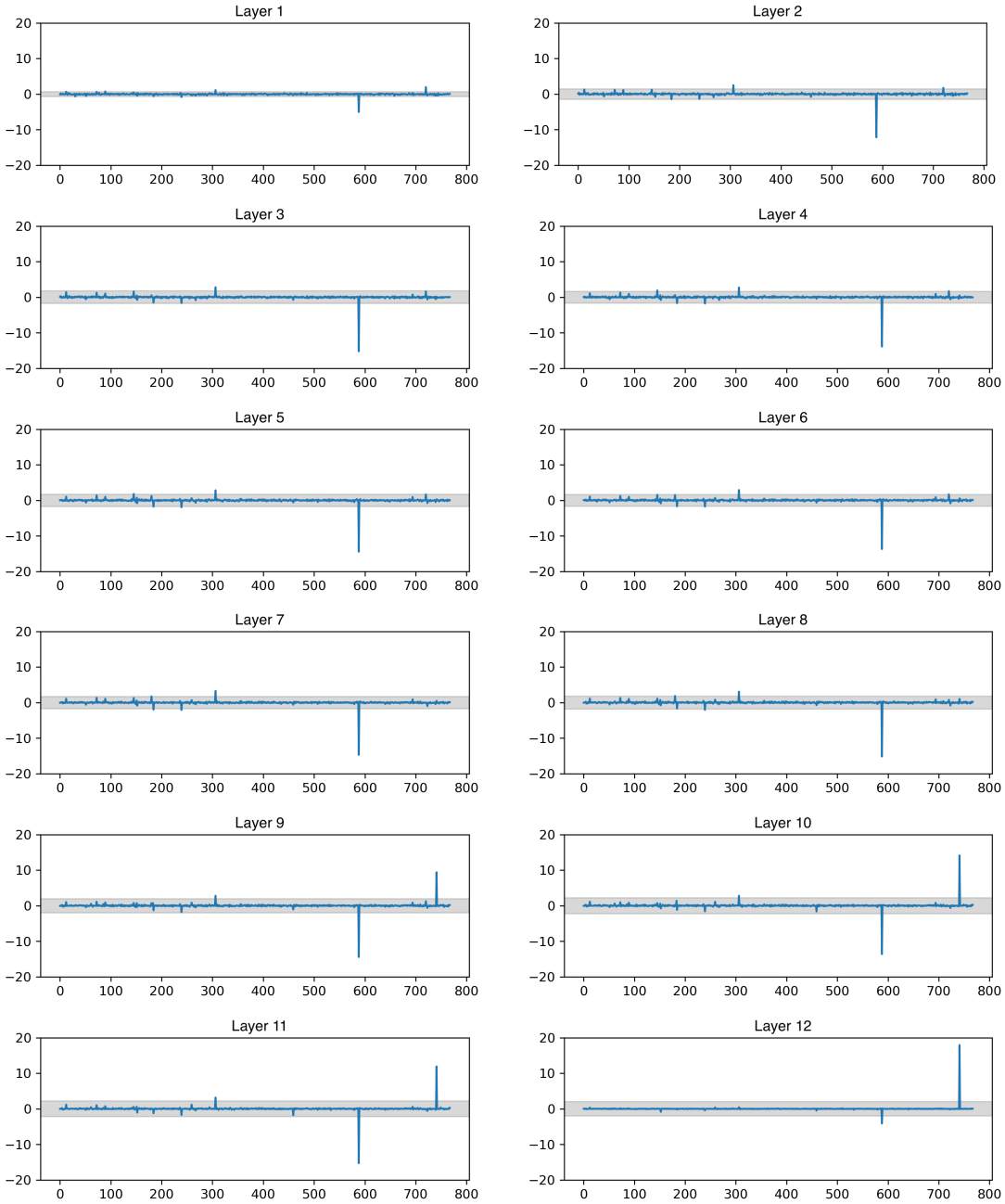


Figure 4.2: Average embeddings of XLM-R over all 12 model layers.

As shown in Figure 4.2, XLM-R produces outlier dimensions in each of the 12 layers. In order to formalize which dimension exactly to consider a rogue one, the method established in previous research [Kovaleva et al., 2021] is applied (see also section 3.1.1): the mean and standard deviation σ of all embedding dimensions are computed. **A dimension is then considered to be an outlier if it exceeds the mean by 3σ or more in either direction**. The grey area marks the space of the mean plus 3σ in the positive and negative direction in each layer respectively.

Following this formalism, the following outlier dimensions are established in XLM-R:

layer	outlier
1	89, 239, 306, **588**, 720
2	306, **588**, 720
3	306, **588**
4	145, 239, 306, **588**, 720
5	145, 184, 239, 306, **588**
6	184, 239, 306, **588**, 720

layer	outlier
7	180, 184, 239, 306, **588**
8	180, 239, 306, **588**
9	306, **588**, **741**
10	306, **588**, **741**
11	306, **588**, **741**
12	**588**, **741**

Table 4.2: Established outlier dimensions for all layers of XLM-R following the 3σ rule.

It is worth to note that the two most obvious outliers are dimensions `588` and `741`, which are visible in Figure 4.2 without additional formalisms. Interestingly, `588` is most prominent from layers 1 through 11 and drastically reduces in layer 12, whereas `741` only begins developing from layer 9 onwards and reaches its peak in the last layer.

4.3.4 Outliers in X2S-MSE and -CCA

An analogous analysis is performed with the models X2S-MSE and X2S-CCA. The following rogue dimensions are determined following the 3σ rule.

layer	outlier
1	12, 89, 184, 239, 306, **588**, 720
2	184, **588**, 720
3	**588**, 720
4	184, **588**, 720
5	184, **588**, 720
6	502, **588**, 720
7	180, 184, **588**
8	180, **588**
9	180, 404, **588**, 720, **741**
10	**588**, **741**
11	**588**, **741**
12	**588**, **741**

(a) X2S-MSE

layer	outlier
1	12, 152, 267, 306, **588**, 720
2	306, **588**, 720
3	306, **588**, 720
4	145, 184, 239, 306, **588**, 720
5	145, 184, 239, 306, **588**
6	145, 184, 306, **588**
7	184, 239, 306, **588**
8	184, 239, 306, **588**
9	306, **588**, **741**
10	306, **588**, **741**
11	306, **588**, **741**
12	12, **588**, **741**

(b) X2S-CCA

Table 4.3: Established outlier dimensions for all layers of X2S-MSE and X2S-CCA following the 3σ rule.

A few observations can be made here: for one, most importantly, both models' most prominent outliers are again `588` and `741`. This means that the continued pretraining as described in 4.1.2 did not influence the emergence of these two major outliers.
Apart from this it is also observed that the points of outlier emergence are highly similar to the pattern in XLM-R, where `588` exhibits the highest magnitude from the beginning layer and is maintained until almost the end, and `741` only emerges starting from layer 9. This is very similar in the two extended models, where `588` is the most outstanding until layer 9 in both X2S-MSE and -CCA, starting from where the `741` outlier emerges and becomes more prominent than `588`[7].

The last notable observation concerns the rest of the outlier dimensions established with the 3σ rule. Although being not as significant outliers, many of the dimensions established for XLM-R are again observed in X2S-MSE and -CCA. These include for example the dimensions `180, 184, 239, 306` and `720`.

[7]Note: this is not intended to be deduced from Table 4.3.

4.3.5 Summary

This section established rogue dimensions for all layers of XLM-R as well as two extensions of it, X2S-MSE and X2S-CCA. As the method for identifying outliers, the "3σ rule" was employed, which has also established itself in previous research in the field of rogue dimensions (see chapter 3).
As the two most significant outliers, dimensions `588` and `741` are observed, by far exceeding the magnitudes of any other dimension. It is worth noting that `588` is present from layer 1 onwards, whereas `741` emerges only starting from layer 9, where it begins "outgrowing" `588`. This is the case for all three models.
Apart from the two major outliers, **a number of other dimensions is marked as being rogue**, see Tables 4.2 and 4.3. With respect to `588` and `741` they may not look as significant, but compared to the average embedding value they do deviate far enough to fall out of the norm. Interestingly, the same "less significant" outliers in XLM-R are likewise observed in its extensions X2S-MSE and -CCA.
This indicates that neither the big outliers of XLM-R nor its less outstanding ones were affected by the continued pre-training procedure which extended the model to X2S-MSE and -CCA, respectively. **Each of XLM-R's outliers still emerged in either of the two extended models**.

After having established the models' rogue dimensions, the next question that arises is what role they may play in the representative quality of the embeddings. For investigations on this matter, they are examined in the context of the Tatoeba similarity search task.

4.4 Outliers' impact on similarity search

As explained in section 4.2, the Tatoeba similarity search task utilizes the cosine similarity measure to match the correct translations in an `xx-en` collection of sentences based on their sentence embeddings.
In the following it is investigated to what extent rogue embedding dimensions influence the Tatoeba task performance of the examined language models.

4.4.1 Baseline Tatoeba performance

The *unmodified* models' results on Tatoeba are referred to as the baseline task performances. To obtain them, each model is tested on the task in the context of the XTREME benchmark, where **specifically layer 8 embeddings are used**. The evaluation is done per each of the available 36 languages, averaging over which determines the overall accuracy of the model. The following scores are obtained:

Model	af	ar	bg	bn	de	el	es	et	eu	fa	fi	fr
XLM-R	51.6	35.8	66.9	28.7	88.4	51.6	71.0	44.2	26.1	64.4	63.9	72.5
MSE	10.9	3.9	17.1	2.4	42.5	5.1	15.2	7.9	7.4	10.5	12.7	22.2
CCA	**74.1**	**57.0**	**82.1**	**54.9**	**95.4**	**72.5**	**88.6**	**75.2**	**52.5**	**79.9**	**84.3**	**84.3**
	he	hi	hu	id	it	ja	jv	ka	kk	ko	ml	mr
XLM-R	51.7	50.5	58.7	68.6	64.7	52.8	15.1	37.1	33.2	50.1	**54.7**	**38.0**
MSE	10.1	9.0	13.4	14.3	11.5	10.0	5.4	4.9	6.1	10.5	4.5	5.3
CCA	**71.7**	**70.1**	**80.2**	**86.4**	**82.3**	**74.0**	**22.9**	**63.8**	**62.3**	**63.2**	25.5	34.9
	nl	pt	ru	sw	ta	te	th	tl	tr	ur	vi	zh
XLM-R	76.8	76.6	69.8	15.6	25.1	30.8	34.7	29.7	54.9	31.1	67.7	59.4
MSE	17.8	19.7	12.5	4.1	1.9	3.4	1.6	6.8	6.8	2.5	15.6	6.1
CCA	**89.3**	**90.4**	**85.6**	**23.8**	**56.3**	**59.4**	**68.4**	**45.1**	**78.0**	**45.9**	**84.4**	**85.2**

XLM-R	X2S-MSE	X2S-CCA
50.35	10.05	**68.06**

Table 4.4: Baseline Tatoeba performances of XLM-R, X2S-MSE and X2S-CCA.

Table 4.4 shows how the X2S-CCA model consistently outperforms the other two with an average accuracy of 68.06%. It is followed by the original XLM-R model with a performance of 50.35%, whereas the X2S-MSE model evaluates to a significantly worse score at 10.05%.

It should be pointed out here that these performances on sentence similarity search were carried out with the original models' embeddings, which *do* include outlier dimensions as established before. Specifically, the embeddings that are produced by the 8$^{\text{th}}$ layer of each model were used for the results in Table 4.4.
It is in the following examined whether their removal is able to improve or worsen the task performance on Tatoeba.

4.4.2 Removing outliers

The most straightforward method to remove rogue dimensions from the embeddings is to simply zero them out, i.e. in each sentence representation, replace the value in the outlier dimension with a zero (see also 3.3.1). In this section, this is done for each outlier by letting the models encode the sentences, then removing one outlier at a time from the resulting embeddings, after which the modified embeddings are used for the task.
The goal of this series of experiments is to understand whether the removal of a rogue dimension from the embeddings influences the similarity search performance in any way.

Since the task evaluation in XTREME uses the 8$^{\text{th}}$ layer of each model for the embeddings, the rogue dimensions established for specifically the 8$^{\text{th}}$ layers are zeroed out here. Referring to the outliers identified in tables 4.2 and 4.3, the following dimensions are tested:

model	layer 8 outliers
XLM-R	180, 239, 306, **588**
X2S-MSE	180, **588**
X2S-CCA	184, 239, 306, **588**

Table 4.5: Layer 8 outliers of all three models.

As dimension `588` was found to be the most outstanding one in layer 8, it is the first to be tested by removal. The following accuracy scores are achieved by the models when clipping `588` from each of their embeddings:

Model	af	ar	bg	bn	de	el	es	et	eu	fa	fi	fr
XLM-R	51.2	35.8	67.6	28.5	88.9	52.7	70.9	48.4	31.7	66.2	66.5	73.1
MSE	11.8	4.5	16.9	2.0	42.8	5.0	16.0	8.2	8.2	11.0	14.5	23.2
CCA	73.9	59.7	82.7	55.8	95.5	74.5	87.9	74.3	52.1	81.1	84.2	85.4
	he	hi	hu	id	it	ja	jv	ka	kk	ko	ml	mr
XLM-R	54.8	48.5	64.0	71.6	65.6	57.0	15.6	42.6	41.0	54.9	59.1	41.9
MSE	11.8	9.2	14.2	15.6	12.0	11.1	6.3	5.4	6.3	11.4	5.7	6.2
CCA	71.3	75.4	82.0	87.2	82.4	75.8	26.3	65.8	62.4	67.9	63.2	52.6
	nl	pt	ru	sw	ta	te	th	tl	tr	ur	vi	zh
XLM-R	78.7	78.4	70.9	16.7	27.7	35.9	41.1	31.5	61.3	35.8	70.3	61.4
MSE	19.2	20.1	13.3	3.8	2.6	4.3	2.6	7.0	7.8	2.9	17.3	7.6
CCA	89.3	90.5	86.3	26.1	63.9	70.9	74.6	45.2	78.7	57.9	86.6	85.1

XLM-R	X2S-MSE	X2S-CCA
52.99	10.77	71.52
(+2.64)	(+0.72)	(+3.46)

Table 4.6: Tatoeba performances of the models, `588` removed.

In Table 4.6, green values indicate an increased score for the particular language, red values denote a decreased score, and black ones mean that the score has not changed.

The major observation in this experiment is the consistent increase in task performance after removing the rogue dimension `588`: for nearly all languages, the similarity search performs better without the outlier. In some extreme cases even, the score increases an especially big amount, e.g. for the X2S-CCA model, the accuracy for Malayalam (ml) jumps from 25.5 to 63.2, and for Marathi (mr) it increases from 34.9 to 52.6, which are improvements of +37.7 and +17.7, respectively. In some cases, the performance worsens when the outlier is removed. However the decrease is not particularly big: on average, it lies at around -0.43.

Overall the clipping of the major outlier `588` benefits all three models. On average, their performances rise by +2.64, +0.72 and +3.46, respectively. The biggest difference is seen in XLM-R and X2S-CCA, whereas the performance of X2S-MSE does not improve much. This may be the case, because the baseline performance of the model was rather low already. Its embeddings may therefore be flawed in another way that makes it perform worse on tasks like similarity search. Therefore, zeroing out the rogue `588` dimension has improved its accuracy score by a bit, but it could not drastically rise its task performance.

Moving forward, the rest of the established rogue dimensions (as shown in Table 4.5) are likewise zeroed out for testing on Tatoeba. For clarification, *each run has only a single clipped outlier*, i.e. the following experiments do not additionally remove outliers after already having zeroed out `588` – each new outlier is removed from the original, unmodified sentence embeddings. This shows each outlier's effect in isolation.

outlier	**XLM-R**	**X2S-MSE**	**X2S-CCA**
180	50.57 (+0.22)	10.1 (+0.05)	–
184	–	–	68.15 (+0.09)
239	51.11 (+0.76)	–	68.59 (+0.53)
306	50.59 (+0.24)	–	68.18 (+0.12)

Table 4.7: Tatoeba performances of the models, other outliers removed.

While the other dimensions, which were likewise considered rogue by the 3σ rule before, also cause an increase in task performance, the improvements are comparatively small in contrast to the effect of removing `588`.
It is however notable how there is no case in which the removal of an outlier dimension causes a *decrease* in accuracy: whether the gain is small or big, removing a rogue dimension seems to be beneficial for similarity search either way.

4.4.3 Rescaling outliers

As an alternative method for outlier elimination, the idea of rescaling is proposed in this section. This suggestion stems from the idea that the outliers may not be inherently harmful dimensions, i.e. the features they respectively encode may in reality be important to the representational power of the embeddings. Possibly the dimensions which ended up rogue are simply scaled wrong.

In order to test this theory, the same outliers from Table 4.5 are rescaled in a way that brings them closer to zero, instead of zeroing them out entirely as in the previous section. As a starting point, it is experimented with the biggest outlier `588` first to find a reasonable scaling factor.

To estimate by which factor to rescale a dimension to reduce its magnitude, it is sensible to base this value roughly around the value itself: i.e. if a dimension's value on average lies at around 8, rescaling it by ⅛ (meaning, dividing it by itself) would reduce the value to 1. It can then be experimented with further factors that rescale the dimension to be closer to zero, or on the other hand make the rescaling less drastic.

Applying this method to the outlier dimension 588, the factors are based around its average value, which lies around -15. This is established through the average embeddings previously calculated in section 4.3.3, which computed one mean embedding averaged over all sentence representations for each layer. Note that currently specifically layer 8 is investigated.
In the following, different factors around the value 15 are tested. In detail, the rescaling is applied to the rogue dimension 588, and the resulting modified embeddings are used for testing on the Tatoeba task.

scaling factor	performance
none	50.35
$\frac{1}{5}$	52.91
$\frac{1}{10}$	52.96
$\frac{1}{15}$	**53.02**
$\frac{1}{20}$	52.99
zero out	52.99

Table 4.8: Tatoeba scores for different scaling factors, outlier 588.

Table 4.8 shows the results of different scaling factors for 588, sorted from reducing the outlier's magnitude the least ("none") to reducing it the most ("zero out"). The performance of the factor "none" corresponds to the model's baseline performance with unmodified embeddings.

It can be seen that reducing the outlier by mid-range factors like $\frac{1}{5}$ and $\frac{1}{10}$ already shows significant improvement in model performance, with gains of +2.56 and +2.61. The best result however is achieved with the factor $\frac{1}{15}$, which even produces a slightly better result than the method of zeroing the outlier out. By bringing the outlier, which has its average value at around -15, even closer to zero by using the factor $\frac{1}{20}$, no further improvement can be achieved. In fact, the exact same result is produced as with zeroing out.

It is deduced from the results that a reasonable scaling factor is to divide the rogue dimension in question by its average value. In the case of the biggest outlier 588, this is shown to be a good choice. Consecutively, this hypothesis is tested for the other outlier dimensions of XLM-R[8].

factor	performance
none	50.35
$\frac{1}{\text{avg}}$	**50.63**
zero out	50.57

(a) 180

factor	performance
none	50.35
$\frac{1}{\text{avg}}$	51.01
zero out	**51.11**

(b) 239

factor	performance
none	50.35
$\frac{1}{\text{avg}}$	50.55
zero out	**50.59**

(c) 306

Table 4.9: Tatoeba scores after rescaling, other XLM-R outliers.

XLM-R's remaining layer 8 outliers 180, 239 and 306 are each scaled down by their respective average values. For example, the average value of dimension 239 lies at around -2. Its scaling factor $\frac{1}{\text{avg}}$ therefore corresponds to ½.

[8]The X2S-CCA and -MSE models are omitted here.

As can be seen in the results of Table 4.9, the hypothesis that rescaling a dimension by its average value may be a better choice than zeroing the dimension out does not hold by default. In the cases of 588 and 180, rescaling produced a slightly better result on Tatoeba. For 239 and 306 however, zeroing out was the better choice.

It is important to note that this section does not perform an exhaustive search for scaling factors. It may therefore be the case that for each of the dimensions there exists an optimal factor that is not taken into account here.

The overall takeaway from these experiments is that reducing the magnitude of an outlier, whether by rescaling or zeroing it out entirely, is beneficial for the similarity search in either case. In some cases, if a good scaling factor is found, rescaling can be a better option. However the advantage of rescaling compared to zeroing out is minimal in these instances (+0.03 for 588, +0.06 for 180).

4.4.4 Testing random dimensions

In order to verify that the above results, i.e. the performance improvements after removing or rescaling the outliers, are not just a coincidence that may be caused by removing any other dimension, a number of random dimensions is considered as a counter-check.
For this, 10 numbers in the range of [0, 768] are randomly generated, making sure that none of them correspond to outlier dimensions in neither model. The counter-check removes (i.e. zeroes out) each of the random dimensions for each model, after which the modified embeddings are tested on Tatoeba.

model	547	64	48	13	397
XLM-R	-0.01	-0.02	+0.00	-0.01	+0.04
CCA	+0.01	-0.01	+0.02	-0.06	-0.02
MSE	+0.01	+0.02	+0.00	+0.02	+0.01

model	209	358	97	567	702
XLM-R	-0.02	-0.01	-0.01	+0.00	-0.01
CCA	+0.01	+0.02	+0.01	-0.02	+0.00
MSE	+0.02	+0.00	+0.00	+0.00	-0.01

Table 4.10: Removing random dimensions for all three models, scores on Tatoeba.

The above table shows the in-/decrease in Tatoeba task performance for each model, after the respective dimension is zeroed out. It can be clearly seen how in none of the overall 30 runs the score increases nearly as much as it does when real outliers are removed. In several cases, there is even a decrease in performance, which was not seen after removing any of the rogue dimensions.

The important insight here is that, whether it is an increase or a decrease in score, the effects of removing these dimensions are practically non-existent. This supports the results seen with outlier dimensions before in the sense that those results are no coincidence: a visible improvement in similarity search can be achieved by reducing rogue dimensions, it cannot be achieved with random, non-rogue ones.

4.5 Observations

This chapter has proven the presence of rogue dimensions in the embeddings that are produced by XLM-R, as well as its two expansions X2S-MSE and X2S-CCA. The following key observations are made during the outliers' examination.

Analyzing embeddings across all 12 model layers, it is observed how especially the two major outliers, dimensions `588` and `741`, develop. Figure 4.2 illustrates clearly how `588` dominates most layers, but `741` emerges later on and becomes the most prominent by the time the final layer is reached. A similar phenomenon is the case for the two other models.
It can therefore not clearly be said, *which dimension(s) is/are the models' biggest outlier – it highly depends on the layer in question.*

The 3σ rule used for identifying rogue dimensions establishes a number of additional dimensions beside the two major ones. When removing each of the outliers, whether through zeroing out or rescaling, the performance on the Tatoeba similarity search improves. The best improvement is seen after removing the biggest outlier of layer 8, dimension `588`. All smaller outliers cause smaller score improvements.
It needs to be further investigated *whether there is a direct relationship between the magnitude of an outlier and the effect of its removal.*

In previous research, see especially section 3.3.2, it was shown how removing a language model's outliers causes big drops in performance on downstream tasks. The same could have been expected to be the case for the Tatoeba similarity search task, however the opposite is the case: removing any of the identified outliers causes an *improvement* in task performance.
This opens up the question, *why in specifically this task the presence of such outlier dimensions seems to be harmful.*

Chapter 5

Detailed Analysis

The previous chapter explored the chosen models on the topic of rogue dimensions in a generalized way in order to gain first insights about outliers in their embeddings. This, for one, meant to average over all available languages' sentence representations and work with one single embedding per model layer. Secondly, regarding the evaluations on the Tatoeba task, only the average score improvements are focused on. Although, e.g. for `588`, the accuracy scores were presented for each language (see Table 4.6), the results were still not analyzed in detail.

Since this work is concerned with a *multilingual* model, as opposed to previous research which focused largely on monolingual ones, it is important to pay attention to this factor and include the multilinguality into the analysis. This is in the following done for the outliers themselves, as well as for the models' performance on similarity search. In this chapter, only XLM-R is considered, specifically the embeddings of its 8^{th} layer since they are tied to the results on Tatoeba.

5.1 Lower level embeddings

To establish the models' rogue dimensions, chapter 4 examined *one single representation* per layer, which is a result of averaging over all sentence representations, encompassing 36 different languages and 31,692 sentences in total. This section now revisits the investigation of outlier dimensions by decomposing these generalized embeddings into "lower levels": the language embeddings, which are a product of their sentence embeddings, and the sentence embeddings themselves.

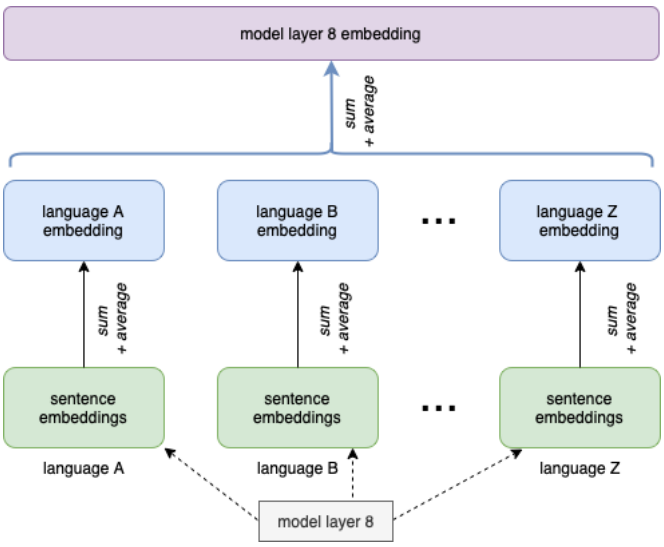


Figure 5.1: The averaging process leading to the generalized embeddings.

The reason for returning to these components is that some information may have been lost in the process of averaging: as an extreme example, dimension 588 in embeddings of language A may lie at +35, whereas language B may lie at -5. Averaging over the two languages would produce the result that the outlier 588 lies at the value of +15, but this would be a very inaccurate statement regarding single languages.

5.1.1 Language level

Stepping one level lower from the generalized embedding leads to language level representations, as also visualized in Figure 5.1. For each language, these are the product of averaging over each such language's sentence representations.

Returning to them allows one to examine *language-specific* embeddings on the topic of outliers, rather than one generalized embedding representing *all* languages.

For reference, the following rogue dimensions were established for the generalized layer 8 embedding of XLM-R:

	180	**239**	**306**	**588**
value	1.87	-2.06	3.07	-15.16

To now analyze the language embeddings, the same procedure as in chapter 4 is applied, establishing rogue dimensions through the 3σ rule. For each of the found outliers, it is examined how pronounced they are in the respective languages magnitude-wise and whether any big differences can be observed. For this experiment, only XLM-R is considered, specifically its layer 8 embeddings.

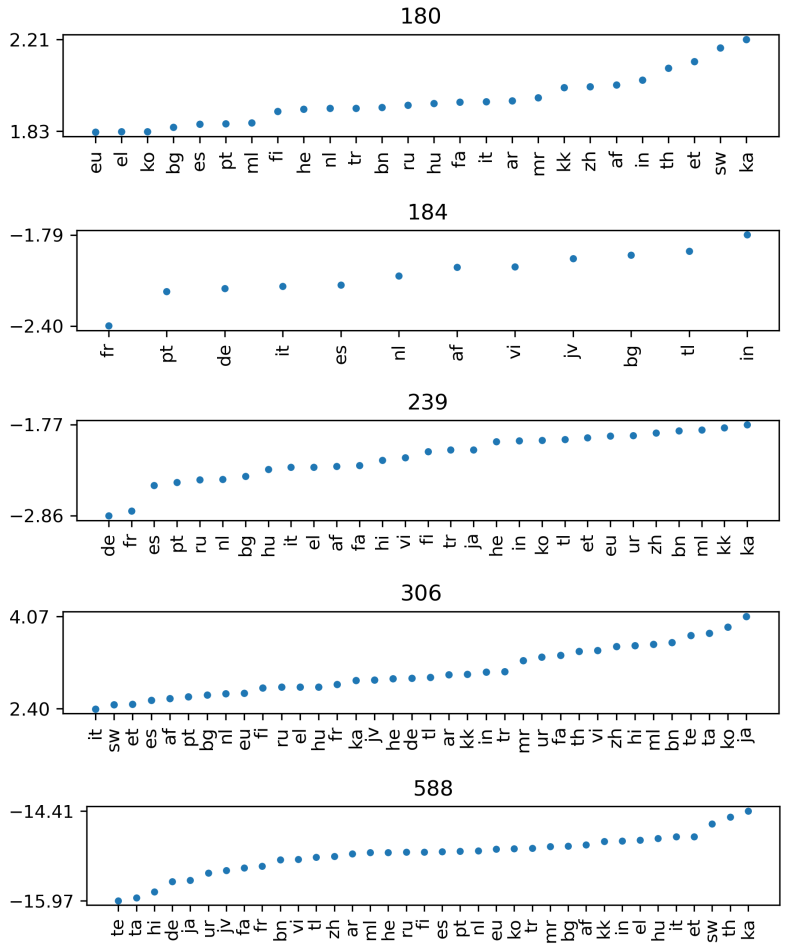


Figure 5.2: Rogue dimension values per language embedding (XLM-R layer 8).

The above figure features all dimensions that were established to be outliers by any of the language embeddings. While including all outliers that were found for XLM-R in the previous chapter in a generalized manner (`180, 239, 306` and `588`), the results here feature an additional outlier `184`, which however holds for only 12 of the 36 languages. This is also an interesting observation concerning the rest of the outliers: `306` and `588` are both outliers in all of the 36 languages. However `239` and `180` are considered rogue in the embeddings of only 29 and 27 languages respectively.

Concerning the actual magnitudes of the outliers in each language, it can be seen that none of them deviate far from their averaged value of the generalized embedding: for example, the generalized embedding reached -15.16 in its dimension `588` – looking at the language representations that make up this generalized embedding, none of them deviate extremely far from this average, as e.g. suspected in the introduction of this section.
Formalizing this observation, the ranges[1] spanned by the language-level outlier values are also representative of this finding:

	180	**184**	**239**	**306**	**588**
range	0.38	0.61	1.09	1.67	1.56

The wider the value-range exhibited by the languages, the bigger the chance is that a language itself influences how pronounced an outlier becomes. Here, this is the case to some extent, as certain languages like `ka` tend to produce the highest values, whereas others like `de` tend to settle in the lower range. The effect should however be studied in more detail to draw definite conclusions.

Following this analysis, it can be concluded that the rogue dimensions established for XLM-R in chapter 4 are overall stable observations. As it was found, there are no languages whose outliers differ extremely from the outlier values of the generalized embedding. Therefore in this sense, **no crucial information was lost in the process of averaging**.
It is also found that there are very stable outliers like `306` and `588`, since they are found as outliers in all of the considered languages, and less stable ones like `180` and `239`, which are found to be rogue in only a part of the languages. Therefore, the previously established dimensions `306` and `588` can be considered as universal outliers of XLM-R, at least in its 8^{th} layer.

5.1.2 Sentence level

An even more fine-grained way of going about the analysis is to examine outliers per sentence in each language. This is done to understand whether big outlier values may be caused by specific sentences.
For insights on this matter, the rogue dimensions `588` and `306` are chosen, since firstly, they were established to be consistent outliers across all languages in the previous section, and secondly, exhibit the largest average values and therefore may return interesting results.

In detail, this is investigated the following way: the sentence embeddings of all languages are considered. For each language, the values of their sentence embeddings in the respective dimensions are plotted, such that the x-axis denotes each single sentence, while the y-axis denotes their embedding values in the respective outlier dimension. While the results are examined for all languages, the plots of only two

[1] In this context, *range* denotes the difference between the biggest and smallest value of the distribution.

languages per outlier are shown here due to space reasons. The languages chosen for visualization are the ones that produced the lowest and highest values in the language embeddings in the previous section, see Figure 5.2. These are `it` and `ja` for `306`, and `te` and `ka` for `588`.

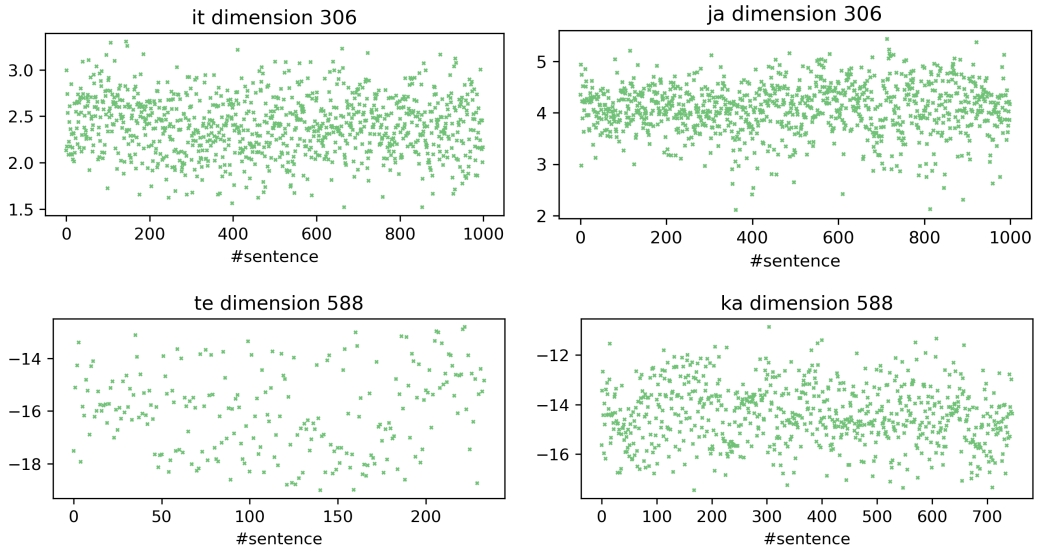


Figure 5.3: Rogue dimension values for sentence embeddings of chosen languages (XLM-R layer 8).

Figure 5.3 visualizes the distribution of outlier values in the chosen languages' sentence embeddings. It is noted that `te` and `ka` exhibit less instances, because the Tatoeba dataset provides less sentences for these languages (further specified in Table 4.1).
As can be seen, single sentences likewise exhibit varying values in the outlier dimensions. To place these results into the general context of outlier magnitudes, the following is a reference for their values in the *generalized embeddings*, and the lower level of *language embeddings*:

generalized embedding values

306	**588**
3.07	-15.16

language embedding values

	306		**588**
`it`	2.40	`te`	-15.97
`ja`	4.07	`ka`	-14.41

Since for language embeddings, the two languages were chosen which exhibited the lowest and highest values respectively, the table of *language embedding values* can also be seen as the minimum and maximum values produced in those rogue dimensions on a language level. The following now formalizes the above plots, including the smallest and biggest values exhibited by *sentence embeddings*:

sentence embedding values

	306		**588**
`it`	(1.52 / 3.30)	`te`	(-18.99 / -12.81)
`ja`	(2.11 / 5.43)	`ka`	(-17.45 / -10.88)

Having collected all of the results above, it can now be seen how accurate the generalized embeddings established in the previous chapter are.

Taking for instance `306`, its value after all levels of averaging results to 3.07. Going one step back to language embeddings, the lowest value produced by a language is 2.40, and the highest is 4.07. Now stepping even one level further down to sentence embeddings, there is a sentence that produced the value 1.52 in this dimension, while another reached 5.43.

To summarize, this means that the **values reported in the generalized embedding's rogue dimensions are truly just an average of all its lower level components**: the magnitude of `588` being reported to lie at -15.16 does not mean that all sentence embeddings reach about this value in the outlier dimensions: they reach anything between -18.99 and -10.88, which is a rather wide span.

The analyses of these sections have also shown that firstly, **different languages reach different magnitudes** in the established outliers, and secondly, **different sentences can heavily influence how pronounced an outlier becomes**. It is however *not* the case that, especially in the biggest outlier `588`, there are instances that show no rogue behavior – all produced values on any embedding level *are* rogue in these dimensions. The specific values vary within some range around the ones in the generalized embeddings, but they don't completely fall out of their scope.

5.2 English side of Tatoeba

Up until this point, this work was only concerned with one side of the Tatoeba dataset. However the evaluation on the task always involves two languages: the sentences of the language in question, i.e. one of the 36 languages listed before, and their potential *English translations*.

Because the similarity search performed in the context of the task relies directly on the sentence representations of both languages in a pair, it is crucial to also inspect the English side of embeddings on outliers. As English is the language that XLM-R has seen the most data in during pre-training (see Figure 4.1), it may for example be the case that the outlier dimensions of English sentence embeddings are not as pronounced, or even non-existent.

If such discrepancy is the case, this may be a big factor that influences the cosine similarity search of Tatoeba, and consequently, the mediocre task performance, which for XLM-R lies at 50.35%. Therefore it is in this section analyzed, whether English embeddings exhibit any behavior that deviates from the rest of the languages.

5.2.1 Outliers

As a first step it needs to be verified that the same outlier dimensions are likewise present in English sentence representations. For this, all English sentences available in the Tatoeba dataset are encoded and averaged over. This resulting average representation is then examined for rogue dimensions following the previously used 3σ convention.

The following generalized representation for English sentences is observed:

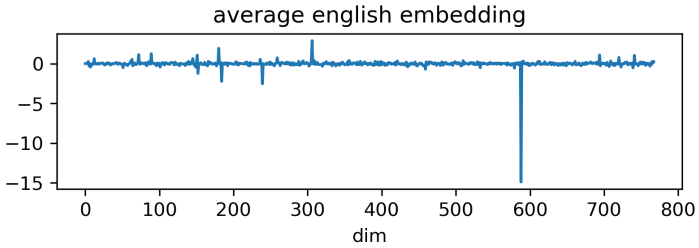


Figure 5.4: English embedding averaged over all English Tatoeba sentences. All sentence embeddings were produced by XLM-R's layer 8.

In this averaged English representation, the following rogue dimensions with their respective magnitudes are established. For reference, the results of the previously analyzed generalized embedding, which averaged over all other languages, are included here again.

english embedding outliers

	180	**184**	**239**	**306**	**588**
value	1.96	-2.21	-2.51	2.93	-14.91

generalized (36 languages) outliers

	180	**239**	**306**	**588**
value	1.87	-2.06	3.07	-15.16

The above results show that encoding English sentences with XLM-R, specifically layer 8, produces the same rogue dimensions as for the other languages. The magnitudes of the outliers are likewise close to the previously established ones.
The only difference is the English embedding's outlier `184`, which could before only be established for 12 out of the 36 languages (see section 5.1.1).
Whether or not `184` being more of an outlier in English embeddings plays a distinctive role in the performance of the Tatoeba task will be addressed in the next chapter.

5.2.2 Comparison of parallel sentences

Since it is verified that encoding English input leads to the same rogue dimensions[2] as seen for the other languages, it is now investigated whether there are any systematic differences in the representations on a sentence level.

Because the Tatoeba similarity search is carried out relying directly on the sentence embeddings of a language pair `xx-en`, it is sensible to compare the parallel sentences of these pairs on the topic of the established rogue dimensions.

The way this is done in this section is to visualize the sentence representations of both languages in a pair in a similar manner as in Figure 5.3, but including both languages in the plot. These visualizations are generated for all outliers established for English, i.e. `180, 184, 239, 306` and `588`. After analyzing the findings one by one for each language pair per dimension, the key observations are presented here.

Note that the following plots show *parallel sentences* of the language in question and its English translations. Because of this, it is desired for the single values to be as close to each other as possible. It is therefore primarily looked out for visible deviations of the English sentence embeddings from the ones of language `xx`.

[2] I.e. each outlier established before is also an outlier for English encoded input, not the other way around.

In the plots, green markers denote language `xx` and blue markers denote English.
It can be observed that overall, what concerns the dimensions `180, 239, 306` and `588`, the values in the language pairs are reasonably well aligned. In this context, "aligned" means that the values are in the same area magnitude-wise and follow similar shapes, making single values be close to each other, which is desired in representations of sentences that are translations of each other. "Well aligned" instances can e.g. be seen in the following, taking specifically `588` as an example:

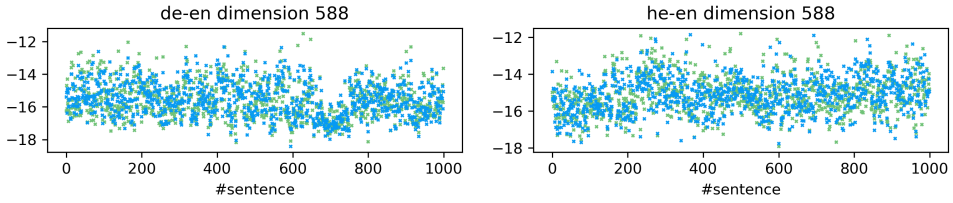


Figure 5.5: Well aligned examples of parallel Tatoeba sentences in rogue dimensions.

Such well aligned cases can be found numerous times in each of the above mentioned outlier dimensions, although they are most common in `588` and `180`. There are however also plenty of instances where the parallel sentences exhibit values more differing. In those cases, it seems as though the two languages are scaled slightly differently and therefore accumulate in different areas of magnitude:

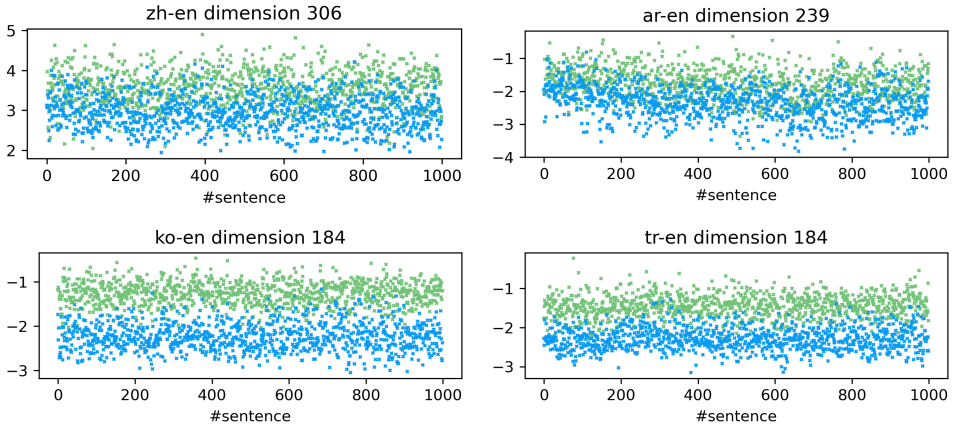


Figure 5.6: Worse aligned examples of parallel Tatoeba sentences in rogue dimensions.

In dimensions `180, 239, 306` and `588`, in such instances it is predominantly the case that the two groups of embedding values still overlap to some extent. Specifically in dimension `184` however, which was marked as rogue for English input but not generally for the other languages, the groups are separated very clearly.

In both of the cases it is not determined how much single values, which should ideally be as close as possible to each other, actually deviate from one another. The above analysis is to be seen as an approximate evaluation of the situation between `xx-en` sentence pairs in context of the rogue dimensions.

The main conclusion to be drawn from this is that there are indeed differences between the languages tested in Tatoeba and their English counterparts, which are to be matched through cosine similarity in the process of the task. How big of a role each rogue dimension actually plays for the similarity search will be evaluated further as this work progresses.

5.3 Impact on Tatoeba rankings

Sections 5.1 and 5.2 of this chapter have in greater detail analyzed rogue dimensions focusing on the embeddings themselves. As a result, the representations generated by XLM-R are now better understood with respect to outliers. This section now extends the detailed analysis into the direction of the Tatoeba task and the performance of XLM-R's embeddings on it.

The way to gain more specific insights about its results in context of removing certain outlier dimensions is to firstly understand what exactly happens during evaluation. In an `xx-en` collection of sentences, where for each `xx` sentence there is a correct `en` translation, for each `xx` sentence the `en` collection of potential translations is searched for the most suitable candidate. This is done by calculating the cosine similarity of this sentence with *all* available English ones based on their sentence embeddings which are generated by the model to be evaluated on the task.

This process produces a ranking of the English sentences for each `xx` sentence, sorted from having produced the highest cosine similarity to the lowest. The one which exhibits the highest cosine similarity is then regarded as the translation of the current `xx` sentence in question. This process is repeated for all `xx` sentences until each of them is paired with a potential `en` translation. The accuracy of this result is then established by comparing the produced pairings with the correct ones, i.e. the original `xx-en` parallel sentences.

To now analyze what changes happen to the task performance when rogue dimensions are removed from the embeddings, more detailed results can be extracted instead of simply the achieved accuracy scores: for the following analyses, the evaluation code is modified such that not only the final pairings are saved, but the entire rankings produced by cosine similarity search for each `xx` sentence. Additionally to the ranking of English sentences, each of their achieved cosine similarity scores is saved. For insights on this matter, the outlier dimensions `588` and `306` are chosen.

5.3.1 Ranking correlations

One aspect to further investigate is the extent to which the similarity rankings change when an outlier dimension is removed from the sentence embeddings.

The accuracy scores include only information about the first positions in those rankings, i.e. the English sentences that were actually chosen as the correct translations. This section however examines how *all* ranking positions change. The reason for doing this is to understand whether the removal of a rogue dimension affects the rankings just enough to push a different sentence to the first position, but otherwise doesn't do much, or if by doing so an entirely different ranking is produced.

To assess the described effect, the *Spearman rank-order correlation* is employed, which "measures the strength and direction of the monotonic relationship between two ranked variables" [Laerd Statistics, 2018]. It is calculated as:

$$\rho = \frac{1 - 6 \sum d_i^2}{n(n^2 - 1)} \tag{5.1}$$

In the above formula, n denotes the number of observations and d stands for the difference between the two rank values of each observation. This measurement is applied to the Tatoeba similarity search results by comparing the rankings produced by the original embeddings with the rankings produced by the embeddings where an outlier dimension had been removed.

In order to compare the rankings correctly, they need to be transformed into real rankings first. What is produced by the similarity search are lists of sentence IDs, sorted from being at rank 1 to being at last rank. For the Spearman correlation to work, these results need to be modified the following way:

produced by original embeddings

ranked sentence IDs	141	23	285	10	97	...	103
ranking	1	2	3	4	5	...	250

produced by modified embeddings

ranked sentence IDs	285	23	141	88	43	...	56
ranking	3	2	1	45	19	...	187

Table 5.1: Transformation of ranked sentences to rankings for Spearman correlation.

Taking the sentence ID ranking produced by the original embeddings, each position is simply replaced by a number in an ascending manner starting from 1. This ranking itself is always the same, although the associated sentence IDs will change for each tested instance. It is only important to save which sentence ID is linked to which rank in this original ranking.

As for the results produced by the modified embeddings, the ranked list of sentence IDs will have changed. For example, sentence `285` which came in 3$^{\text{rd}}$ place originally[3], may have risen to 1$^{\text{st}}$ place here. In order to correctly compare how this ranking changed from the original one, *each sentence ID needs to be given the ranking number it achieved originally.* Therefore here, `285` is assigned the rank 3. The same principle applies to all other sentence IDs of this instance.

The reason why the IDs themselves can't be used for comparison here, is because of the way the Spearman correlation works: as pointed out, the variable d in Equation (5.1) stands for the difference between two ranks. By including this into the calculation, the Spearman correlation accounts for how far each rank has moved. If instead of the ranks, the actual IDs are compared, for position 1 a difference of $285-141 = 144$ would be produced, instead of $3-1 = 2$. A difference of 2 is however correct, since the sentence `285` has indeed moved up by 2 ranks.

Computing the correlation for each sentence-ranking of each language and then taking the mean value, the following results show how much the rankings shift on average, when a certain outlier dimension is prior removed from the embeddings. The correlation coefficient ρ can take on values from -1 to +1, where -1 corresponds to a negative association of ranks, +1 means a positive association of ranks, and 0 means that there is no association [Laerd Statistics, 2018].

In this use case, -1 does not generally occur since this would mean that the ranks are perfectly inverted. +1 means that the two compared rankings are identical, and 0 means that they are entirely different. It is also noted that, for saving storage space, only the first 250 positions of each sentence-ranking are saved and evaluated instead of up to a thousand[4].

The following shows the correlation coefficients reached by the layer 8 embeddings of the XLM-R model by comparing the original results with the ones achieved after removing dimension `588`, `306` and `741`, respectively. Since `741` was not an

[3]Note: the sentence ID rankings in this example are fictional.

[4]Reminder: most Tatoeba languages include up to 1000 sentences. For those including less than 250, all available ranks are saved.

outlier in layer 8, it is here shown as comparison as to what happens to the ranks, when a non-rogue dimension is removed.

	original vs. no `588`	original vs. no `306`	original vs. no `741`
first 250 positions	0.31	0.86	0.98
first 50 positions	0.11	0.81	0.98
first 10 positions	0.06	0.76	0.97

Table 5.2: Spearman ranking correlations between results of original and modified embeddings (XLM-R layer 8).

Since for the final result it is most important which sentence gets first rank, the correlation is also measured for ranges closer to the first positions, i.e. aside from the first 250 positions, the first 50 and finally the first 10 are also taken into account to visualize a dynamic.

As can be deduced from Table 5.2, the sentence-rankings produced by embeddings where dimension `588` was zeroed out are drastically different from the original rankings: considering the first 250 positions, the correlation lies at 0.31. As explained above, the closer this value gets to 0, the less of a correlation there is. Falling even lower to 0.11 for the first 50 positions and 0.06 regarding positions 1 through 10, this shows that these rankings *are almost completely different to the original ones*. Removing the outlier `306`, the effect is less pronounced with values from 0.86 to 0.76 in the first positions, meaning that the ranks produced here are to some extent different, but not as extremely as with `588`.

To compare the effects of removing outliers with removing non-outliers, it can be seen that embeddings without dimension `741`, which is not rogue in layer 8 representations, the *produced rankings are almost identical* to the originals, as the correlations are consistently very close to 1.

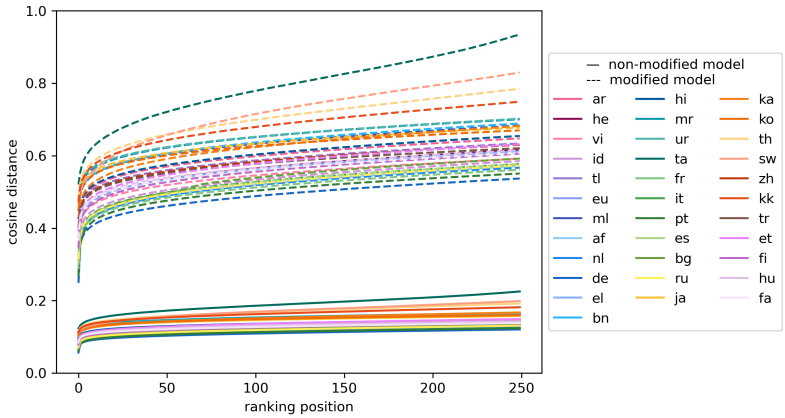
The analysis of ranking correlations in this section has shown how much the removal of rogue dimensions from the sentence representations impacts the result of similarity search in detail: not only do different sentences move to rank 1, thus resulting in better accuracy on the task – **the outliers' removal instead shuffles nearly the entire rankings**, making these single dimensions impact the entire task result to a high degree. This is especially the case for the biggest outlier `588`, as its removal produces almost completely different sentence-rankings.

5.3.2 Cosine scores

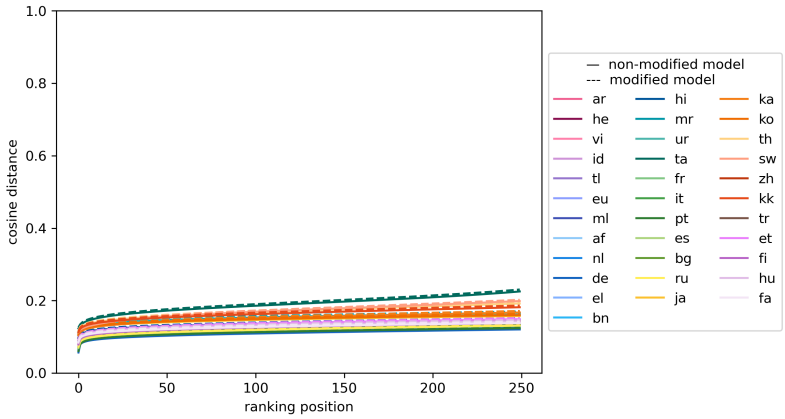
Another aspect to consider are the achieved cosine similarity scores associated with the sentence-rankings. Potential differences in this area are analyzed in order to understand how much the actual scores that control the rankings are affected by outlier removal. It is possible that, while producing different sentence-rankings, the scores themselves do not differ much, but it could also be the case that the cosine similarities achieved between `xx` and `en` sentences noticeably change after modifying the embeddings.

This section visualizes cosine distances (= 1 - cosine similarity) of the rankings *by language*. In the following plots, the x-axis denotes the position in the ranking, going from 1 to 250. The y-axis marks the magnitude of each ranking position's cosine distance. I.e. a distance of 0.1 at position 10 means that, for the respective language, on average the `en` sentence coming in rank 10 achieves a cosine distance of 0.1 with the `xx` sentence.

The following graphics show the cosine distance in rankings produced by the unmodified model vs. the ones produced by modified ones. This is once done for embeddings without dimension 588 and once without 306.[5]



(a) original vs. no 588



(b) original vs. no 306

Figure 5.7: Cosine distance scores of sentence-rankings between results of original and modified embeddings (XLM-R layer 8).

In the above figures, the dotted lines denote values achieved by the modified models, whereas the solid lines are produced by the original model. While the visualization considering the no-306 embeddings show that the cosine scores do not noticeably differ from the ones of the original model, the effect of removing 588 is drastic: it can be seen how running similarity search with the original model's embeddings squeezes all resulting cosine values into the range of about 0.0 to 0.2. As soon as the single dimension 588 is removed from the representations, the calculated cosines collectively rise up in magnitude and also spread apart.

This means that **by removing 588, the compared xx and en sentences on average become by far more distant from each other**, which may play a role in the increased Tatoeba task performance when this dimension is removed[6]. This may be helpful in the sense that the cosine distance to the correct en sentence, which should become top rank, is significantly more delimited from the rest of the candidates. This is not the case in the originally produced cosines, which are all comparatively close to each other. For removing 306, this effect is extremely slight.

[5]Note: The languages jv and te are omitted here since they exhibit less than 250 sentences, i.e. less than 250 ranks. Their results were however observed to be identical to the rest.

[6]For reference, the removal of 588 caused an increase of +2.64% in task performance.

5.4 Observations

This chapter has demonstrated a closer investigation of the previously gained insights on rogue dimensions with respect to the embeddings themselves, as well as the outliers' influence on Tatoeba task performance.

It was for one verified that *rogue dimensions consistently occur on all levels of embeddings*, i.e. they are not only a product of heavy averaging over all generated representations, but persistently appear on all sentence level embeddings across all languages. While also being present for English input, the *English counterparts in the Tatoeba parallel datasets are often not aligned well* with the other language's sentences in the outlier dimensions, which may disturb the similarity search of the task.

Concerning the factual influence of outliers on Tatoeba task performance, it was shown how their removal heavily affects the overall outcome of the similarity search, not only the chosen sentence pairings that are important to the accuracy. By removing especially dimension `588`, the *entire sentence-rankings are shuffled* as well as do the *cosine distances between the sentences significantly rise overall*, making `xx` and `en` sentences consistently more different from each other in this dimension.

The next chapter investigates the reason as to *why* these outliers' removal has such drastic effects on the similarity search task. It needs to be looked into the key characteristics of an outlier to draw connections to its impacts. Chapter 6 is therefore guided by the question: *what makes a rogue dimension?*

Chapter 6

What makes a rogue dimension?

This chapter is concerned with a more practical outlook on embedding outliers. While rogue dimensions have been identified and analyzed in the previous chapters, it is now aspired to establish the properties of such dimensions which make them influence task performance, especially in the context of cosine similarity search.

In the process, this shifts the previous definition of a rogue dimension, which is solely characterized by high magnitude, to the definition of *a dimension influencing task performance*. Such outliers can then be leveraged to make the right adjustments to a model's embeddings to achieve practical benefits.

6.1 Expanded set of outliers

Up until now, this thesis has treated an embedding dimension as rogue if its value surpasses the 3σ convention, as is likewise done in previous research. This however is a rather strict rule which may not be the final definition of embedding outliers.

For systematically identifying obviously rogue dimensions like `588` in XLM-R, this may be a good method. The hypothesis at this point is however that there could be other dimensions which also influence task performance when they are removed. Since so far the only criterion for a rogue dimension is its magnitude, the potential set of outliers is expanded according to this principle.

6.1.1 Method

To find more potentially rogue dimensions to analyze, the 3σ rule is replaced by a slightly different convention. The magnitude is still the key factor, but the dimension values are not compared to a certain threshold. Instead, the *top dimensions farthest from zero* are considered. In detail, this means the following:

For each of the 36 language-embeddings (as defined in 5.1.1), the top 10 dimensions are selected which are farthest from zero into either direction. These dimensions are then included into the new, expanded set of outliers for XLM-R[1].

Because not each of these dimensions occurs equally across all languages, the number of languages in which a dimension is an outlier is also reported:

dimension	`588`	`306`	`180`	`239`	`184`	`72`	`741`	`12`	`145`
# languages	36	36	36	36	36	36	26	23	22

dimensions	`89`	`151`	`694`	`259`	`267`	`459`	`723`	`152`	`720`
# languages	19	13	11	7	6	5	5	5	2

Table 6.1: Expanded set of outliers for XLM-R's layer 8 embeddings.

[1] As all previous experiments, this analysis also uses the model's layer 8 embeddings.

Looking at Table 6.1, there are 6 dimensions which were among the top-10 outliers for *all* languages, namely `588, 306, 180, 239, 184` and `72`, which are already more than established for XLM-R's layer 8 with the 3σ rule (see section 4.3.3). Taking the entire list of outliers produced by the current method, 18 dimensions are given as candidates to investigate in this section.

6.1.2 Effects on Tatoeba

The key question that is asked in this chapter is whether there are other dimensions that exhibit similar behavior to the rogue dimensions established before. The most obvious effect can be seen by looking at their influence on task performance. Since this thesis focuses on the Tatoeba similarity search, this section analyzes the newly found potential outliers on this task.

In detail, each of the dimensions from the new expanded set of outliers are zeroed out one by one as described in 4.4.2. The modified embeddings are then used for evaluation on Tatoeba.

dimension	`588`	`306`	`180`	`239`	`184`	`72`	`741`	`12`	`145`
score effect	+2.64	+0.24	+0.22	+0.76	+0.12	-0.06	+0.01	+0.04	+0.31

dimensions	`89`	`151`	`694`	`259`	`267`	`459`	`723`	`152`	`720`
score effect	+0.20	+0.57	-0.04	+0.15	+0.30	+0.34	+0.40	+0.97	+0.13

Table 6.2: Effects of removing new outliers on Tatoeba accuracy.

The results above show the influence on the accuracy achieved on the task after each of the respective dimensions is zeroed out. Note that the effects of `588, 306, 180` and `239` had already been established in section 4.4.2.

As can be seen, the extended list of potential outliers **did find other dimensions that also have an impact on task performance**: for instance, removing dimension `152`, which was *not* marked as rogue by the 3σ convention and was also among the top-10 outliers for *only 5 out of 36 languages*, has a positive effect of +0.97 on Tatoeba. Meanwhile removing dimension `306` which *was* established as rogue before and is an outlier for *all 36 languages*, produces an improvement of only +0.24. Similar observations can be made for a number of the dimensions above.

Since these new dimensions were reported to be rogue for only a fraction of the tested languages, it is natural to assume that the score improvements may be led by exactly those languages in the Tatoeba evaluation. This hypothesis is tested in the following.

6.1.3 Role of single languages

Because e.g. dimension `152` was established as an outlier by the new method for only 5 languages out of 36, it may be the case that its removal improves the similarity search by a big amount on exactly those languages, but does not have an effect for the other ones.

Therefore, before other possibilities are looked into, *it first needs to be verified whether the results are tied to specific languages only.* For this, the two new outliers `152` and `723` are picked out, which both were considered rogue in only 5 languages. The following compares for these two dimensions, whether there is a direct relationship between the languages that reported these outliers, and the per-language score improvements on Tatoeba.

language	score	outlier?
th	+3.83	no
kk	+2.08	no
jv	+1.95	no
tr	+1.79	no
ml	+1.75	no
...	...	no
pt	**+1.3**	**yes**
...	...	no
ru	**+0.79**	**yes**
...	...	no
fr	**+0.5**	**yes**
...	...	no
de	**+0.39**	**yes**
...	...	no
bg	**+0.09**	**yes**
sw	+0.001	no

(a) removing `152`

language	score	outlier?
ru	+1.29	no
he	+1.2	no
pt	+1.1	no
...	...	no
hu	**+1.09**	**yes**
...	...	no
kk	**+0.87**	**yes**
...	...	no
ka	**+0.67**	**yes**
et	**+0.6**	**yes**
...	...	no
eu	**+0.39**	**yes**
...	...	no
ml	-0.14	no
vi	-0.39	no
te	-0.43	no

(b) removing `723`

Table 6.3: Tatoeba improvements per language for new outliers `152` and `723`.

The above results show the score improvements on Tatoeba per evaluated language, once after zeroing out `152`, and once `723`. The sorting goes from the language with the biggest improvement to the lowest. The rows in bold font mark the languages for which the respective dimension was reported to be rogue in the first place, see section 6.1.1.

As can be seen, the 5 respective languages that exhibited the two outliers are *not* the only ones that benefit from the dimensions' removal: in fact, there is a number of other languages which benefit even *more* from zeroing those dimensions out, although they did not report these dimensions as rogue in their embeddings.

This indicates that **the similarity search improvements of clipping the new outliers are not dominated by the languages that exhibited them** - their removal affects the performance on other languages just as well and even more.

6.1.4 Role of outlier magnitude

As it was shown that not the single languages exhibiting certain rogue dimensions are responsible for the overall score improvements, the factor that was crucial up until this point is the next to be assessed: *magnitude*. The 3σ convention, as well as the method of the expanded set of outliers of section 6.1.1, rely on the magnitude of each embedding dimension to establish it as being rogue or not.

The question which in general needs to be addressed then, is *whether there is a linear relationship between a dimension's magnitude and its effects on the embeddings*, or in other words, whether the bigger a dimension's value is, the more a task like Tatoeba would benefit from its removal.

For insights on this matter, each of the outliers established in 6.1.1 are compared by their magnitudes and the score improvements they caused respectively. The following table takes one of Tatoeba's languages, `kk`, as an example for this. `kk` is also the language which benefited the most from the removal of the biggest outlier, `588`.

dim	best to worst task improvement	average dim value
588	+7.8	-14.9
152	+2.08	-0.76
151	+1.56	0.59
267	+1.39	-0.84
239	+1.2	-1.81
12	+0.87	1.33
723	+0.86	-1.02
184	+0.7	-1.31
89	+0.69	1.07
180	+0.52	1.99
694	+0.52	0.61
720	+0.52	0.81
145	+0.52	1.12
306	+0.34	3.03
72	+0.34	1.43
259	+0.34	0.96
741	+0.17	0.99
459	-0.002	-0.51

Table 6.4: Outlier dimensions compared by magnitude and task effect for the example Tatoeba language `kk`.

The above table shows that by removing e.g. dimension `588` from the embeddings, the accuracy of the Tatoeba task performance on language `kk` improves by +7.8%. At the same time, `588` exhibits the highest absolute value in the `kk` embeddings with an average of -14.9. This coincides with the hypothesis that "bigger" outliers cause bigger effects on the task.

However, the second biggest outlier, which is `306` with an average value of 3.03, improves the performance by only 0.34%, coming after *12 other dimensions which all exhibit smaller values*. Removing dimension `152` causes the second biggest increase in accuracy, but magnitude-wise it lies at only -0.76 on average. It had not even been considered an outlier for this language (see Table 6.3).

The magnitude of a dimension does therefore not equal big effects in terms of e.g. improved task performance when removed. The reason why this is important depends on the perspective one has on outliers and what one plans to do with them: for example, if the Tatoeba performance of XLM-R needs to be improved, and it is known that removing outliers in the model's embeddings is beneficial for the task, it is not the best to define outliers purely by their magnitude. A dimension with significantly high magnitude such as `588` may have such effect, but, as the results show, there can exist other, low magnitude dimensions which also produce similar results.

6.2 Search for outlier criteria

Based off of the previous section's outcome, the central question of this chapter becomes apparent, namely: *what makes a rogue dimension?* It is reasonable to characterize an outlier by simply being a dimension with exceptionally high magnitude. This is a valid theoretical definition.

If however the intent is to *make use* of outliers, like improving task performance by removing the right dimensions, other criteria may define a rogue dimension better. In the context of this thesis, the central task for evaluation was chosen to be the Tatoeba similarity search. In the following, outliers are treated as a means to improve XLM-R's performance on this task. To find out which characteristics a dimension leading to improvement holds, the next analyses are performed.

6.2.1 Anisotropy

The first point that is looked into is anisotropy, since magnitude-wise outliers have already been connected to this phenomenon by previous research: "The dimensions which drive anisotropy are centered far from the origin relative to other dimensions" [Timkey and van Schijndel, 2021, p. 4529].

For reference, anisotropy means that *randomly selected word representations are highly similar to each other*. Removing the dimensions associated with anisotropy causes the embedding spaces to become by far more isotropic (see also section 3.2).

Knowing the above, it is plausible to assume that a) outliers of high magnitude may likewise cause anisotropy in XLM-R's embeddings, and b) removing these outliers is benefiting similarity search in the Tatoeba task because anisotropy is reduced by that.
In order to draw further conclusions, it needs first be verified whether anisotropy is present in XLM-R's sentence embeddings in the first place. For this, the same procedure as introduced by Timkey and van Schijndel [2021] is applied, which first calculates the overall expected anisotropy in a given embedding space (i.e. how similar randomly chosen representations are expected to be), and then establishes the top embedding dimensions contributing to the phenomenon. For a detailed mathematical description of this method, see section 3.2.3.

Because in the case of Tatoeba, cosine similarity is always calculated between `xx` and `en` sentences, it is in the following investigated how similar a randomly chosen `xx` sentence and a randomly chosen `en` sentence are expected to be. If it is found that on average, any such two sentences are highly similar to each other, this is harmful for the similarity search task, since only the correct pairings should have a high cosine similarity.

For each of the 36 Tatoeba `xx-en` language pairs, 10,000 random pairs of `xx` and `en` sentence representations generated by XLM-R's layer 8 are sampled. Based off of their cosine similarities it is through the above mentioned method calculated, how anisotropic these embeddings are and which dimensions are the top contributors to the problem.

`kk-en`
Estimated anisotropy: 0.902

dimension	contribution
588	0.836
306	0.032
180	0.016
239	0.016
184	0.010

`ja-en`
Estimated anisotropy: 0.911

dimension	contribution
588	0.839
306	0.044
239	0.019
180	0.011
184	0.010

`de-en`
Estimated anisotropy: 0.927

dimension	contribution
588	0.833
306	0.032
239	0.028
184	0.018
180	0.009

`ar-en`
Estimated anisotropy: 0.917

dimension	contribution
588	0.844
306	0.031
239	0.015
180	0.015
184	0.014

Table 6.5: Anisotropy between `xx-en` sentences with top contributing dimensions (XLM-R layer 8 embeddings).

Table 6.5 shows four example language-pairs and their respective anisotropy results. For instance in the case of `de-en`, they are to be read as following: any randomly paired `de-en` sentence pair has an average cosine similarity of 0.927. The top contributing embedding dimension is `588`, which contributes around 83% to this expected cosine value.
For all 36 `xx-en` pairs, the anisotropy on average lies at 0.914, making the embeddings highly anisotropic: this means that during the Tatoeba similarity search between `xx` and `en` sentences, **any two sentences are highly similar to each other**.

The biggest contributor across all languages is dimension `588`, after which the dimensions `306, 239, 184` and `180` follow. Referring back to previously established outliers by essentially considering the dimensions with the biggest values, it can be observed that **the dimensions contributing to anisotropy are exactly the magnitude-wise outliers**. This aligns with the results of related research, where for multiple language models, the dimensions with the highest magnitudes were found to add the most to anisotropy (see 3.2.3).

6.2.2 Reducing anisotropy

Removing the above *anisotropy-related outliers* `588, 306, 239, 184` and `180` is supposed to reduce the anisotropy in the embeddings. In section 6.1.2, these outliers have already been removed from the embeddings, after which the effect on Tatoeba performance was measured. This section now again zeroes out the dimensions, but instead measures how much more *isotropic* the representations become. The average anisotropy of the unmodified embeddings was reported to be **0.914**.

removed dim	`588`	`306`	`239`	`184`	`180`
score effect	+2.64	+0.24	+0.76	+0.12	+0.22
anisotropy	*0.643*	*0.911*	*0.913*	*0.913*	*0.913*

Table 6.6: Effects of removing anisotropy-related outliers on anisotropy and Tatoeba scores.

The results in Table 6.6 show, how much anisotropy reduces after each of the dimensions is zeroed out. It is observed that, without `588`, the average cosine similarity between random embeddings drops to 0.643 from the initial 0.914. This does not make the embeddings isotropic, however it significantly reduces the problem. This result is also in line with the effects on cosine distances reported in Figure 5.7, where overall the sentences became by far more distant from one another, once `588` is discarded.

Removing all other dimensions, which have also been identified as contributing to anisotropy, does not have a similarly large effect. This is however attributed to the fact that, percentage-wise, they do not contribute nearly as much to the phenomenon as `588` (see Table 6.5).

Overall, there could be observed no linear relationship between decrease of anisotropy and benefits for similarity search: for example, removing `239` improves the Tatoeba task by +0.76, while `184` achieves an increase of only +0.12. Both of their removals however have identically small effects on anisotropy.

Nonetheless, the biggest outlier `588` **has a clear influence on the embeddings' anisotropy**: as long as it is present, all compared sentence embeddings are almost identical to each other. By removing the dimension, their similarity decreases by 0.271 which could not be achieved by any other single (!) dimension.

What all of the above dimensions have in common however, is that they could be identified as outliers in the sense of the similarity search task by analyzing anisotropy: all top-contributors to anisotropy are simultaneously beneficial for removal for Tatoeba, whether the effect is big or smaller.

6.2.3 Analyzing parallel data

The previous sections established the group of *anisotropy-related* outliers: these are dimensions which contribute to the embeddings' anisotropy, at the same time exhibit values of high magnitudes, and are beneficial for Tatoeba when removed.

Referring back to section 6.1, dimensions other than these had been established, whose removal likewise leads to task improvement. However, these are neither associated with high magnitudes, nor do they contribute to anisotropy. The question therefore remains, which other criteria may indicate such task-improving outliers.

Searching for further insights, the existence of Tatoeba parallel sentences is utilized. The following analysis is led by the idea that sentences, which are translations of each other, should ideally exhibit high cosine similarities.

Therefore firstly, the cosines between all `xx-en` parallel sentences are calculated. In order to then find dimensions that are influential to the score, *each of the 768 dimensions is zeroed out one by one*, each time re-calculating cosine similarity.

By doing this it can be observed, which removed dimension improves the similarity of the parallel data. Since parallel sentences *should* be as similar as possible, these dimensions may be good candidates for removal before the Tatoeba task: *if the correct pairs are made more similar, this may help find the correct translation pairs in Tatoeba.*

positive effect

```
267: 0.0008459803130891588, 459: 0.0007256915171941122, 152: 0.0006728818019231161,
 151: 0.0006419519583384196, 12: 0.0004984868897332086, 63: 0.0004864020480049981,
145: 0.0004662424325942993, 723: 0.00042956901921166316, 266: 0.0003915992048051622,
                                728: 0.0003851883941226535,
```

⋮

```
[...]
```

negative effect

```
 237: 8.386042382982043e-06, 101: 3.299779362148709e-06, 89: -4.872679710388184e-06,
72: -6.183816326989068e-05, 239: -0.00014217860168880888, 741: -0.00018046961890326606,
 180: -0.00021540621916453043, 306: -0.0011701716317070855, 588: -0.17203106813960606
```

Figure 6.1: Effect on parallel data cosine similarities after removing each of the 768 embedding dimensions.

Figure 6.1 shows the effect on the cosine similarity between parallel sentences after removing each of the embedding dimensions, averaged over all sentences and languages of the Tatoeba dataset. The list includes each dimension with the amount by which similarity was improved/worsened. It is sorted from dimensions causing the biggest improvement, to the ones causing the opposite.

It is observed that the removal of most dimensions causes a small increase in the cosine similarity. The dimensions at the top of the list however include many dimensions reported in the expanded set of outliers of section 6.1 – specifically, the ones not associated with anisotropy or universally big magnitudes: these include dimensions `267, 459, 152, 151, 12, 145` and `723`[2].

[2]Other dimensions from Table 6.1 may also lie near the top. It is chosen to only display the top-10 of the list here.

Simultaneously, the dimensions reported at the end of the list, i.e. whose removal caused the parallel data cosines to *decrease*, are observed to include the previously identified anisotropy-related outliers. This section's experiment therefore seems to identify two groups of outliers, which is further discussed in the following.

6.3 Established outlier groups

Searching for criteria able to characterize an outlier in the sense that its removal benefits the similarity search task, the previous section seems to have identified two groups of them.

After collecting the effects on parallel data cosine similarity of each of the 768 dimensions' removal, the ones improving and the ones worsening the similarity are pointed out, marked as green and red respectively in Figure 6.1. Looking at the actual dimensions these two groups include, it is observed that **the majority of them matches the outliers whose removal benefits Tatoeba**:

dimension	588	306	180	239	184	72	741	12	145
score effect	+2.64	+0.24	+0.22	+0.76	+0.12	-0.06	+0.01	+0.04	+0.31

dimensions	89	151	694	259	267	459	723	152	720
score effect	+0.20	+0.57	-0.04	+0.15	+0.30	+0.34	+0.40	+0.97	+0.13

Table 6.7: Effects of expanded magnitude-outliers on Tatoeba, but marked the dimensions identified by parallel data analysis. Green: Removal improves parallel data cosine. Red: Removal worsens parallel data cosine.

6.3.1 Anisotropy-related outliers

The method proposed in the previous section 6.2.3 produced dimensions which decrease the similarity between parallel sentences when zeroed out from the embeddings. Interestingly, this group includes the dimensions contributing to anisotropy as established in 6.2.1, i.e. the dimensions `588, 306, 239` and `180`. The only one missing is `184`.
These are in the following referred to as *anisotropy-related outliers*. What defines them, is a) their association with the embeddings' anisotropy, b) their magnitudes[3] and c) their removal being beneficial for the similarity search task.

The reason as to why the method in 6.2.3 has identified this group through them *decreasing* parallel data similarity when removed, can be explained:

As these dimensions are contributors to anisotropy, they "inflate" cosine similarity in general, i.e. all sentences are more similar to each other than they should be. By removing them, the cosine similarity level, i.e. anisotropy, is reduced. Reducing anisotropy however helps the task of similarity search, since now not any random sentences are more similar than desirable.
This is the reason why these dimensions' removal on one hand *worsens* parallel data similarity, but then on the other hand *helps* with similarity search.

Referring back to the "red group" identified in the previous section, there are three dimensions which were not specifically associated with anisotropy: `72, 741` and `89`. These are likewise *not* in the following included in anisotropy-related outliers for the next reasons:

[3]Clarification: "magnitude" here means that they were universally, across all 36 Tatoeba languages, reported to be among the biggest dimensions. For details refer to 6.1.1.

72, while worsening parallel data similarity when removed, also worsens similarity search results when removed – its removal therefore simply seems to generally have a negative effect. 741 has only a very minimal effect on task performance and is therefore also disregarded. 89 has the smallest negative effect on parallel data out of the entire red group (see Figure 6.1) and may have ended up there by accident, i.e. through the process of heavy averaging over all sentences and languages. Regardless, it does lead to a considerable improvement on Tatoeba. For formality reasons, since it does not pass criteria a) and b), it is not included in the group.

Dimension 184 *is* related to anisotropy, it has simply not been identified by the method of 6.2.3. Since it still passes criteria a), b) and c), **the final group of anisotropy-related outliers consists of:** 588, 306, 239, 184 **and** 180.

6.3.2 Similarity-harming outliers

The "green group" has likewise reported dimensions which have previously already been identified as task-beneficial outliers, namely 12, 145, 151, 267, 459, 723 and 152. Here, in contrast to the "red group", their removal *improves* the cosine similarities of parallel data.

The reason as to why these dimensions are also outliers is different from the anisotropy-outliers, but more staightforward: since their removal causes parallel sentences to become more similar to each other, which is desirable, this should likewise make it easier to find those correct translation pairs during the similarity search of the task.
This implies that the *presence of these dimensions may be somewhat harmful* to the representational quality of the sentence embeddings. The fact that specifically *parallel* sentences become closer in their representations when certain dimensions are absent, indicates that those dimensions may e.g. not encode the feature they represent properly. Because they essentially harm the similarity of sentences, which are supposed to be similar, this group is in the following referred to as *similarity-harming outliers.*

Apart from the dimensions of this group whose removal already was shown to be beneficial for the Tatoeba task, the parallel data method identified other ones, which had not been considered before. These are 63, 266 and 728. According to the above reasoning, these dimensions' removal could potentially also improve the Tatoeba task. Testing this, the following effects could be observed: 63: +0.16, 266: +0.22, and 728: +0.24. This means, that the proposed method has **successfully identified new task-improving outliers**, which were not found before by any other method. Subsequently, **the similarity-harming outliers consist of:** 12, 63, 145, 151, 152, 266, 267, 459, 723 **and** 728.

Concluding the search for outliers whose removal improves similarity search, the following dimensions are being regarded as beneficial:

Anisotropy-related

dim	588	306	239	184	180
score	+2.64	+0.24	+0.76	+0.12	+0.22

Similarity-harming

dim	12	63	145	151	152	266	267	459	723	728
score	+0.04	+0.16	+0.31	+0.57	+0.97	+0.22	+0.30	+0.34	+0.40	+0.24

Table 6.8: Outlier groups with their respective effects on the Tatoeba task.

6.4 Counter-check on BUCC task

The outlier-groups established above were identified and tested through the Tatoeba framework, i.e. on Tatoeba sentences and the Tatoeba similarity search task. In order to now verify, whether the removal of these dimensions can generally be beneficial for cosine-based similarity search, another sentence retrieval task from the XTREME benchmark is considered: BUCC [Zweigenbaum et al., 2018].

6.4.1 Task and Data

The third BUCC (Building and Using Comparable Corpora) shared task focuses on the retrieval of parallel sentences between monolingual corpora. While Tatoeba searches *parallel* corpora, i.e. each language in a pair has the same number of sentences and all of them have a corresponding translation in the other language, BUCC uses *comparable* corpora. Here, there is only a certain number of parallel sentences, as the corpora are unaligned and therefore not every sentence from language A can be paired with one from language B.

The BUCC dataset is made up of Wikipedia articles, which make up the monolingual corpora, and News Commentary[4], which are aligned parallel sentences that are artificially inserted into the monolingual data. They are then treated as the gold standard of sentence pairs that should be identified. Although there may be other valid parallel sentence pairs in the corpora, they are not accounted for during evaluation. [Zweigenbaum et al., 2018]

Similarity search is performed on four language pairs: `de-en`, `fr-en`, `ru-en` and `zh-en`, where the available data is split up into sample, training and test set as following:

Pair	Sample (2%)			Training (49%)			Test (49%)		
	fr	en	gold	*fr*	en	gold	*fr*	en	gold
de-en	32593	40354	1038	413869	399337	9580	413884	396534	9550
fr-en	21497	38069	929	271874	369810	9086	276833	373459	9043
ru-en	45459	72766	2374	460853	558401	14435	457327	566356	14330
zh-en	8624	13589	257	94637	88860	1899	91824	90037	1896

Figure 6.2: BUCC 2018 corpus statistics: number of monolingual sentences (*fr*, en) and of parallel pairs (gold) for each split and each language pair. The *fr* column stands for the non-English language in each pair. [Zweigenbaum et al., 2018, p. 2]

The BUCC task is carried out in the context of the XTREME benchmark using XLM-R's layer 8 embeddings. Because the gold labels are only made available for the *Sample* split, this part of the data is evaluated on. As for the similarity measure, cosine similarity is applied as in Tatoeba. Therefore the only differences between the two tasks are the type of corpora (parallel vs. comparable), the number of language pairs (36 vs. 4) and the actual sentences.

6.4.2 Baseline performance

In order to compare the impact of the outlier dimensions on BUCC, the task first needs to be evaluated using the unmodified sentence embeddings, which is done in this section. The performance of this task is evaluated with Precision, Recall and F1-score with the regular formulas [Zweigenbaum et al., 2018, p. 2].

[4]http://www.casmacat.eu/corpus/ news-commentary.html

	P	R	F1
de-en	89.8	76.6	82.7
fr-en	80.4	69.9	74.8
ru-en	89.8	75.5	82.0
zh-en	64.8	63.8	64.3
average	**81.2**	**71.4**	**75.9**

Table 6.9: Baseline results on BUCC using XLM-R layer 8 embeddings.

Though there are some discrepancies between the languages, i.e. zh-en generally performing worse than the others, overall XLM-R manages to produce reasonably good results. It is in the following examined whether the removal of the outlier dimensions established for Tatoeba are likewise beneficial for the performance on BUCC.

6.4.3 Removing outliers

The dimensions established as *anisotropy-related* and *similarity-harming* outliers are tested for their effects on the BUCC task by removing each of them from the embeddings separately. After each removal, the similarity search is run, showing by how much the modification has improved or worsened the performance. In the following, only the average scores are reported.

Anisotropy-related outliers

dim	588	306	239	184	180
F1	+1.4	-0.1	+0.6	+0.5	+0.2
P	+1.3	+3.2	+1.2	+2.9	-0.4
R	+1.4	-2.3	-0.3	-0.9	+0.7

Similarity-harming outliers

dim	12	63	145	151	152	266	267	459	723	728
F1	+1.2	+0.4	+0.9	+0.7	+0.2	+0.1	+0.3	+0.1	+0.0	+0.3
P	+1.6	-0.3	+2.1	+1.3	+0.5	+0.1	+0.7	+3.3	+0.4	+2.5
R	+0.7	+0.9	-0.2	+0.0	-0.1	+0.0	-0.1	-1.9	-0.5	-1.5

Table 6.10: Effects of removing outliers established with Tatoeba on the BUCC task.

Looking at the above results, it can be observed that the overall most beneficial dimension to remove is 588, as it consistently improves F-score, Precision and Recall, where the F-score improves the most out of all considered dimensions. Moving further, this effect is not as consistent with all dimensions, as for example removing 306 actually worsens F1 slightly. At the same time, it improves Precision by a good amount, but also worsens Recall. This kind of discrepancy is found throughout both anisotropy-related and similarity-harming outliers' tables. Considering only F1 however, the removal of the above dimensions is shown to rise the score.

6.4.4 Discussion

This section has run a counter-check of the outliers established for Tatoeba by removing them for BUCC. Regarding the results in Table 6.10 it first needs to be pointed out that for one, the scores are averages over 4 languages. This means that one language-pair, like zh-en as mentioned in 6.4.2, can influence the average score rather heavily, as opposed to an average that is composed of 36 individual scores as in Tatoeba. Secondly, it plays an important role that the gold pairs for the datasets,

i.e. the correct translation pairs that are to be identified, may not be the only ones present – the similarity search may match sentences that are also translations, however occurred naturally in the unaligned, monolingual data parts of the datasets and were therefore not taken into account in the gold pairs. This case specifically is addressed by the authors as increasing the amount of false positives, which is why the Precision of the results may be underestimated [Zweigenbaum et al., 2018, p.2].

Keeping in mind how the above may decrease the individual scores of Table 6.10, it can still be observed how **the outlier groups established for Tatoeba are likewise beneficial for BUCC** when removed. Considering the F-score as the main metric, nearly all dimensions have shown an improvement. This may be an indicator that zeroing out the anisotropy-related and similarity-harming outliers is **generally beneficial for cosine-based similarity search tasks**[5].

6.5 Further improving similarity search

Led by the question "what makes a rogue dimension?", this chapter has viewed outlier dimensions as a way to improve the performance of similarity search. A dimension was considered rogue not solely by its magnitude, but by its ability to be beneficial for task performance, when removed. This has produced groups of dimensions which fulfil exactly this requirement.

To finish this chapter, further methods of embedding modifications, which may improve similarity search even more, are explored.

6.5.1 Multiple removed outliers

Up until this point, the established outlier dimensions were only removed in an isolated way from one another, i.e. the effects on similarity search were reported only for single dimensions.

Naturally, the next step would be to consider bigger groups of rogue dimensions at once, as this may improve task performance even more. In the following, different groups of outliers are zeroed out simultaneously, after which the Tatoeba task is evaluated on.

Anisotropy-related outliers

dimensions	accuracy improvement
two most anisotropy-associated (`588, 306`)	+2.81
all	**+5.74**

Similarity-harming outliers

dimensions	accuracy improvement
top-3 score-improving (`151, 152, 459`)	+1.87
all	**+3.77**

Both groups

dimensions	accuracy improvement
all outliers	**+9.74**

Table 6.11: Effects on Tatoeba after removing entire groups of outliers.

[5]Specifically regarding XLM-R's layer 8 embeddings.

For both outlier groups, a smaller subgroup was removed first, after which the entire group was also zeroed out. As for anisotropy-related outliers, the subgroup was chosen to be the two dimensions most consistently associated with anisotropy, `588` and `306`. The similarity-harming group first tests the top-3 score improving dimensions, `151, 151` and `459`.

While removing subgroups already shows good score improvements, the most benefit can be gained from removing the entire groups altogether: by doing this, Tatoeba performance gains of +5.74 and +3.77 can be achieved. For reference, the previously biggest gain, achieved by zeroing out the outlier `588`, reached an improvement of +2.64.

The so far best score on Tatoeba by removing outlier dimensions is attained by removing all 15 from both groups: this **improves performance by +9.74 and reaches an accuracy of 60.09%.**

An identical experiment is in the following performed for BUCC:

Anisotropy-related outliers

dimensions	F1	P	R
all	+1.9	+1.8	+1.8

Similarity-harming outliers

dimensions	F1	P	R
all	+3.5	+4.2	+2.9

Both groups

dimensions	F1	P	R
all outliers	**+6.2**	**+8.2**	**+4.8**

Table 6.12: Effects on BUCC after removing entire groups of outliers.

As can be seen, zeroing out entire groups of outliers is likewise more beneficial for BUCC. While the previously biggest reached improvement was +1.4 on F1 (see Table 6.10), **a gain of +6.2 can be achieved by removing all outliers**. Between the two outlier groups, interestingly removing the similarity-harming outliers is almost two times more beneficial than removing the anisotropy-related ones. With Tatoeba, it is the opposite.

This section has explored the maximum of what can be reached by zeroing out outlier dimensions from the embeddings. The score may get even slightly better, if rescaling is performed instead (as done in section 4.4.3). As the improvements of rescaling in comparison to removing are rather minimal, it is not tested here, but could potentially push the scores a little further.

6.5.2 Fine-tuned embeddings

Another possibility of embedding modification, which does not directly target outlier dimensions, is to use representations from a fine-tuned model. For this, it was chosen to fine-tune XLM-R on two of the XTREME benchmark's cross-lingual sentence classification tasks, PAWS-X [Yang et al., 2019a] and XNLI [Conneau et al., 2018]. These tasks were considered suitable for fine-tuning on, as their training procedures may likely lead to embeddings performing better on `xx-en` similarity search.

PAWS-X For PAWS-X, which aims to detect whether two sentences are paraphrases, a model is trained on the PAWS training set [Zhang et al., 2019], consisting of approximately 49,000 labeled *English* sentence pairs. By translating a part of the

original English dataset into six other languages, Spanish, French, German, Chinese, Japanese, and Korean, the development and test sets for PAWS-X are created, offering around 4,000 sentence pairs to classify for each language[6]. [Yang et al., 2019a]

XNLI As for XNLI, which requires to decide if sentence B is an entailment, a contradiction or neutral to sentence A, training is performed on the *English* MultiNLI training set [Williams et al., 2018b] consisting of over 392,000 examples. The development and test sets encompass the languages English, French, Spanish, German, Greek, Bulgarian, Russian, Turkish, Arabic, Vietnamese, Thai, Chinese, Hindi, Swahili and Urdu, offering around 7,500 human-annotated sentence pairs[7] per language [Conneau et al., 2018]. Overall, XNLI is more data intensive than PAWS-X, as it includes a magnitude more sentences as well as more languages, including low-resource ones.

For this section's analysis regarding the improvement of Tatoeba's similarity search, XLM-R is fine-tuned on both of these tasks using the following parameters.

task	epochs	learning rate	batch size
PAWS-X	5	2e-5	8
XNLI	1	2e-5	8

After each training has finished, the fine-tuned XLM-R model's layer 8 is used to produce sentence embeddings for the sentences of Tatoeba. With the help of these "fine-tuned embeddings", the task's similarity search is then performed. The following results could be achieved.

fine-tuned on	Tatoeba accuracy
PAWS-X	59.90 (+9.55)
XNLI	**73.39 (+23.04)**

Table 6.13: Results on Tatoeba using fine-tuned embeddings.

As shown in Table 6.13, the **embeddings produced by the fine-tuned models are highly beneficial for the similarity search**. As expected, XNLI is the most effective even though only trained for 1 epoch, as it uses more data overall. Training on this task improves the performance of the Tatoeba task by +23.04%, which could not be achieved by removing outliers alone.

This raises the question, whether the outlier dimensions disappeared during the training process: it may be the case that during fine-tuning, certain dimensions became less important for the tasks at hand, making them shrink. This concerns dimensions which can be detected magnitude-wise.

The following shows the average sentence embeddings produced by the fine-tuned XLM-R models' 8^{th} layers, using the previously employed "3σ rule" to visualize dimensions exceeding the norm with a greyed out area. The process of obtaining the average representations as well as the 3σ convention are described in section 4.3.

[6] 2,000 dev and 2,000 test.
[7] 2,500 dev and 5,000 test.

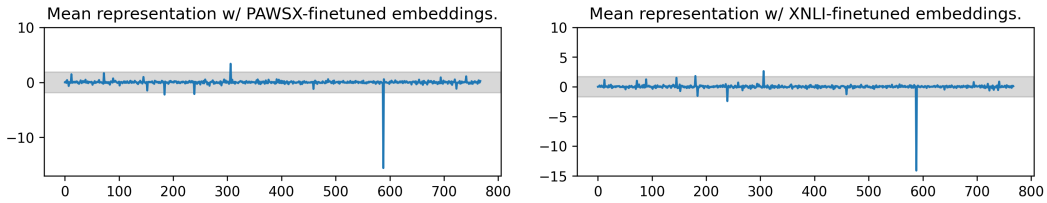


Figure 6.3: Mean representations of PAWS-X and XNLI-finetuned sentence embeddings.

Interestingly, the **magnitude-wise outliers established previously are still present**: the dimensions exceeding the grey 3σ area are `184, 239, 306` and `588` for PAWS-X and `180, 239, 306` and `588` for XNLI. Since with these embeddings, a significant improvement on the similarity search task could still be achieved, the magnitude-wise outliers don't seem to be the biggest reason for the mediocre performance of XLM-R on Tatoeba.

The embeddings were instead possibly modified in some other way during the fine-tuning process, which led to more significant improvements than removing these dimensions. Possibly however, removing all previously established rogue dimensions from the fine-tuned embeddings, i.e. anisotropy-related and similarity-harming outliers, may produce sentence representations even more suited for a good result on similarity search.

Chapter 7

Discussion

This thesis has investigated the topic of rogue embedding dimensions occurring in the multilingual XLM-R model. Specifically, their role in the performance of sentence similarity search, which directly uses these embeddings, has been examined.
The main findings as well as possible future work are laid out in the following.

7.1 Findings

7.1.1 Outliers across model layers

The XLM-R_{base} model consists of 12 layers, which can all be used for obtaining embeddings. Generating average sentence representations taking into account all 36 languages of the Tatoeba dataset, it has been established which dimensions are magnitude-wise outliers for each layer (see 4.3.3). To our knowledge, no other work has yet reported the exact dimensions that are rogue for the XLM-R model.

Examining the findings, it is observed how the most outstanding outliers change as the number of layers progresses. While dimension `588` is most prominent until layer 9, after this point `741` begins to rise in value, becoming the dimension with the biggest magnitude in the final layer.

7.1.2 Influence on similarity search

Investigating the main objective of this work, i.e. the influence of outlier dimensions on similarity search, the Tatoeba task was evaluated on using XLM-R's sentence embeddings from layer 8. Analyzing the effect of outlier dimensions established for this layer, it is found that their removal is beneficial for task performance. This is in contrast to their effect on other tasks, where previous research has reported a decrease in performance when such dimensions are discarded (see 3.3.2).

This indicates that such outliers may be useful for the model when performing certain tasks, as they do not seem to disappear during fine-tuning (as likewise observed in 6.5.2), but are not helpful in the sense of representational quality of the resulting embeddings.

7.1.3 Redefining rogue dimensions

In the search for the key characteristics of dimensions, which harm the representational quality of sentence embeddings and therefore impair the similarity search, the concept of a rogue dimension has been redefined.

The most intuitive definition, which is likewise the one assumed in previous research, makes a dimension an outlier if its value significantly deviates from the norm, i.e. if it exhibits exceptionally high magnitude. While the removal of such outliers was found to improve similarity search, the reason for this is not rooted solely in the

dimensions' magnitude. It is rather the case that dimensions of high magnitudes are simultaneously the ones making the embeddings anisotropic. This reasoning is especially sensible in the context of similarity search, as an anisotropic embedding space makes all representations highly similar to one another, making the search for correct translation pairs more difficult. This defines the group of *anisotropy-related* outliers, whose removal makes the sentence embeddings more isotropic and through this improves the task performance.

Another group established to be harmful for similarity search was identified by analyzing the embeddings of parallel Tatoeba sentences. Since they are aligned translations of each other, their representations should ideally be as similar as possible. By removing each embedding dimension one by one and re-computing the cosine similarity between the sentences, it was identified which of the removed dimensions caused the biggest increases in parallel cosine similarity. These dimensions were found to neither be of exceptionally high magnitudes, nor being associated with anisotropy. They therefore constitute their own separate group of *similarity-harming* outliers.

7.1.4 Improving similarity search

Having identified two groups of dimensions which are rogue in the sense of being harmful to similarity search, this information has been leveraged to improve the performance on such tasks in a targeted manner. Removing all outliers from the sentence embeddings, before evaluating on the similarity search tasks Tatoeba and BUCC, score improvements of +9.74% and +6.2% could be achieved (see 6.5.1). This improvement is accomplished by targeting only 15 out of 768 embedding dimensions.

The research of this thesis has shown characteristics which define dimensions affecting cosine-based similarity search. These methods can in the future be used to identify such outliers for other models and layers in order to push their performance on similar tasks.

7.2 Future Work

7.2.1 Role of single languages and sentences

One point worth further investigations is how different languages and even different sentences influence outliers. As observed in Chapter 5, not all languages exhibit (magnitude-wise) outliers in the same dimensions, nor do they all reach the same values: in dimension `588`, one language's embeddings on average reached a value of around -16, while another language reported -14.4 (see 5.1.1).

Going even further, single *sentences* seem to be able to influence how pronounced the value in such dimensions becomes: the same dimension `588` has shown how, in a single language, one sentence can produce a value of almost -19, while another sentence produces -12.8.

There might be some underlying patterns that cause this to happen, i.e. certain language families or certain types of sentences scoring higher values in such dimensions. A systematic analysis of this matter could give valuable insights on the entire topic of rogue dimensions.

7.2.2 Other outlier criteria

This thesis has produced two groups of outliers for similarity search, which can be identified by either probing anisotropy or analyzing cosine similarity changes in

parallel data, if available. This does however not mean that these are the only dimensions able to affect task performance. In fact, Table 6.7 of section 6.3 shows, how there are a few dimensions, whose removal improves the Tatoeba results, however they do not fall under any of the two proposed categories.

An exhaustive list of such outliers could have been obtained by applying brute-force search over all embedding dimensions, i.e. zeroing out each of the 768 dimensions and running e.g. the Tatoeba task each time, in order to directly see, which dimensions' removal was beneficial. These dimensions could have then been analyzed on their properties, which may have revealed other criteria for task-affecting outliers.

Since this was not possible due to computational constraints, it is a possibility for future work to, for example, apply the above brute-force search method for obtaining other criteria.

7.2.3 Revisiting similarity-harming outliers

The similarity-harming outliers established in 6.3.2 are effective in improving task performance, however they do not exhibit a defining characteristic, as e.g. the other group of outliers associated with anisotropy does. Their removal improves the similarity between parallel data, implying that these dimensions are somewhat harmful to the embeddings' representational quality, but it remains unclear, *why* exactly that is.
Possibly they exhibit another underlying characteristic, which makes them undesirable to keep. Finding it, if such distinctive cause exists, may in turn reveal other outliers which the method proposed in 6.3.2 could not effectively capture.

7.2.4 Exploring different models and tasks

Lastly, a possibility for further research is to expand the insights of this work to other models, model layers, and tasks. For instance, it would be of interest to test the established outlier groups on tasks other than similarity search. In detail this means to remove anisotropy-related and/or similarity-harming outliers from XLM-R's layer 8 and perform inference on another NLP task. The outcome may be decreased performance (as reported for other outliers by previous research, see 3.3.2), or even improved performance as was seen with similarity search. Such investigations would show the established outliers to either be exclusive to similarity search, or in the other case universally affecting different tasks.

Naturally, the alternative outlier definition itself can also be extended to other models and tasks. It may well be the case for other language models and other NLP tasks that there exist certain dimensions, which are harmful to the performance. Identifying those can help improve the results by targeting the right embedding dimensions.

Bibliography

Galen Andrew, Raman Arora, Jeff Bilmes, and Karen Livescu. Deep Canonical Correlation Analysis. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1247–1255, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL `https://proceedings.mlr.press/v28/andrew13.html`.

Mikel Artetxe and Holger Schwenk. Massively Multilingual Sentence Embeddings for Zero-Shot Cross-Lingual Transfer and Beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610, 03 2019. doi: 10.1162/tacl_a_00288.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and Improving Bilingual Word Embedding Mappings with a Multi-Step Framework of Linear Transformations. In *AAAI*, 2018.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=r1xMH1BtvB`.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. XNLI: Evaluating Cross-lingual Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1269. URL `https://aclanthology.org/D18-1269`.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL `https://aclanthology.org/2020.acl-main.747`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16

Words: Transformers for Image Recognition at Scale. *CoRR*, abs/2010.11929, 2020. URL https://arxiv.org/abs/2010.11929.

Kawin Ethayarajh. How Contextual are Contextualized Word Representations? Comparing the Geometry of BERT, ELMo, and GPT-2 Embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 55–65, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1006. URL https://aclanthology.org/D19-1006.

Prakhar Gupta and Martin Jaggi. Obtaining Better Static Word Embeddings Using Contextual Embedding Models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5241–5253, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.408. URL https://aclanthology.org/2021.acl-long.408.

Katharina Hämmerl, Jindřich Libovický, and Alexander Fraser. Combining Static and Contextualised Multilingual Embeddings. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2316–2329, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.182. URL https://aclanthology.org/2022.findings-acl.182.

Harold Hotelling. Relations Between Two Sets of Variates. *Biometrika*, 28(3/4): 321–377, 1936. ISSN 00063444. URL http://www.jstor.org/stable/2333955.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization. *CoRR*, abs/2003.11080, 2020. URL https://arxiv.org/abs/2003.11080.

Olga Kovaleva, Saurabh Kulshreshtha, Anna Rogers, and Anna Rumshisky. BERT Busters: Outlier Dimensions that Disrupt Transformers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3392–3405, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.300. URL https://aclanthology.org/2021.findings-acl.300.

Laerd Statistics. Spearman's Rank-Order Correlation, 2018. URL https://statistics.laerd.com/statistical-guides/spearmans-rank-order-correlation-statistical-guide.php. Accessed: 2022-07-09.

Guillaume Lample and Alexis Conneau. Cross-lingual Language Model Pretraining. *CoRR*, abs/1901.07291, 2019. URL http://arxiv.org/abs/1901.07291.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019. URL https://arxiv.org/abs/1907.11692.

Ziyang Luo, Artur Kulmizev, and Xiaoxi Mao. Positional Artefacts Propagate Through Masked Language Model Embeddings. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5312–5327, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.413. URL https://aclanthology.org/2021.acl-long.413.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *CoRR*, abs/1609.07843, 2016. URL http://arxiv.org/abs/1609.07843.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013. URL http://arxiv.org/abs/1301.3781.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://aclanthology.org/D14-1162.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 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, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL https://aclanthology.org/N18-1202.

Telmo Pires, Eva Schlinger, and Dan Garrette. How Multilingual is Multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1493. URL https://aclanthology.org/P19-1493.

Giovanni Puccetti, Anna Rogers, Aleksandr Drozd, and Felice Dell'Orletta. Outliers Dimensions that Disrupt Transformers Are Driven by Frequency, 2022. URL https://arxiv.org/abs/2205.11380.

Alec Radford and Karthik Narasimhan. Improving Language Understanding by Generative Pre-Training. 2018.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. 2019.

Sara Rajaee and Mohammad Taher Pilehvar. An Isotropy Analysis in the Multilingual BERT Embedding Space. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1309–1316, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.103. URL https://aclanthology.org/2022.findings-acl.103.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *CoRR*, abs/1908.10084, 2019. URL http://arxiv.org/abs/1908.10084.

Braden Riggs and George Williams. The Role of Transformers in Natural Language Processing and Google's Revolutionary B.E.R.T., 2020.

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*, pages 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, 2017. URL https://arxiv.org/abs/1706.03762.

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*, pages 353–355, Brussels, Belgium, November 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. URL https://aclanthology.org/W18-5446.

Adina Williams, Nikita Nangia, and Samuel Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. 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 1112–1122, New Orleans, Louisiana, June 2018a. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://aclanthology.org/N18-1101.

Adina Williams, Nikita Nangia, and Samuel Bowman. A Broad-Coverage Challenge Corpus for Sentence Understanding through Inference. 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 1112–1122, New Orleans, Louisiana, June 2018b. Association for Computational Linguistics. doi: 10.18653/v1/N18-1101. URL https://aclanthology.org/N18-1101.

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. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*,

pages 3687–3692, Hong Kong, China, November 2019a. Association for Computational Linguistics. doi: 10.18653/v1/D19-1382. URL https://aclanthology.org/D19-1382.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. *CoRR*, abs/1906.08237, 2019b. URL http://arxiv.org/abs/1906.08237.

Yuan Zhang, Jason Baldridge, and Luheng He. PAWS: Paraphrase Adversaries from Word Scrambling. 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 1298–1308, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1131. URL https://aclanthology.org/N19-1131.

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. *CoRR*, abs/1506.06724, 2015. URL http://arxiv.org/abs/1506.06724.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. Overview of the Third BUCC Shared Task: Spotting Parallel Sentences in Comparable Corpora. In *Workshop on Building and Using Comparable Corpora*, Miyazaki, Japan, May 2018. URL https://hal.archives-ouvertes.fr/hal-01898360.

List of Figures

List of Tables

CD Contents

- The PDF version of this thesis
- All code associated with this thesis
- All graphics associated with this thesis
- Model checkpoints
- The Tatoeba dataset
- Sentence embeddings for Tatoeba data

Appendix

Appendix A

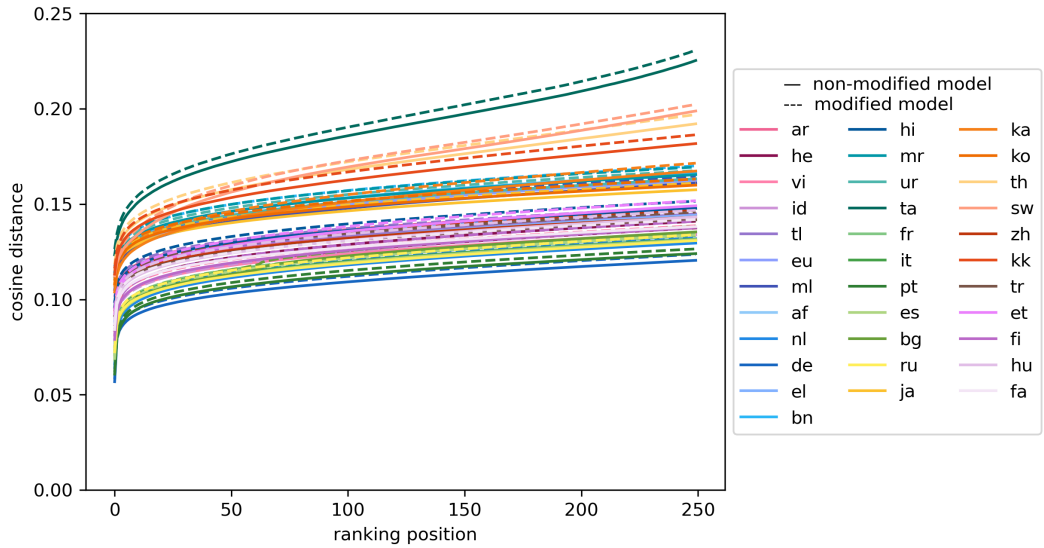


Figure 7.1: Cosine distance scores of sentence-rankings between results of original and no-306 embeddings – a closeup view. (Reference: section 5.3.2)

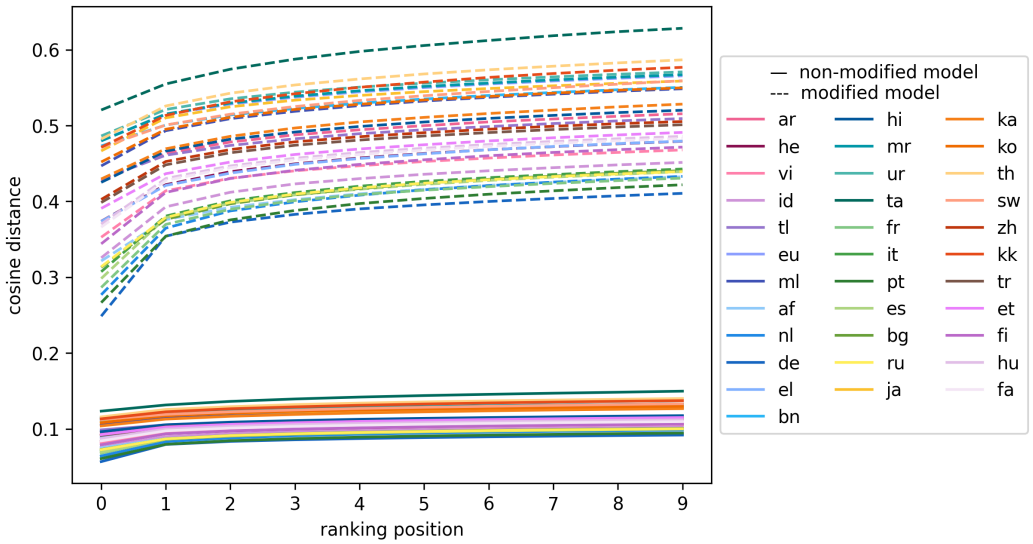
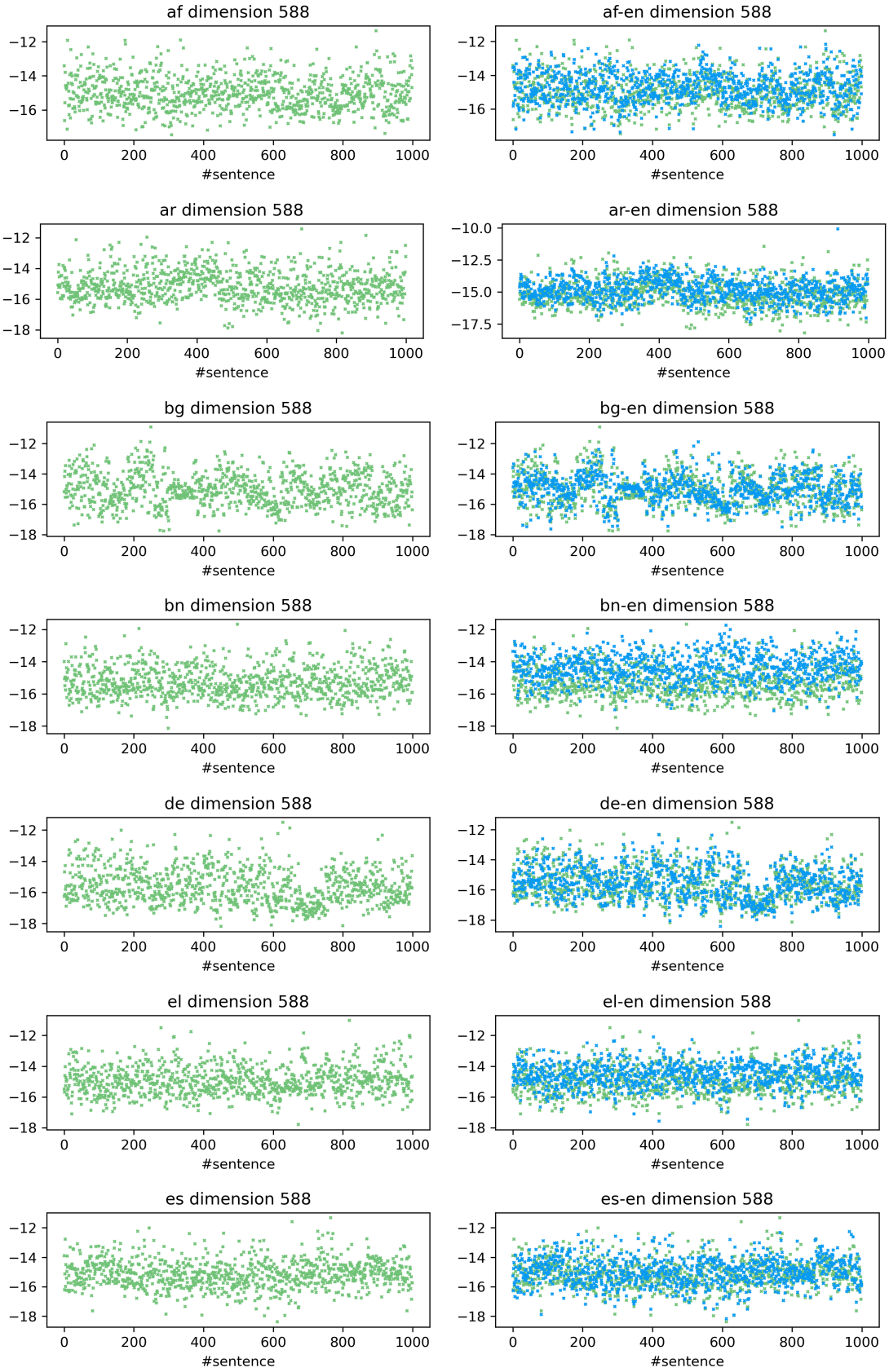
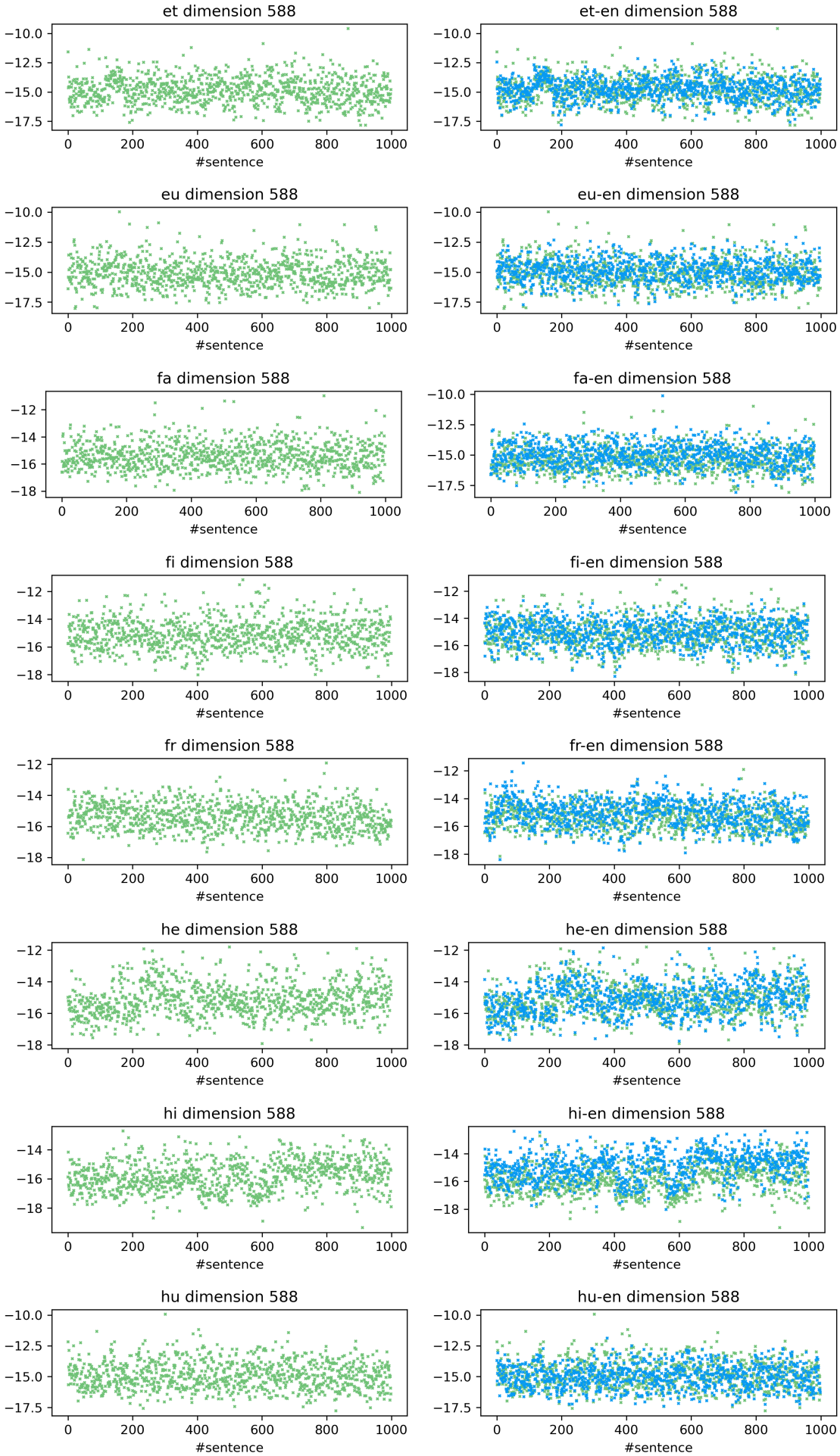


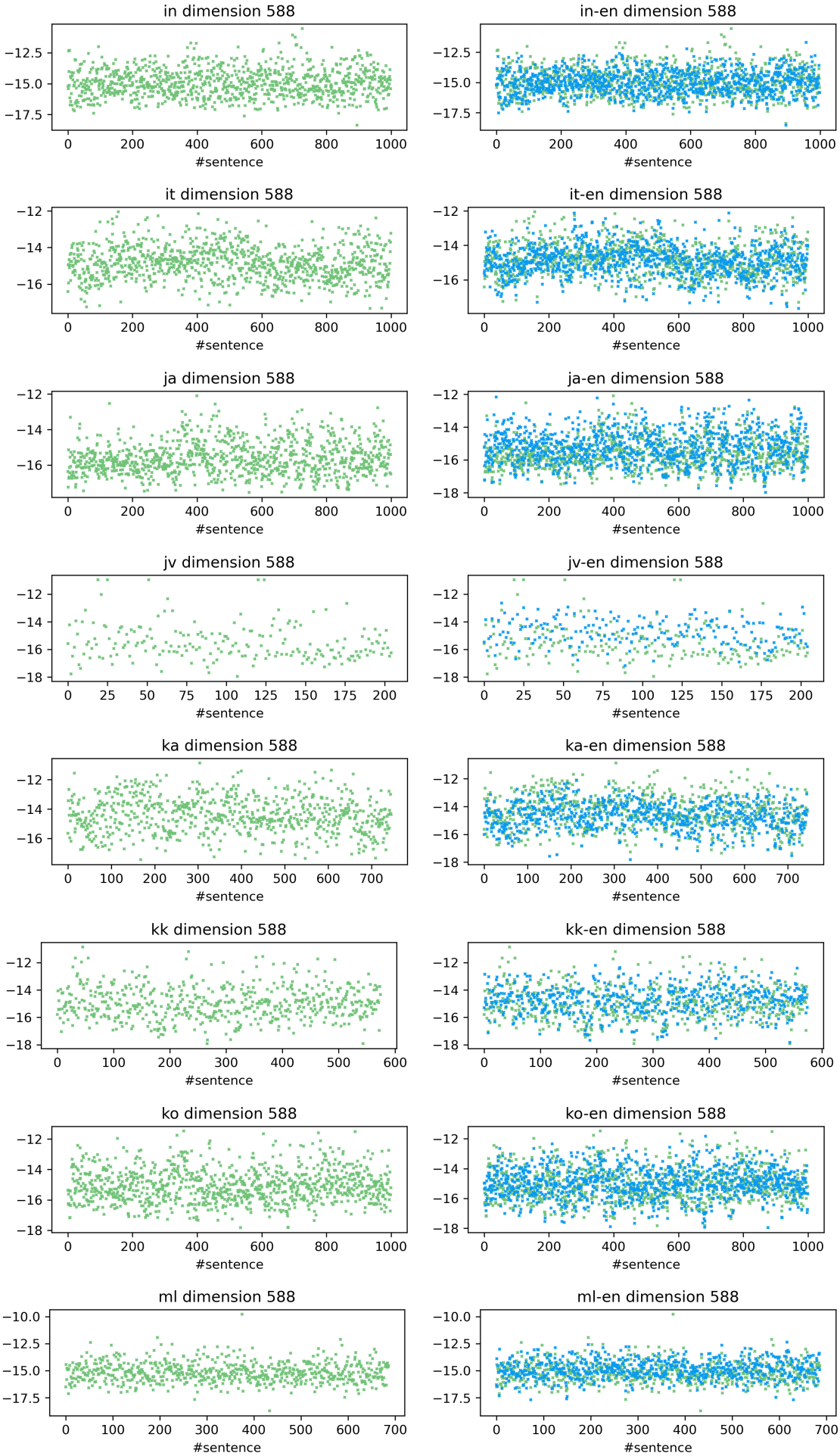
Figure 7.2: Cosine distance scores of sentence-rankings between results of original and no-588 embeddings – first 10 positions. (Reference: section 5.3.2)

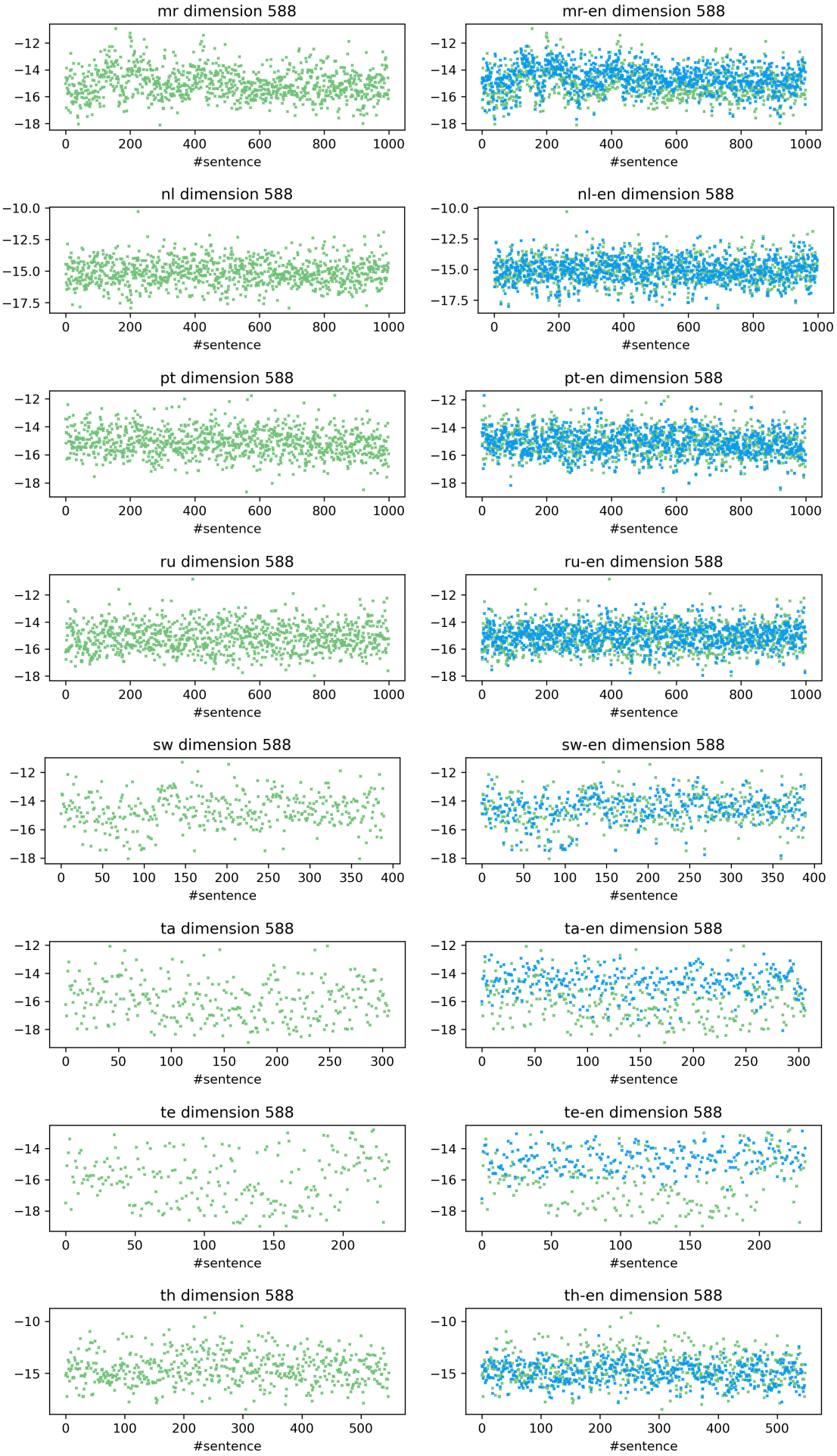
Appendix B

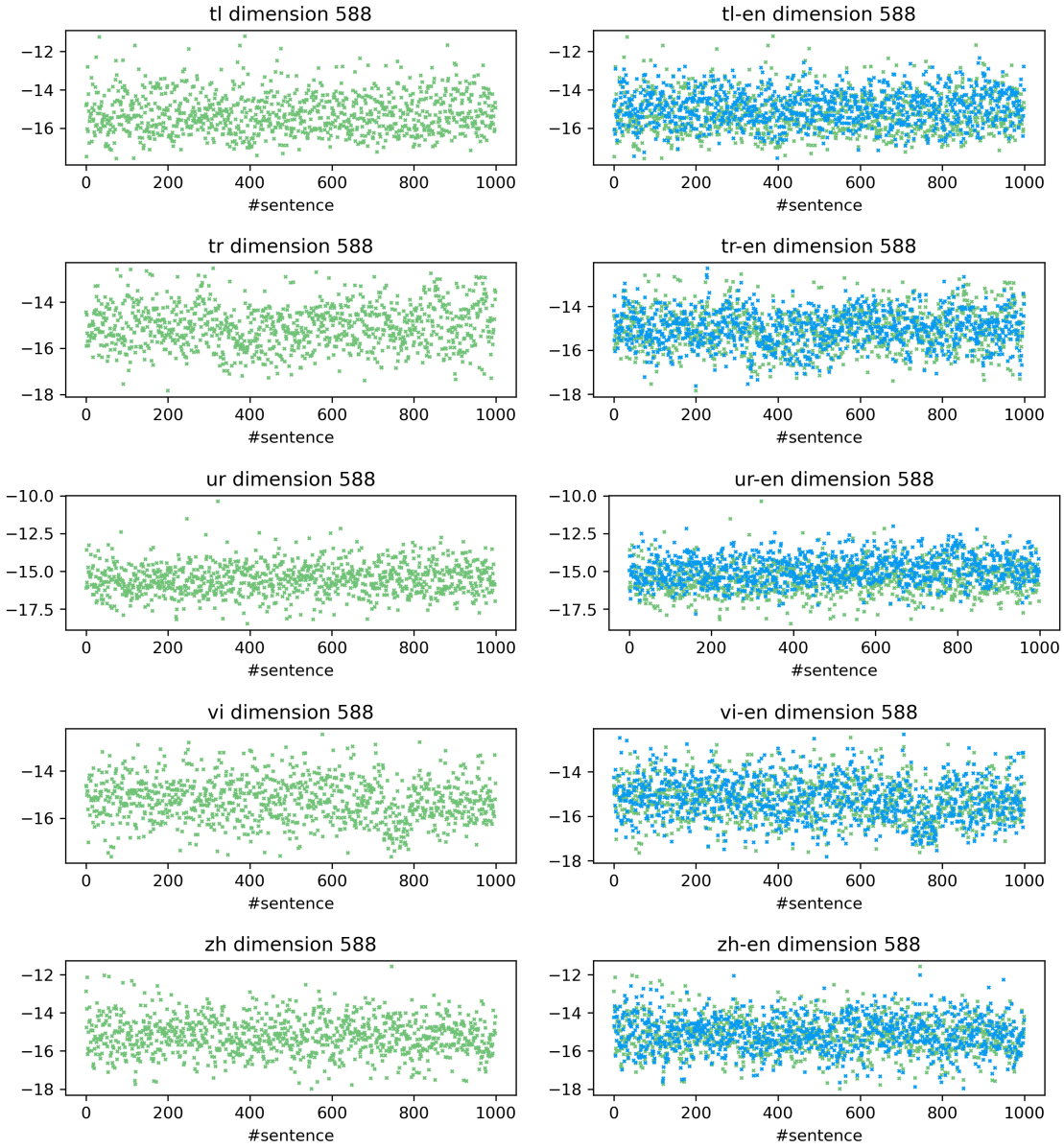
Sentences from all 36 Tatoeba ***xx-en*** *pairs: their values in dimension* ***588*** *once showing only language* ***xx****, once aligned together with their parallel* ***en*** *sentences.*




et dimension 588
et-en dimension 588
eu dimension 588
eu-en dimension 588
fa dimension 588
fa-en dimension 588
fi dimension 588
fi-en dimension 588
fr dimension 588
fr-en dimension 588
he dimension 588
he-en dimension 588
hi dimension 588
hi-en dimension 588
hu dimension 588
hu-en dimension 588
#sentence


in dimension 588
in-en dimension 588
it dimension 588
it-en dimension 588
ja dimension 588
ja-en dimension 588
jv dimension 588
jv-en dimension 588
ka dimension 588
ka-en dimension 588
kk dimension 588
kk-en dimension 588
ko dimension 588
ko-en dimension 588
ml dimension 588
ml-en dimension 588
#sentence


mr dimension 588
mr-en dimension 588
nl dimension 588
nl-en dimension 588
pt dimension 588
pt-en dimension 588
ru dimension 588
ru-en dimension 588
sw dimension 588
sw-en dimension 588
ta dimension 588
ta-en dimension 588
te dimension 588
te-en dimension 588
th dimension 588
th-en dimension 588
#sentence


tl dimension 588
tl-en dimension 588
tr dimension 588
tr-en dimension 588
ur dimension 588
ur-en dimension 588
vi dimension 588
vi-en dimension 588
zh dimension 588
zh-en dimension 588
#sentence