Static Word Embeddings for Sentence Semantic Representation

Takashi Wada Yuki Hirakawa Ryotaro Shimizu Takahiro Kawashima Yuki Saito
ZOZO Research
`{firstname.lastname}@zozo.com`

Abstract

We propose new static word embeddings optimised for sentence semantic representation. We first extract word embeddings from a pre-trained Sentence Transformer, and improve them with sentence-level principal component analysis, followed by either knowledge distillation or contrastive learning. During inference, we represent sentences by simply averaging word embeddings, which requires little computational cost. We evaluate models on both monolingual and cross-lingual tasks and show that our model substantially outperforms existing static models on sentence semantic tasks, and even surpasses a basic Sentence Transformer model (SimCSE) on a text embedding benchmark. Lastly, we perform a variety of analyses and show that our method successfully removes word embedding components that are not highly relevant to sentence semantics, and adjusts the vector norms based on the influence of words on sentence semantics.

1 Introduction

In natural language processing (NLP), it has been a major topic to learn fixed-length embeddings that represent sentence semantics. To achieve this goal, most existing methods fine-tune pre-trained language models (LMs) such as BERT (Devlin et al., 2019) using labelled data of various NLP tasks (e.g. natural language inference). Recent work further employs large language models (LLMs) such as LLaMA 3 (AI@Meta, 2024) as backbone models and shows that scaling up models can yield better performance (BehnamGhader et al., 2024; Lee et al., 2025). However, this approach requires a large amount of computational cost, making it difficult to process billions of sentences cost-efficiently or deploy models on resource-constrained devices such as smartphones. As such, it is crucial to explore more cost-efficient models that work well without GPUs, and yet this direction remains largely overlooked in the research community.

In this work, we propose new static word embeddings (**SWEs**) — both monolingual and cross-lingual ones — that can represent sentence semantics fairly well with little computational cost. Specifically, our method first extracts SWEs from a pre-trained Sentence Transformer (Vaswani et al., 2017; Reimers and Gurevych, 2019), and improves them with *sentence-level* principal component analysis (PCA), followed by either knowledge distillation or contrastive learning. During inference, we encode sentences by simply looking up SWEs and taking their average, which can be done comfortably with limited resources. Our main contributions are: (1) we propose new SWEs that substantially outperform existing SWEs on a semantic textual similarity (STS) task, and even surpass SimCSE (Gao et al., 2021) on Massive Text Embedding Benchmark (Muennighoff et al., 2023) in English; (2) we propose cross-lingual SWEs optimised for sentence representation, which achieve surprisingly good performance on translation retrieval even between distant languages (e.g. 94.6% in F1 scores on English–Chinese); and (3) we conduct a variety of analyses and provide new insights into how to obtain SWEs suitable for sentence representation. We will release our code and models at `https://github.com/twadada/swe4semantics`.

2 Related Work

Most recent text embedding models are built on pre-trained Transformer LMs. For instance, GTE (Li et al., 2023) fine-tunes BERT with contrastive learning on labelled data retrieved from various sources (e.g. STS data sets). Later, it is extended to the multilingual model called mGTE (Zhang et al., 2024), which is trained on both monolingual and cross-lingual data sets. LLM2Vec (BehnamGhader et al., 2024) scales up the backbone model and fine-tunes LLMs such as LLaMA with contrastive learning. Similarly, NV-Embed (Lee et al., 2025) fine-tunes Mistral-7B (Jiang et al., 2023) with contrastive learning and also trains an additional attention layer on top. While these models achieve impressive results, they are computationally expensive and inefficient for real-world applications. To mitigate this, several methods to train compact models are proposed (e.g. knowledge distillation (Hinton et al., 2015; Xu et al., 2023)), but they are mostly applied to Transformer models, which are designed with GPU acceleration in mind.

On the other hand, little attention has been paid to building cost-efficient models (esp. cross-lingual ones) that can be easily run on CPUs. Although such models inevitably lag behind heavy-weight models in accuracy, they offer significant advantages in cost and runtime efficiency, and can be applied to, for instance, quickly mining semantically similar content from billions of posts on social media. One traditional model in this direction is Sent2Vec (Pagliardini et al., 2018), which extends the CBOW algorithm (Mikolov et al., 2013a) and represents sentences by averaging static word and bigram embeddings. Very recently, although not published in papers, two static word embedding models called WordLlama (Miller, 2024) and Model2Vec (Tulkens and van Dongen, 2024) have been released from different GitHub projects (MIT license),[1] and demonstrate strong performance on sentence-level tasks in English. Based on their code and documentations, both models extract SWEs from Transformer models. Specifically, WordLlama extracts SWEs from the input layer of LLMs like LLaMa 3 (AI@Meta, 2024), and refines them with contrastive learning on various labelled data such as STS and text retrieval data sets. Model2Vec extracts SWEs from a Transformer text encoder called BGE-base (Xiao et al., 2023), by feeding each word w in a vocabulary into BGE without context (i.e. "[CLS], w, [SEP]") and using the output as the embedding of w. It then fine-tunes the SWEs by minimising the mean cosine distance between the sentence embeddings produced by BGE and the average of the SWEs for the same input text. To reduce the dimensionality, both WordLlama and Model2Vec apply PCA to a word embedding matrix as a post-processing step, and Model2Vec also applies the smooth inverse frequency (SIF) heuristic (Arora et al., 2017), which multiplies the norm of w by $\frac{\alpha}{\alpha+\mathrm{p}(w)}$, where $\mathrm{p}(w)$ denotes the uni-gram probability of w and $\alpha = 0.001$.

Amid the rapid growth in English models, there is a paucity of *cross-lingual* SWEs optimised for sentence representation, as most existing ones are designed for word-level tasks such as bilingual lexicon induction and cross-lingual word alignment (Mikolov et al., 2013b; Xing et al., 2015; Conneau et al., 2018; Artetxe et al., 2018; Wada et al., 2019, 2021). Although Model2Vec also releases a multilingual model that extracts SWEs from LaBSE (Feng et al., 2022), its cross-lingual performance is limited as we show in our experiments. We address this problem and propose effective SWEs not only for monolingual tasks but also for cross-lingual ones, such as (sentence) translation retrieval.

3 Methodology

Our proposed method obtains SWEs in three steps: (1) embedding extraction from Sentence Transformer (Sec. 3.1); (2) dimensionality reduction with *sentence-level* PCA (Sec. 3.2); and (3) embedding refinement with knowledge distillation or contrastive learning (Sec. 3.3). Note that only the last step involves the training (fine-tuning) of SWEs.

3.1 Embedding Extraction

We first extract SWEs from a pre-trained Sentence Transformer (**ST**). Specifically, to obtain the embedding of a word $w \in V$ (denoted as $\mathrm{E}(w)$), we sample $N = 100$ sentences $z_i^{(w)}$ $(i = 1, 2, ..., N)$ that contain w from a large corpus.[2] Then, we encode each sentence $z_i^{(w)}$ using ST, which produces $|z_i^{(w)}|$ vectors at the last Transformer layer ($|z_i^{(w)}|$ denotes the number of tokens in $z_i^{(w)}$). We then retrieve the vector at the position of w (denoted as $\mathrm{ST}(w, z_i^{(w)})$), and obtain $\mathrm{E}(w)$ as:

$$\mathrm{E}(w) = \frac{1}{N}\sum_{i=1}^{N} \mathrm{ST}(w, z_i^{(w)}). \qquad (1)$$

This way, we *decontextualise* the embeddings and obtain the SWE of w. Furthermore, we can also generate cross-lingual SWEs using multilingual ST, as it embeds different languages in the same space. For the vocabulary V, we include more words than those in the ST's original vocabulary, and if w is tokenised into subwords, we average its subword embeddings to obtain $\mathrm{ST}(w, z_i^{(w)})$ in Eqn. (1).

[1] `https://github.com/dleemiller/WordLlama` and `https://github.com/MinishLab/model2vec`, respectively.

[2] In our preliminary experiments, setting $N = 100$ did not bring a significant improvement compared to $N = 30$. Therefore, we did not experiment with larger values for N.

The idea of extracting SWEs from Transformers has been explored in previous studies (Ethayarajh, 2019; Bommasani et al., 2020; Wada et al., 2022), which show that SWEs extracted from masked LMs perform well on word-level tasks (e.g. lexical substitution). However, we find that these embeddings are not effective for sentence-level tasks (as we will show in our ablation studies in Section 5.1), and hence we further refine them in the following steps.

3.2 Sentence-level PCA

Next, we apply sentence-level PCA to the extracted SWEs in Section 3.1 to improve performance and reduce the embedding dimensionality. In previous work (including WordLlama and Model2Vec), the dimensionality reduction of SWEs is typically done by applying PCA on a word embedding matrix $Y \in \mathbb{R}^{|V| \times d}$, where d denotes the embedding dimension; we denote this as *word-level* PCA. On the other hand, we apply PCA to a *sentence* embedding matrix (as explained below), which emphasises the variance of sentences rather than words and improves performance on sentence-level tasks.

Specifically, we randomly sample $M = 100\text{k}$ sentences from those used in Section 3.1, and represent each sentence by averaging SWEs. When training cross-lingual SWEs, we sample M sentences from L languages and concatenate them; in our experiments, we train bilingual SWEs (i.e. $L = 2$) unless otherwise specified. Given the sentence embedding matrix $X \in \mathbb{R}^{LM \times d}$ (with $L = 1$ denoting monolingual SWEs) and its mean vector $\bar{X} \in \mathbb{R}^d$, we apply PCA to the centred matrix $X - \mathbf{1}_M \bar{X}^\top$, where $\mathbf{1}_M \in \mathbb{R}^M$ is a vector of ones, and then obtain the transformation matrix W. This matrix transforms the embedding space onto new axes that effectively capture the variance of the sentence representations. Importantly, since we represent each sentence by averaging SWEs, we can *pre-transform* the SWE $\mathrm{E}(w)$ into $\hat{\mathrm{E}}(w) = W^\top(\mathrm{E}(w) - \bar{X})$ for each word $w \in V$ once and save $\hat{\mathrm{E}}(w)$ in memory, as $W^\top(\frac{1}{|z|}\sum_{w' \in z} \mathrm{E}(w') - \bar{X}) = \frac{1}{|z|}\sum_{w' \in z} \hat{\mathrm{E}}(w')$.

To reduce the dimensionality using PCA, a common approach is to keep the principal components (PCs) with the $d'(< d)$ largest eigenvalues. However, we instead discard the *first* r PCs, and keep the $(r+1)$-th to $(r+d')$-th components. This is based on the previous finding that removing a few dominant PCs from SWEs can *counter-intuitively* improve the embedding quality (Mu and Viswanath, 2018), and we actually get positive results in our experiments, especially in cross-lingual tasks (e.g. +49.4% in F1 scores). Intriguingly, our analyses in Section 5.2 reveal that it helps remove information that is not very relevant to sentence semantics. Following Mu and Viswanath (2018), we set r to $\lfloor \frac{d}{100} \rfloor$, and refer to this method as **ABTT** (All-But-The-Top).

3.3 Embedding Refinement

Lastly, we fine-tune the dimensionality-reduced embeddings obtained in Section 3.2 to further optimise them for sentence representation. For monolingual SWEs, we perform teacher-student knowledge distillation, where we fine-tune our SWEs (= student) to reproduce the sentence similarities calculated by ST (= teacher). The teacher model is frozen during distillation and only SWEs are fine-tuned; therefore, it works with modest computational resources. Besides, this refinement step does not require labelled data, unlike WordLlama.

Formally, for a mini-batch of $K = 128$ sentences, we minimise the following loss $\mathcal{L}$:

$$\mathcal{L} = -\frac{1}{K}\sum_{i=1}^{K}\sum_{j \neq i}^{K} \sigma(\mathbf{T}, i, j) \log \sigma(\mathbf{S}, i, j)\,, \quad (2)$$

$$\sigma(\mathbf{X}, i, j) = \frac{e^{\frac{x_{i,j}}{\tau}}}{\sum_{k \neq i}^{K} e^{\frac{x_{i,k}}{\tau}}}\,, \quad (3)$$

where $\mathbf{T}, \mathbf{S} \in \mathbb{R}^{K \times K}$ denote the similarity matrices calculated by the teacher and student models, with the value at the i-th row and j-th column corresponding to the cosine similarity between the i-th and j-th sentences in the mini-batch. The temperature τ is set to 0.05 as commonly done in contrastive learning (Gao et al., 2021).[3] Note that the student model represents each sentence by averaging SWEs, while the teacher model encodes them with Transformers and applies its default pooling method. Since the teacher and student models calculate T and S independently, their embedding dimensions can be different and that makes it possible to fine-tune the dimensionality-reduced embeddings with PCA. In contrast, Model2Vec minimises the cosine distance of embeddings output by ST and SWEs, requiring both models to share the same embedding space. Consequently, it applies (word-level) PCA *after fine-tuning*, and then needs to apply the SIF heuristic to achieve optimal performance. In Section 5.3, we will show that our

[3]We did not extensively tune this hyper-parameter.

method yields a similar effect to SIF but in a more refined manner.

For cross-lingual (bilingual) SWEs, we find that performing knowledge distillation for each language leads to worse results on cross-lingual tasks, as it does not encourage SWEs to be mapped into the same cross-lingual space. Therefore, we instead make use of a parallel corpus as a source of knowledge, and fine-tune the cross-lingual SWEs in languages $\ell 1$ and $\ell 2$ by minimising the following loss $\mathcal{L}_{CL}$:

$$\mathcal{L}_{CL} = -\frac{1}{K}\sum_{i=1}^{K}(\log \sigma'(\mathbf{U}, i) + \log \sigma'(\mathbf{U}^{\top}, i)), \tag{4}$$

$$\sigma'(\mathbf{X}, i) = \frac{e^{\frac{x_{i,i}}{\tau}}}{\sum_{k=1}^{K} e^{\frac{x_{i,k}}{\tau}}}, \tag{5}$$

where $\mathbf{U} \in \mathbb{R}^{K\times K}$ denotes the similarity matrix calculated by SWEs, and $u_{i,j}$ denotes the cosine similarity between the i-th sentence in $\ell 1$ and j-th sentence in $\ell 2$ in the mini-batch. The i-th sentences in $\ell 1$ and $\ell 2$ are a pair of translations, and we create mini-batches by sampling K translation pairs from a parallel corpus.[4]

4 Experiments

4.1 Our Models

We train our monolingual (English) SWEs using GTE-base (Li et al., 2023), which achieves strong performance on Massive Text Embedding Benchmark (MTEB) among BERT-sized models. We sample sentences from a large-scale unlabelled corpus (CC-100 (Wenzek et al., 2020; Conneau et al., 2020)) and use them in each step of our method. Since CC-100 contains paragraphs as well as sentences, we segment text into sentences using a sentence splitter[5] and retrieve N short sentences for each word $w \in V$.[6] For the vocabulary V, we include the 150k most frequent words (case-insensitive) in CC-100. To generate cross-lingual SWEs, we use mGTE (Zhang et al., 2024), the multilingual version of GTE trained on 75 languages. We train bilingual embeddings for three language pairs, namely English–German (en–de), English–Chinese (en–zh), and English–Japanese (en–ja). We sample short sentences from large-scale parallel corpora (CC-Matrix (Schwenk et al., 2021)) and use them to extract SWEs in Section 3.1 and calculate the cross-lingual loss in Eqn. (4). For en–ja, we also use JParaCrawl v3.0 (Morishita et al., 2020, 2022) to augment data. For V in English and German, we choose the 150k most frequent words (case-sensitive) in the corresponding corpora, and 30k most frequent *tokens* (which can be a subword or consist of multiple words) in Japanese and Chinese.[7]

In Section 3.2, we reduce the dimensionality to $d' \in \{256, 512\}$,[8] and in Section 3.3 we fine-tune SWEs using Adam (Kingma and Ba, 2015) with the learning rate of 0.001 for 30k steps, and apply early stopping based on the validation loss; the train and validation data consist of sentences sampled from those used in Section 3.1. During inference, we tokenise the text z into words (or tokens in Japanese and Chinese) and strip punctuation, and average SWEs to represent z. When a word is not in V, we tokenise it into subwords using (m)GTE's tokeniser, and truncate it by removing the last subword until the remaining string is found in V.[9]

4.2 Baselines

In monolingual experiments, we include two traditional SWEs as our baselines, namely fastText (Bojanowski et al., 2017) and Sent2vec. Both models are self-supervised based on word co-occurrence information, with Sent2vec performing better on sentence-level tasks. We also compare our model against two recent GitHub-hosted models: WordLlama and Model2Vec. We use *wordllama-l3-supercat* for WordLlama, which is based on LLaMa 3, and *potion-base-32M* for Model2Vec, the latest and state-of-the-art model (released in January 2025) based on BGE-base. SWE models, including ours, produce sentence vectors by averaging SWEs and applying L2-normalisation.

In cross-lingual experiments, we compare our model against the multilingual Model2Vec model (*M2V_multilingual_output*). For English–German,

[4]We transpose U in Eqn. (4) to calculate the denominator in Eqn. (5) for each language.

[5]We use the one at `https://github.com/microsoft/BlingFire`.

[6]We find that using long text for extracting and fine-tuning our SWEs results in worse performance, likely due to the gap of information accessible to the teacher and student models during knowledge distillation (i.e. word order information).

[7]This is because mGTE's tokeniser segments sentences into tokens without performing word segmentation in both languages (which have no whitespace word boundary).

[8]In most cases, 99% of the variance can be explained with $d' = 512$ when the original dimensionality d is 768.

[9]For instance, if *tokeniser* is out of vocabulary, we split it into subwords (e.g. *token #ise #r*), and strip the last subword(s). Then, if *tokenise* (or *token*) is in V, we use its embedding.

Models (d)	STS12	STS13	STS14	STS15	STS16	STS17	STS22	STS-B	SICK-R	BIO
Sentence Transformers (STs)										
MiniLM-L6 (384)	72.4	80.6	75.6	85.4	79.0	87.6	67.2	82.0	77.6	81.6
SimCSE (768)	75.3	84.7	80.2	85.4	80.8	89.4	62.0	84.2	80.8	68.4
BGE-base (768)	78.0	84.2	82.3	88.0	85.5	86.4	65.9	86.4	80.3	**86.9**
GTE-base (768)	74.4	84.7	80.1	87.2	85.0	**90.6**	**68.6**	86.0	79.4	83.6
LLM2Vec (4,096)	**79.3**	**84.8**	**82.9**	**88.1**	**86.5**	89.6	67.7	**88.0**	**83.9**	84.9
Static Word Embeddings (SWEs)										
fastText (300)	57.2	69.2	62.8	73.0	64.2	70.2	52.1	56.5	60.1	63.0
Sent2Vec (700)	54.1	66.3	65.8	78.0	70.6	82.2	54.5	68.0	63.2	55.2
WordLlama (512)	63.9	73.3	69.1	81.5	76.1	85.2	60.2	77.0	67.0	71.3
Model2Vec (512)	62.7	77.6	72.9	80.8	76.9	**87.1**	**64.3**	76.8	65.7	77.6
OURS (256)	**68.3**	**79.3**	**75.9**	**83.1**	**79.4**	86.1	59.3	**79.2**	**68.0**	**80.3**

Table 1: Results on STS data sets. The best scores among the ST and SWE models are boldfaced.

Models (d)	100ρ	runtime (s)	
		GPU	CPU
MiniLM-L6 (384)	85.4	1.7	8.8
GTE-base (768)	87.2	3.4	52.7
LLM2Vec (4,096)	88.1	30.3	>10,000
WordLlama (512)	81.5	–	0.4
Model2Vec (512)	80.8	–	0.3
OURS (256)	83.1	–	0.4

Table 2: Scores and runtime (in second) on STS15. The runtime is averaged over three independent runs, and is measured on the same virtual machine with or without enabling a GPU (NVIDIA A100-SXM4-40GB).

we also evaluate the cross-lingual SWE available at the MUSE GitHub repository[10] (Conneau et al., 2018), which aligns English and German fastText embeddings using a bilingual dictionary. This model has been widely used in research papers.

4.3 Experiments on English Tasks

We first evaluate monolingual (English) models on semantic textual similarity (STS), a well-established NLP task for sentence semantic evaluation. It is a task of calculating textual similarity for each sentence pair, and models are evaluated based on Spearman's rank correlation ρ between their predictions and human judgements. Table 1 shows the results on STS12–17, 22, STS Benchmark (STS-B), SICK-R, and BIOSSES (BIO) data sets (Agirre et al., 2012, 2013; Bandhakavi et al., 2014; Biçici, 2015; Agirre et al., 2016; Chen et al., 2022; Marelli et al., 2014; Soğancıoğlu et al., 2017). The first five rows show performance of STs for reference (*not* as baselines of our model): SimCSE[11] is a pioneering ST, which often serves as a strong baseline in STS research; MiniLM-L6[12] is a popular light-weight ST based on MiniLM (Wang et al., 2020); BGE and GTE are the models used to train Model2Vec and OURS, respectively; and LLM2Vec is a recent large-scale model built on LLaMa-3-8B. The table shows that OURS substantially outperforms all SME baselines on most data sets, even though it has the smallest dimensionality ($d = 256$). Our model also outperforms MiniLM-L6 on STS14 and STS16 and SimCSE on BIOSSES, where the sentences are sampled from biomedical domains, indicating the robustness of OURS to various types of text. Table 2 compares the scores and runtime on STS15. While falling behind STs in accuracy, OURS (and other SWEs) can run extremely fast (20 times faster than MiniLM on CPUs) without sacrificing the accuracy very much.

Additionally, to further evaluate the effectiveness of our model on other tasks such as sentence classification, we also run experiments on Massive Text Embedding Benchmark (MTEB) (Muennighoff et al., 2023). It consists of 56 data sets covering 7 different tasks, namely classification, pair classification, clustering, reranking, retrieval, STS, and summarisation evaluation. On each task, a model encodes text and returns its fixed-length vector, which is then used on downstream tasks

[10]https://github.com/facebookresearch/MUSE

[11]We use the supervised one, which performs better than the unsupervised one.

[12]sentence-transformers/all-MiniLM-L6-v2

Models (d)	Avg-s2s	Avg
MiniLM-L6 (384)	65.95	56.26
SimCSE (768)	62.75	48.86
BGE-base (768)	71.18	63.55
GTE-base (768	71.51	64.11
LLM2Vec (4,096)	**72.94**	**65.01**
fastText (300)	53.68	43.05
Sent2Vec (700)	56.33	46.29
WordLlama (512)	60.40	50.23
Model2Vec (512)	62.37	51.33
OURS (256)	63.76	51.97
OURS (512)	**64.06**	**52.35**

Table 3: Results on MTEB experiments in English. The best scores among STs and SWEs are bold-faced. Avg of Model2Vec is slightly lower the reported one due to a small bug in their MS MARCO evaluation.

Models (d)	BUCC		Tatoeba		
	en-de	en-zh	en-de	en-zh	en-ja
mGTE (768)	98.6	98.2	97.8	95.7	93.1
MUSE (300)	46.6	–	44.3	–	–
Model2Vec (256)	60.0	0.0	73.5	26.6	15.3
OURS (256)	**96.3**	**94.6**	**95.1**	**87.9**	**79.0**

Table 4: Results (F1) on translation retrieval tasks.

(e.g. for calculating textual features or similarities). Models are evaluated using task-specific metrics (e.g. precision) on each data set. Since MTEB contains both sentence-level and document-level tasks, and our focus is on sentence semantic representation, we report two scores: **Avg**, the average across all 56 data sets; and **Avg-s2s**, the average across the 33 data sets labelled as "s2s", where the input text is a sentence rather than a document. Table 3 presents the results, showing that our model substantially outperforms the other SWEs both in Avg and Avg-s2s, even with the smallest embedding size. Moreover, our model outperforms SimCSE, which is surprising given that our model does not use any sequential function like Transformer. Interestingly, we also find that ensembling OURS with WordLlama and Model2Vec further boosts performance (64.55 and 53.28 in Avg-s2s and Avg), which we describe in detail in Appendix D. We also show the scores on each task in Table 13 in Appendix.

Models (d)	STS17	STS22	
	en-de	en-de	en-zh
mGTE (768)	84.7	62.3	72.9
MUSE (300)	11.2	36.9	–
Model2Vec (256)	54.2	39.3	26.2
OURS (256)	**70.5**	**62.0**	**72.6**

Table 5: Results on cross-lingual and monolingual (German and Chinese) STS tasks.

4.4 Experiments on Cross-Lingual Tasks

We first evaluate cross-lingual SWEs on translation retrieval, a task of retrieving the translation of the input text based on cosine similarity. Table 4 shows the results (F1 scores) on BUCC (Zweigenbaum et al., 2017) and Tatoeba (Artetxe and Schwenk, 2019; community, 2021) data sets for three language pairs: English–{German, Chinese, Japanese}. The first row shows the performance of mGTE, and the rest are cross-lingual SWEs. The table clearly shows that OURS outperforms the SWE baselines by a large margin, and performs fairly well on all language pairs (e.g. 96.3% for English-German on BUCC). This suggests the applicability of our model as a lightweight approach to extracting translation candidates, which could be further validated by computationally expensive models such as mGTE. Lastly, we also evaluate SWEs on cross-lingual STS (STS17 and STS22), a task of measuring textual similarity across two different languages, and Table 5 presents the results. Again, OURS is the best performing SWE, demonstrating its effectiveness on cross-lingual tasks.

It should be noted that mGTE and Model2vec are multilingual models (i.e. aligned across more than two languages) whereas MUSE and OURS are bilingual ones, and this can place our model in a favourable position in bilingual evaluation. Given this concern, we also try generating multilingual SWEs aligned across the four languages (English, German, Chinese, and Japanese) using our model, and find that it still outperforms the SWE baselines by a large margin; see Appendix A for details.

5 Analysis

5.1 Ablation Studies

To verify the effectiveness of our method, we evaluate our SWEs ($d = 256$) obtained in Sections

	en	en-de	en-zh	en-ja
Section 3.1	44.2	51.6	45.0	28.6
Section 3.2	49.9	93.8	88.6	74.6
w/o ABTT	48.8	49.6	39.2	33.2
Section 3.3	**52.0**	**95.7**	**91.2**	**79.0**

Table 6: Performance of our SWEs obtained at each step. The "en" column denotes Avg on MTEB, and the rest are the average scores on BUCC and Tatoeba. The third row shows the results without ABTT in Section 3.2.

	SimCSE	Nomic	GTE		
d	768	768	384	768	1,024
ST	62.8	69.5	69.3	71.5	**71.7**
OURS	61.2	62.7	60.8	**63.8**	62.9

Table 7: Performance (Avg-s2s on MTEB) of different STs and OURS (256) trained with them.

3.1, 3.2, and 3.3, and see whether the performance improves after each step. Table 6 shows the results. The column under "en" shows performance on MTEB (Avg), demonstrating that sentence-level PCA and knowledge distillation substantially improve the SWEs extracted in Section 3.1. Importantly, our model surpasses the performance of fastText (43.1) *even without fine-tuning in Section 3.3*, demonstrating the effectiveness of extracting SWEs from Sentence Transformers. Other important findings that are **not** presented in Table 6 (due to space restrictions) include: (1) applying *word-level* PCA in Section 3.2 performs worse than *sentence-level* PCA (48.1 vs. 49.9); and (2) performing knowledge distillation without PCA (i.e. ablating Section 3.2) results in poor performance (44.4)[13], indicating that parameter initialisation is a key to success in knowledge distillation. The last three columns in Table 6 show the average scores on BUCC and Tatoeba for each language pair. The results indicate that the SWEs obtained in Section 3.2 already achieve good scores, and fine-tuning them in Section 3.3 further boosts performance. Another important observation is that applying ABTT in Section 3.2, i.e. removing the *first* r PCs from SWEs, plays a vital role in generating cross-lingual SWEs; we will provide an in-depth analysis on its effect in Section 5.2.

Table 7 shows the performance of our models trained with different STs, namely SimCSE, Nomic (Nussbaum et al., 2024),[14] and GTE-{small/base/large}. It indicates that OURS performs better using better STs (SimCSE < Nomic < GTE) as its teacher model,[15] suggesting that its performance might further increase as ST models evolve. That being said, our model does not benefit from using the largest GTE model, possibly due to the large discrepancy of the parameter sizes between the teacher and student models during knowledge distillation. Performance on each MTEB task is shown in Table 15 in Appendix.

5.2 Effects of PCA

To investigate why PCA with ABTT is particularly effective on cross-lingual tasks, we analyse what information is encoded in the first two PCs, and find that it captures the variance largely based on the *language identity*. To visualise this, we randomly sample 1,000 language-specific words/tokens from English and Japanese and plot their embeddings based on the values in the first two PCs (PC1 and PC2). The result is presented in Figure 1 (the leftmost plot), which clearly shows that the embeddings are separated based on language. Given this finding, we also visualise the embeddings using t-SNE (van der Maaten and Hinton, 2008) before and after applying PCA wo/w ABTT, and the results are also shown in Figure 1. We can see that the embeddings create language clusters before PCA and after PCA *without* ABTT, but applying PCA *with* ABTT renders the embeddings more language-agnostic by removing the language-identity components.[16] We also observe the same trend in English–Chinese/German, as shown in Figures 5 and 6 in Appendix. Such language-specific dimensions can act as noise when measuring semantic similarity across different languages,[17] explaining why ABTT is crucial in cross-lingual SWEs. Furthermore, this observation makes us wonder whether

[13]Performance increases on some tasks but drops on others.

[14]Nomic is trained to encode text with a task-specific prefix prompt, and hence we prepend the prompt "classification: " to every input text in Section 3.1 and Section 3.3.

[15]Note that SimCSE uses [CLS] pooling while Nomic and GTE use mean pooling for generating sentence embeddings. While our model works reasonably well with both model types, this could be another reason why it performs better with Nomic and GTE than SimCSE, as our method extracts SWEs in Section 3.1 by averaging embeddings across N sentences.

[16]We also confirm this finding by measuring the silhouette scores, which are: 0.10 before PCA; 0.11 after PCA w/o ABTT; and 0.02 after PCA w/ ABTT, respectively.

[17]For instance, Japanese sentences with many English loanwords can have high similarities to English sentences regardless of the semantics.

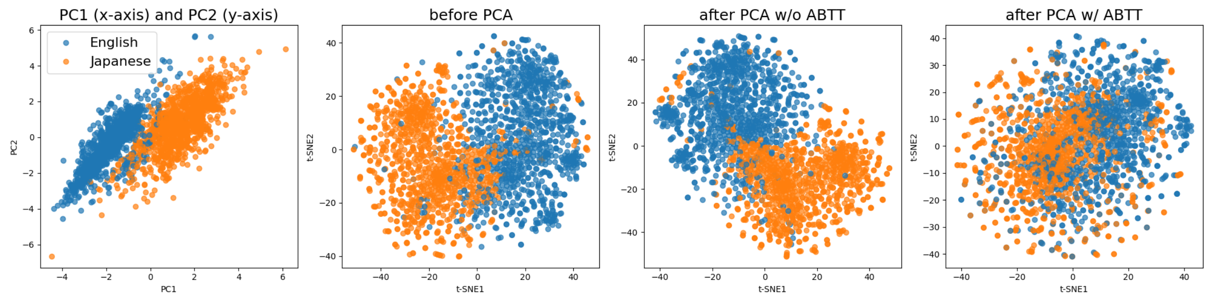

Figure 1: Scatter plots of our cross-lingual SWEs in English (blue) and Japanese (orange). The leftmost shows the 1st and 2nd principal components, and the rest are t-SNE visualisation before and after applying PCA wo/w ABTT.

i	Smallest	Largest
1	advantages, maps, permits, categories, manufacturers	haha, ha, grinned, omg, oh
2	appreciate, hey, haha, btw, anyways	shook, sheriff, snapped, glanced, sighed
6	nov, oct, dec, 2020, feb	charming, compliment, courteous, friendly, softly

Table 8: Examples of words with the smallest and largest values in the i-th principal component.

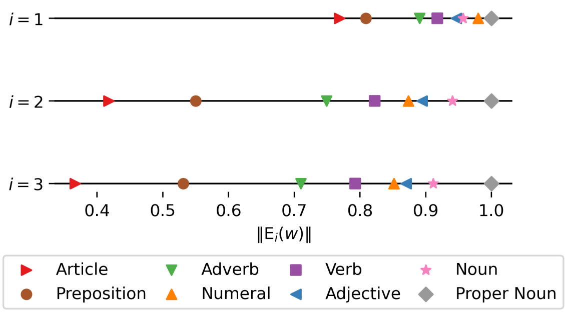

Figure 2: Comparison of word embedding norms obtained in Sections 3.1, 3.2, and 3.3. For normalisation, norms of each POS tag are divided by the maximum value of all tags (i.e. the norm of Proper Noun).

such language segregation also occurs in the mGTE embedding space; we investigate this and find that it *does* create language clusters at lower layers of Transformer, but generates more language-agnostic embeddings at higher layers. Since it deviates from the main subject of this study, we discuss more details in Appendix C.

Lastly, we also analyse the dominant PCs of monolingual (English) SWEs, and find that some of them seem to capture linguistic styles or domains. Table 8 shows examples of the words that have the five largest and smallest values in PC1, PC2, and PC6 among the 10k most frequent words in V. In PC1 and PC2, words seem to be clustered based on the word frequency in casual text, whereas in PC6, date-related words seem to have large values. To verify whether PC1 and PC2 are actually related to textual formality, we conduct a simple experiment using the data set released by Chhaya et al. (2018), where 10 human annotators judge the formality level of 960 emails. We encode each email by averaging SWEs and measure the Pearson correlation coefficient between the values in PC1, PC2, or PC6, and the mean scores of the 10 annotators. We observe that the correlations are: -0.63, 0.37, and -0.11 for PC1, PC2, and PC6, confirming a moderate or weak correlation for the first two PCs.[18] Albeit those components capture the variance of sentences effectively, they would not necessarily serve as useful features on semantic tasks, explaining why PCA w/ ABTT works well for monolingual SWEs as well.[19]

5.3 Analysis of Word Embedding Norm

To investigate how our method optimises word embeddings, we analyse the norms of SWEs obtained in Sections 3.1, 3.2, and 3.3; we denote them as $\|\mathrm{E}_i(w)\|$ $(i = 1, 2, 3)$. For the purpose of analysis, we compare the norms of the 10k most frequent words in Brown Corpus, which consists of text from various domains and provides part-of-speech (POS) tags for each word. First, we compare the mean norms of word embeddings for each POS tag; for words that are annotated with multiple POS tags (e.g. *notice*), we use the most frequent one. The results are shown in Figure 2. For normalisation, we divide the norms of each POS tag by that of

[18]We surmise that this may be attributed to the use of sentences sampled from a web corpus in our model, which is a mixed bag of formal and casual text. It would be interesting to see how the selection of text data affects the outcome.

[19]But on classification tasks, ABTT often harms performance.

d	$\|V\|$	Avg-s2s	Avg
512	150k	**64.1**	**52.4**
256	150k	63.8	52.0
128	150k	62.6	50.4
64	150k	52.5	41.4
256	100k	63.5	51.8
256	75k	63.4	51.5
256	50k	62.7	51.0
256	30k	62.2	50.2

Table 9: Performance of our model on MTEB with different embedding and vocabulary sizes.

"Proper Noun", which always has the largest value regardless of i. It shows that the differences in the norms are emphasised after PCA and knowledge distillation. In particular, the relative norms of "Article" (e.g. *the*), "Preposition" (e.g. *of*), and "Numeral" (e.g. *2025*, *first*) drop sharply, likely because these types of words do not affect sentence semantics very much. On the other hand, "Proper Noun" and "Noun" have the largest norms after PCA and knowledge distillation, as they often convey information relevant to semantics. This result is somewhat akin to the previous finding that ST tends to emphasise nominal information of a sentence (Nikolaev and Padó, 2023). In Appendix B, we also perform the same analysis for SWE baselines (fastText, WordLlama, and Model2Vec) and reveal that fastText assigns relatively large norms to articles, suggesting it is not well optimised for sentence representation.

Lastly, we find that our method induces a similar effect to SIF, a heuristic used in Model2Vec — we measure Spearman's rank correlation between the norm and word frequency rank and find that it increases after PCA and knowledge distillation ($0.39 \rightarrow 0.43 \rightarrow 0.49$), meaning both steps decrease the norms of high-frequency words. Compared to SIF, however, knowledge distillation adjusts norms in a more nuanced way, considering not only word frequency (which is implicitly encoded by ST (Kurita et al., 2023)) but also POS tags and influence of each word on sentence semantics.

5.4 Impact of Model Size

Table 9 compares OURS trained with different dimensionalities d and vocabulary sizes $|V|$. Our model performs the best with $d \geq 256$ and also achieves good scores with $d = 128$. However, the scores drop sharply with $d = 64$, where knowledge distillation rather harms performance (44.0 → 41.4) likely due to the large discrepancy of the embedding spaces between the teacher and student models. In terms of $|V|$, our model performs well with $|V| \geq 75\text{k}$. Hence, setting $d = 256$ and $|V| \geq 75\text{k}$ would offer a good balance between performance and memory usage (in English). We provide scores on each task in Table 16 in Appendix.

6 Conclusion

We propose new word embeddings optimised for sentence semantic representation. We extract embeddings from Sentence Transformers and improve them with sentence-level PCA followed by knowledge distillation or contrastive learning. Our rigorous experiments show that our model outperforms baselines on both monolingual and cross-lingual semantic tasks. Our in-depth analyses on word embeddings also reveal that our method successfully removes embedding components that are not very relevant to sentence semantics, and adjusts the vector norms based on the influence on sentence semantics.

7 Limitations

Since our model (and other SWEs) does not consider word order information (i.e. a bag-of-words model), its performance is limited on tasks where models are required to understand long context, such as document retrieval. Importantly, however, there is *always* a trade-off between model performance and computational complexity; in this work, we focus on improving the cost-efficiency, an important factor for real-world applications that is nonetheless largely neglected among the research community these days. To improve performance (at the cost of computational cost), one can simply add a small network that can capture sequential information (e.g. Transformer, CNN, LSTM (Hochreiter and Schmidhuber, 1997)) on top of our word embeddings, and train them with labelled data as performed during the training process of Sentence-Transformer models. Another idea to better capture sentence semantics is to represent text with a set of word embeddings with positional information (rather than taking their average), and calculate textual similarity by measuring the distance (or optimal transport cost) between two distributions, e.g. Word Mover's Distance (Kusner et al., 2015).

References

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Eneko Agirre, Daniel Matthew Cer, Mona T. Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *International Workshop on Semantic Evaluation*.

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: a pilot on semantic textual similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, page 385–393, USA. Association for Computational Linguistics.

AI@Meta. 2024. Llama 3 model card.

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

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 789–798, Melbourne, Australia. Association for Computational Linguistics.

Mikel Artetxe and Holger Schwenk. 2019. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond. *Transactions of the Association for Computational Linguistics*, 7:597–610.

Anil Bandhakavi, Nirmalie Wiratunga, Deepak P, and Stewart Massie. 2014. Generating a word-emotion lexicon from #emotional tweets. In *Proceedings of the Third Joint Conference on Lexical and Computational Semantics (*SEM 2014)*, pages 12–21, Dublin, Ireland. Association for Computational Linguistics and Dublin City University.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*.

Ergun Biçici. 2015. RTM-DCU: Predicting semantic similarity with referential translation machines. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 56–63, Denver, Colorado. Association for Computational Linguistics.

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

Rishi Bommasani, Kelly Davis, and Claire Cardie. 2020. Interpreting Pretrained Contextualized Representations via Reductions to Static Embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4758–4781, Online. Association for Computational Linguistics.

Xi Chen, Ali Zeynali, Chico Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw Grabowicz, Scott Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1094–1106, Seattle, United States. Association for Computational Linguistics.

Niyati Chhaya, Kushal Chawla, Tanya Goyal, Projjal Chanda, and Jaya Singh. 2018. Frustrated, polite, or formal: Quantifying feelings and tone in email. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 76–86, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Tatoeba community. 2021. Tatoeba: Collection of sentences and translations.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2018. Word translation without parallel data. In *Proceedings of the 6th International Conference on Learning Representations*, Vancouver, BC, Canada.

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

Kawin Ethayarajh. 2019. 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. Association for Computational Linguistics.

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hila Gonen, Shauli Ravfogel, Yanai Elazar, and Yoav Goldberg. 2020. It‘s not Greek to mBERT: Inducing word-level translations from multilingual BERT. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 45–56, Online. Association for Computational Linguistics.

Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. 2015. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*.

Hiroto Kurita, Goro Kobayashi, Sho Yokoi, and Kentaro Inui. 2023. Contrastive learning-based sentence encoders implicitly weight informative words. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10932–10947, Singapore. Association for Computational Linguistics.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From word embeddings to document distances. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France. PMLR.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2025. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *Preprint*, arXiv:2308.03281.

Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. 2020. On the language neutrality of pre-trained multilingual representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1663–1674, Online. Association for Computational Linguistics.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A SICK cure for the evaluation of compositional distributional semantic models. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. In *Proceedings of the 1st International Conference on Learning Representations (Workshop)*, Scottsdale, Arizona, USA.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. 2013b. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

D. Lee Miller. 2024. Wordllama: Recycled token embeddings from large language models.

Makoto Morishita, Katsuki Chousa, Jun Suzuki, and Masaaki Nagata. 2022. JParaCrawl v3.0: A large-scale English-Japanese parallel corpus. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 6704–6710, Marseille, France. European Language Resources Association.

Makoto Morishita, Jun Suzuki, and Masaaki Nagata. 2020. JParaCrawl: A large scale web-based English-Japanese parallel corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 3603–3609, Marseille, France. European Language Resources Association.

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

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Dmitry Nikolaev and Sebastian Padó. 2023. Representation biases in sentence transformers. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3701–3716, Dubrovnik, Croatia. Association for Computational Linguistics.

Zach Nussbaum, John X. Morris, Brandon Duderstadt, and Andriy Mulyar. 2024. Nomic Embed: Training a reproducible long context text embedder. *Preprint*, arXiv:2402.01613.

Matteo Pagliardini, Prakhar Gupta, and Martin Jaggi. 2018. Unsupervised learning of sentence embeddings using compositional n-gram features. 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 528–540, New Orleans, Louisiana. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Holger Schwenk, Guillaume Wenzek, Sergey Edunov, Edouard Grave, Armand Joulin, and Angela Fan. 2021. CCMatrix: Mining billions of high-quality parallel sentences on the web. 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 6490–6500, Online. Association for Computational Linguistics.

Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. 2017. Biosses: a semantic sentence similarity estimation system for the biomedical domain. *Bioinformatics*, 33(14):i49–i58.

Stephan Tulkens and Thomas van Dongen. 2024. Model2vec: The fastest state-of-the-art static embeddings in the world.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Takashi Wada, Timothy Baldwin, Yuji Matsumoto, and Jey Han Lau. 2022. Unsupervised lexical substitution with decontextualised embeddings. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4172–4185, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Takashi Wada, Tomoharu Iwata, and Yuji Matsumoto. 2019. Unsupervised multilingual word embedding with limited resources using neural language models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3113–3124, Florence, Italy. Association for Computational Linguistics.

Takashi Wada, Tomoharu Iwata, Yuji Matsumoto, Timothy Baldwin, and Jey Han Lau. 2021. Learning contextualised cross-lingual word embeddings and alignments for extremely low-resource languages using parallel corpora. In *Proceedings of the 1st Workshop on Multilingual Representation Learning*, pages 16–31, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, volume 33, pages 5776–5788. Curran Associates, Inc.

Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. 2020. CCNet: Extracting high quality monolingual datasets from web crawl data. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4003–4012, Marseille, France. European Language Resources Association.

Shitao Xiao, Zheng Liu, Peitian Zhang, and Niklas Muennighoff. 2023. C-pack: Packaged resources to advance general chinese embedding. *Preprint*, arXiv:2309.07597.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, Denver, Colorado. Association for Computational Linguistics.

Jiahao Xu, Wei Shao, Lihui Chen, and Lemao Liu. 2023. DistillCSE: Distilled contrastive learning for sentence embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8153–8165, Singapore. Association for Computational Linguistics.

Xin Zhang, Yanzhao Zhang, Dingkun Long, Wen Xie, Ziqi Dai, Jialong Tang, Huan Lin, Baosong Yang, Pengjun Xie, Fei Huang, Meishan Zhang, Wenjie Li, and Min Zhang. 2024. mGTE: Generalized long-context text representation and reranking models for multilingual text retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1393–1412, Miami, Florida, US. Association for Computational Linguistics.

Pierre Zweigenbaum, Serge Sharoff, and Reinhard Rapp. 2017. Overview of the second BUCC shared task: Spotting parallel sentences in comparable corpora. In *Proceedings of the 10th Workshop on Building and Using Comparable Corpora*, pages 60–67, Vancouver, Canada. Association for Computational Linguistics.

Models	en-de	en-zh	en-ja
Bilingual	95.7	91.2	79.0
Multilingual	93.1	86.6	71.6

Table 10: Performance of our bilingual and multilingual SWEs on translation retrieval tasks. The values denote the average scores on BUCC and Tatoeba for each language pair. Note that the multilingual model is obtained *without fine-tuning* in Section 3.3

.

Models	STS17	STS22	
	en-de	en-de	en-zh
Bilingual	70.5	62.0	72.6
Multilingual	69.7	60.6	70.4

Table 11: Performance of our bilingual and multilingual SWEs on cross-lingual STS. Note that the multilingual model is obtained *without fine-tuning* in Section 3.3.

A Results on Multilingual Embeddings

In addition to bilingual SWEs, we also try generating multilingual SWEs using our model, where English, German, Chinese, and Japanese are all embedded in the same space (as in Model2Vec). To achieve this, we apply the sentence-level PCA to the concatenation of the sentence vectors sampled from each language (i.e. setting L to 4 in Section 3.2). For multilingual SWEs, *we skip the fine-tuning step in Section 3.3* because it is specifically designed for bilingual training; however, note that it could be easily extended to multilingual training by jointly optimising the bilingual loss $\mathcal{L}_{CL}$ for multiple language pairs. Table 10 and Table 11 compare the performance of our bilingual and multilingual SWEs. Although the multilingual model performs worse than the bilingual one (which benefits from fine-tuning), it still substantially exceeds the performance of the SWE baselines (MUSE and Model2Vec) shown in Table 4 and Table 5.

POS	OURS	FT	WL	M2V
Article	0.37	0.55	0.22	0.24
Preposition	0.53	0.66	0.33	0.57
Adverb	0.71	0.72	0.52	0.85
Numeral	0.85	0.76	0.66	0.79
Verb	0.79	0.81	0.67	0.86
Adjective	0.87	0.81	0.74	0.90
Noun	0.91	0.86	0.84	0.91
Proper Noun	1.00	1.00	1.00	1.00

Table 12: Comparison of word embedding norms generated by fastText (FT), WordLlama (WL), Model2Vec (M2V), and our model (OURS). For normalisation, norms of each POS tag are divided by the maximum value of all tags, which is the norm of Proper Noun for all models.

B Analysis of Baseline Embedding Norms

Table 12 compares word embedding norms generated by fastText (FT), WordLlama (WL), Model2Vec (M2V), and our model (OURS). Each value is normalised (divided) by the norm of Proper Noun within each model, as done in Figure 2. Interestingly, it shows that Proper Noun has the largest norm for all models. Looking at each model, fastText has relatively large norms for Article and Preposition (which usually have a small influence on sentence semantics), suggesting it is not well optimised for sentence representation (as we demonstrated in our experiments). For WordLlama, only Noun and Proper Noun have larger values than 0.8, suggesting it emphasises nominal information more than the other models. This partially explains why it performs well on document retrieval tasks on MTEB (as shown in Table 13), where capturing topical information is arguably more important than the semantics. Model2Vec and OURS display similar patterns, with Model2Vec assigning a higher weight to Verb and Adverb and OURS emphasising Numeral more.

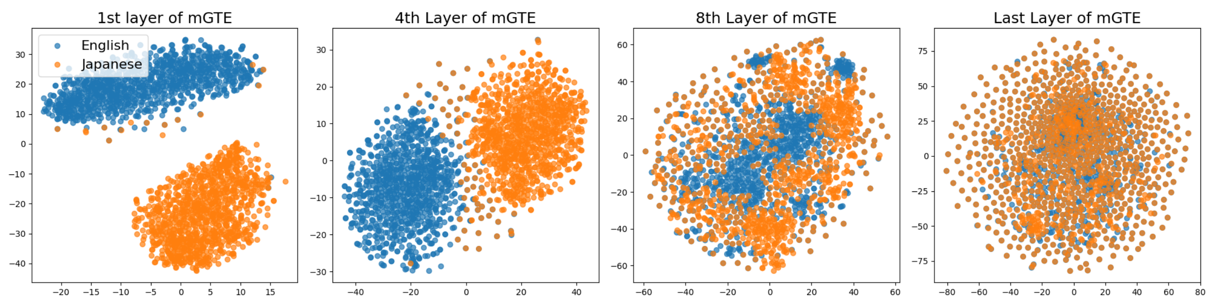

Figure 3: The t-SNE visualisation of sentence embeddings produced by **mGTE-base** at the 1st, 4th, 8th, and 12th (= last) layers of Transformer for English (blue) and Japanese (orange) sentences. The encoded sentences are 1,000 pairs of translations sampled from a parallel corpus.

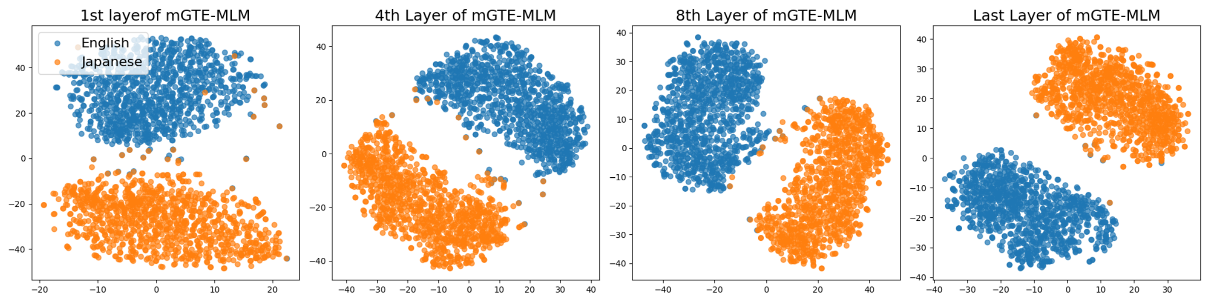

Figure 4: The t-SNE visualisation of sentence embeddings produced by **mGTE-MLM-base** (the backbone masked language model of mGTE before fine-tuning) at the 1st, 4th, 8th, and 12th (= last) layers of Transformer for English (blue) and Japanese (orange) sentences. The encoded sentences are 1,000 pairs of translations sampled from a parallel corpus.

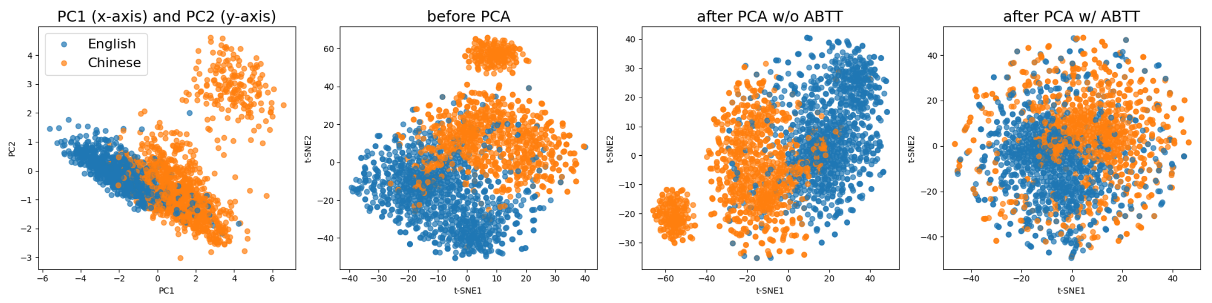

Figure 5: Scatter plots of our cross-lingual SWEs in **English (blue) and Chinese (orange)**. The leftmost shows the 1st and 2nd principal components, and the rest are t-SNE visualisation before and after applying PCA with/without ABTT.

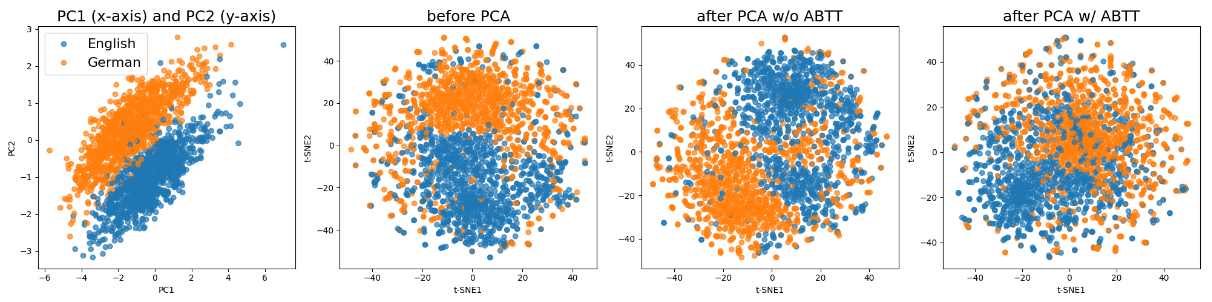

Figure 6: Scatter plots of our cross-lingual SWEs in **English (blue) and German (orange)**. The leftmost shows the 1st and 2nd principal components, and the rest are t-SNE visualisation before and after applying PCA with/without ABTT.

C Analysis of mGTE Embeddings

Figure 3 shows the t-SNE visualisation of sentence embeddings produced by mGTE-base at the 1st, 4th, 8th, and 12th (= last) layers of Transformer. We encode 1,000 translation pairs sampled from an English–Japanese parallel corpus; as such, ideally they should have close representations regardless of language. To obtain sentence vectors, we apply mean pooling at each layer.[20] The figure shows that the embeddings are completely separated by language at the 1st and 4th layers (just like our SWEs before PCA in Figure 1), but are less clustered at the 8th layer and completely mixed at the the last layer.[21] This suggests that mGTE recognises the input language at lower layers, and then captures language-agnostic features such as sentence semantics at higher layers. We also verify this hypothesis by evaluating embeddings at each layer on a translation retrieval task and observing better performance at higher layers.

Notably, this result is somewhat inconsistent with the previous findings that multilingual masked language models (MLMs) like mBERT have language-specific subspaces in their embedding, even at near the last layer (Gonen et al., 2020; Libovický et al., 2020). One possible reason for this is that mGTE is fine-tuned on various tasks to produce sentence representation, which might make it generate language-agnostic embeddings. To verify this hypothesis, we also evaluate the embeddings of mGTE's backbone MLM (i.e. a BERT-like model with several enhancements) before it was fine-tuned on labelled data,[22] and Figure 4 shows the results. It clearly indicates that the embeddings are completely separated based on language at all layers, suggesting that mGTE benefits from fine-tuning in producing language-agnostic representations. Before fine-tuning, multilingual MLMs are pre-trained to predict masked tokens in each language at the last layer, and that would naturally encourage them to have language-specific subspaces, as discussed in Gonen et al. (2020).

[20]For the last layer, we also tried using the [CLS] embedding following the default pooling configuration, and observed a similar result.

[21]As such, applying PCA did not improve mGTE's performance.

[22]We use the model at `https://huggingface.co/Alibaba-NLP/gte-multilingual-mlm-base`.

D Word Embedding Ensembling

Given the success of ensemble learning on many NLP tasks, we try combining J SWEs trained with different algorithms. Specifically, for the input text z, we generate its embedding $\mathrm{f}(z)$ as:

$$\mathrm{f}(z) = \frac{1}{\sqrt{\sum_{i=1}^{J} \lambda_i}} \left[\frac{\sqrt{\lambda_i}}{\|\mathrm{f}_i(z)\|} \mathrm{f}_i(z) \right]_{i=1}^{J}, \quad (6)$$

$$\mathrm{f}_i(z) = \frac{1}{|z|} \sum_{w' \in z} \mathrm{E}_i(w'), \quad (7)$$

where $\mathrm{f}_i(z) \in \mathbb{R}^{d_i}$ denotes the embedding of z represented by the average of the i-th SWE, and λ_i is a scalar hyper-parameter that controls the contribution of $\mathrm{f}_i(z)$ to $\mathrm{f}(z)$; $\|\cdot\|$ denotes the vector norm; and $[\mathrm{x}_i]_{i=1}^{J}$ denotes the concatenation of the vectors $\mathrm{x}_1, \mathrm{x}_2, \cdots, \mathrm{x}_J$. The dot product of $\mathrm{f}(z)$ exactly corresponds to the weighted average of the cosine similarity measured by each embedding model $\mathrm{f}_i(z)$, with the weight given by λ_i. For simplicity, we set $\lambda_1 = \cdots = \lambda_J = 1$ unless otherwise stated. Notably, unlike ensembling ST models, this method does **not** increase computational cost very much because we can pre-concatenate the embeddings of $w \in V$ as $\mathrm{E}(w) = [\mathrm{E}_i(w)]_{i=1}^{J}$, and calculate $\mathrm{f}(z)$ simply by averaging $\mathrm{E}(w)$ and normalising the embedding within each subspace of $\mathrm{E}_i(w)$.[23]

The last three rows in Table 13 show the results when we ensemble OURS with WordLlama (WL) and Model2Vec (M2V); regarding the last row **2*OURS + WL + MV**, we set λ_i to 2 for OURS and 1 for the other models to increase the contribution of OURS. It shows that it performs the best in both Avg-s2s and Avg, indicating the effectiveness of ensembling different SWEs. Looking at the scores on each task, our ensembling method successfully combines the strengths of each model, e.g. ensembling OURS with WordLlama is very effective on Reranking and Retrieval, where WordLlama performs the best of all SWEs likely due to its nature of emphasising nominal information (as discussed in Appendix B).

E Model Details

Table 14 lists the details of the models used in our paper. The Sent2vec model is downloaded from `https://github.com/epfml/sent2vec`; fastText from `https://fasttext.`

[23]On the other hand, the cost for calculating similarity and memory usage increase.

Models (d)	# Data Sets	Class.	Clust.	PairClass.	Rerank.	Retr.	STS	Summ.	Avg-s2s	Avg
		12	11	3	4	15	10	1	33	56
Sentence Transformers (STs)										
MiniLM-L6 (384)		63.06	42.35	82.37	58.04	41.95	78.90	30.81	65.95	56.26
SimCSE-supervised (768)		67.32	33.43	73.68	47.54	21.82	79.12	**31.17**	62.75	48.86
BGE-base (768)		75.53	45.77	86.55	58.86	53.25	82.40	31.07	71.18	63.55
GTE-base (768)		**77.17**	**46.82**	85.33	57.66	54.09	81.97	**31.17**	71.51	64.11
LLM2Vec-LLaMa3 (4,096)		75.92	46.45	**87.80**	**59.68**	**56.63**	**83.58**	30.94	**72.94**	**65.01**
Static Word Embeddings (SWEs)										
fastText (300)		59.09	30.75	65.77	43.63	22.21	62.82	30.33	53.68	43.05
Sent2vec (700)		61.15	31.75	71.88	47.31	27.79	65.80	29.51	56.33	46.29
WordLlama (512) [WL]		60.85	36.16	73.72	**52.82**	**33.14**	72.47	**30.83**	60.40	50.23
Model2Vec* (512) [M2V]		66.23	35.29	77.89	50.92	32.13	74.22	29.78	62.37	51.33
OURS (256)		67.29	36.67	78.99	50.91	31.32	**75.87**	30.55	63.76	51.97
OURS (512)		**67.96**	**37.14**	**79.00**	51.07	32.01	75.59	30.35	**64.06**	**52.35**
Ensemble of SWEs										
OURS + WL (1,024)		66.71	37.56	79.18	**52.82**	34.13	75.63	30.04	64.10	52.87
OURS + WL + MV (1,536)		67.70	37.06	79.37	52.60	**34.52**	75.63	30.20	64.17	53.09
2*OURS + WL + M2V (1,536)		**68.17**	**37.60**	**79.54**	52.39	34.32	**75.89**	**30.22**	**64.55**	**53.28**

Table 13: Results on MTEB data sets. The last three rows denote the scores when we ensemble OURS with WordLlama [WL] and Model2Vec [M2V] using the method described in Appendix D. The best scores within each subtable are bold-faced. *The scores for Model2Vec in Retr. and Avg are slightly lower than the reported scores, due to a small bug in their MS MARCO evaluation.

Models	Emb Size d	Model Path
MiniLM-L6	384	sentence-transformers/all-MiniLM-L6-v2
GTE-small	384	thenlper/gte-small
GTE-base	768	Alibaba-NLP/gte-base-en-v1.5
GTE-large	1,024	Alibaba-NLP/gte-large-en-v1.5
mGTE	768	Alibaba-NLP/gte-multilingual-base
BGE	768	BAAI/bge-base-en-v1.5
SimCSE	768	princeton-nlp/sup-simcse-bert-base-uncased
Nomic	768	nomic-ai/nomic-embed-text-v1.5
LLaMa3	4,096	McGill-NLP/LLM2Vec-Meta-Llama-3-8B-Instruct-mntp-supervised
fastText	300	crawl-300d-2M.vec
Sent2vec	700	sent2vec_wiki_bigrams
WordLlama	512	wordllama-l3-supercat
Model2Vec (English)	512	potion-base-32M
Model2Vec (cross-lingual)	256	M2V_multilingual_output

Table 14: Details of the models used in our experiments.

`cc/docs/en/english-vectors.html` (Mikolov et al., 2018); and the rest from Hugging Face (`https://huggingface.co/models`).

Models (d) # Data Sets	Class. 12	Clust. 11	PairClass. 3	Rerank. 4	Retr. 15	STS 10	Summ. 1	Avg-s2s 33	Avg 56
Sentence Transformers (STs)									
SimCSE-supervised (768)	67.32	33.43	73.68	47.54	21.82	79.12	**31.17**	62.75	48.86
Nomic (768)	73.55	43.93	84.61	55.78	53.01	81.94	30.40	69.52	62.28
GTE-small (384)	72.31	44.89	83.54	57.70	49.46	**82.07**	30.42	69.32	61.36
GTE-base (768)	77.17	46.82	**85.33**	57.66	54.09	81.97	**31.17**	71.51	64.11
GTE-large (1,024)	**77.75**	**47.96**	84.53	**58.50**	**57.91**	81.43	30.91	**71.65**	**65.39**
Static Word Embeddings (SWEs)									
OURS-SimCSE (256)	64.72	32.78	75.65	48.36	26.54	74.83	30.61	61.16	48.83
OURS-Nomic (256)	65.94	34.74	**79.29**	50.15	30.47	75.49	**31.49**	62.69	50.99
OURS-GTE-small (256)	63.90	34.14	78.17	49.66	27.39	72.63	31.14	60.76	49.00
OURS-GTE-base (256)	**67.29**	**36.67**	78.99	**50.91**	31.32	**75.87**	30.55	**63.76**	**51.97**
OURS-GTE-large (256)	65.36	36.11	78.13	50.42	**31.58**	75.59	30.73	62.91	51.39

Table 15: Performance of different ST models and our models ($d = 256$) distilled from each ST.

Models (d) # Data Sets	Class. 12	Clust. 11	PairClass. 3	Rerank. 4	Retr. 15	STS 10	Summ. 1	Avg-s2s 33	Avg 56
OURS (d = 64, $\|V\|$ = 150k)	60.20	29.09	60.12	44.07	17.83	62.12	30.07	52.51	41.39
OURS (d = 128, $\|V\|$ = 150k)	65.92	35.51	78.05	50.09	28.33	75.06	**30.60**	62.62	50.40
OURS (d = 256, $\|V\|$ = 150k)	67.29	36.67	78.99	50.91	31.32	**75.87**	30.55	63.76	51.97
OURS (d = 512, $\|V\|$ = 150k)	**67.96**	**37.14**	**79.00**	**51.07**	**32.01**	75.59	30.35	**64.06**	**52.35**
OURS (d = 256, $\|V\|$ = 150k)	67.29	**36.67**	**78.99**	**50.91**	**31.32**	75.87	30.55	**63.76**	**51.97**
OURS (d = 256, $\|V\|$ = 100k)	67.29	36.12	78.56	50.71	31.18	**75.90**	**30.84**	63.55	51.80
OURS (d = 256, $\|V\|$ = 75k)	**67.38**	35.67	78.55	50.59	30.56	75.67	30.36	63.36	51.51
OURS (d = 256, $\|V\|$ = 50k)	67.26	34.11	78.19	50.30	30.25	75.21	30.68	62.70	50.97
OURS (d = 256, $\|V\|$ = 30k)	66.90	32.89	78.03	50.00	28.61	75.14	29.81	62.25	50.16

Table 16: Performance of OURS trained with different embedding and vocabulary sizes.