doi: 10.20532/cit.2023.1005673

# Efficient Sentence Representation Learning via Knowledge Distillation with Maximum Coding Rate Reduction

Domagoj Ševerdija[1], Tomislav Prusina[2], Luka Borozan[1] and Domagoj Matijević[1]

[1]School of Applied Mathematics and Computer Science, University of Osijek, Croatia
[2]Universität Hamburg, Department of Informatics, Germany

Addressing the demand for effective sentence representation in natural language inference problems, this paper explores the utility of pre-trained large language models in computing such representations. Although these models generate high-dimensional sentence embeddings, a noticeable performance disparity arises when they are compared to smaller models. The hardware limitations concerning space and time necessitate the use of smaller, distilled versions of large language models. In this study, we investigate the knowledge distillation of Sentence-BERT, a sentence representation model, by introducing an additional projection layer trained on the novel Maximum Coding Rate Reduction (MCR$^2$) objective designed for general-purpose manifold clustering. Our experiments demonstrate that the distilled language model, with reduced complexity and sentence embedding size, can achieve comparable results on semantic retrieval benchmarks, providing a promising solution for practical applications.

*ACM CCS (2012) Classification:* Applied computing → Document management and text processing → Document searching

Information systems → Information retrieval → Retrieval models and ranking → Language models

Computing methodologies → Machine learning → Machine learning approaches → Neural networks

*Keywords*: Sentence embeddings, knowledge distillation, Maximum Coding Rate Reduction, semantic retrieval

## 1. Introduction

Word embeddings, known as dense vector representations of words, serve as a fundamental component in various NLP applications. These embeddings can be constructed using context-free methods [1], [2], [3] or contextualized techniques [4], [5]. While word embeddings are crucial, certain NLP applications also benefit from incorporating sentence or document representations alongside word embeddings like text classification, document retrieval, and similarity checking. Often, a weighted average (commonly referred to as pooling) of word embeddings from a sentence or document is employed. Despite its disregard for word order, this pooling approach has demonstrated reasonable performance in the works of Aldarmaki *et al.* [6]. Pre-trained language models like BERT have achieved notable success in various NLP tasks through fine-tuning. However, using contextualized word vectors from these models as sentence representations proves to be significantly inferior in terms of semantic textual similarity compared to approaches that utilize non-contextualized word vectors. These non-contextualized vectors are trained using simpler models, are context-independent, and are less sensitive to minor variations in phrasing, as outlined in Reimers *et al.* [7]. To address this limitation, researchers have developed more sophisticated methods to create efficient and high-performing universal sentence encoders. Reimers *et al.* [7] proposed the Sentence-BERT model, which involves fine-tuning the pre-trained BERT architecture on sentence pair scoring tasks using a Siamese architecture to learn enhanced sentence representations. This approach exhibited considerable improvement in downstream NLP tasks. However, it resulted in a relatively large model size, with hundreds of millions to billions of parameters and a sentence embedding dimension of 768 numbers. These properties pose challenges for efficient search and retrieval operations over databases.

In this paper, we focus on addressing this issue by reducing the dimensionality of sentence embeddings by 50%–70% while achieving comparable results across a range of NLP benchmarks. Motivated by recent advancements in LLM deployment on smaller-scale computers (*i.e.*, SBCs and edge devices), we believe our approach can be implemented for such devices without compromising performance. See [8] for a review of machine learning on microcontroller-class hardware and [9] for production-grade solutions.

This paper serves as an extension and follow-up to our conference paper [10]. The current study presents new experimental results and advancements beyond what was previously presented, making it a valuable continuation of our conference contribution.

For the sake of completeness, we are ensuring a thorough and comprehensive presentation by reiterating the review of related work, methodology, and results from the conference version of the paper. This inclusion enhances the clarity and completeness of the research, allowing readers to grasp the context and foundational aspects that lead to the novel contributions of this study.

### 1.1. Related Work

In line with the distributional hypothesis, Mikolov *et al.* [2] demonstrated the significance of computing word embeddings with lower-dimensional dense vectors, revealing intriguing mathematical properties of words, including semantic relationships, vector arithmetic, and linear structures in vector space. For example, one could compute 'king'-'man'+'woman' = 'queen' using arithmetic over word vectors. Building upon this idea, Kiros *et al.* [11] and Logeswaran *et al.* [12] pursued the development of models that predict surrounding sentences. Sent2Vec [13] emerged as a method to generate context-free sentence embeddings by averaging word vectors and n-gram vectors, analogous to FastText [14] for words. On the other hand, Conneau *et al.* [15] introduced contextualized sentence embeddings using a BiLSTM Siamese network fine-tuned on pairs of semantically similar sentences. This approach was further extended to fine-tune pre-trained language models, such as BERT, as outlined in [7]. Recently, Gao *et al.* [16] made notable improvements to this approach by proposing a contrastive learning method, achieving state-of-the-art results. The idea of projecting sentence embeddings to lower dimensions found inspiration from projecting word vectors. Surprisingly, in most cases, PCA methods yielded favorable outcomes and even retrofitted word vectors to enhance vector isotropy, leading to improved performance on NLP benchmarks. Li *et al.* [17] also demonstrated the presence of this phenomenon in sentence vectors and presented a normalizing flow method to retrofit such vectors. In a similar vein, Zhao *et al.* [18] utilized PCA for knowledge distillation from sentence embeddings. More recently, the work of Yu *et al.* [19] introduced Maximum Coding Rate Reduction (MCR$^2$), a novel learning objective facilitating the acquisition of a subspace representation based on clustering. Additionally, they showcased how this approach can be extended to address the problem of unsupervised clustering.

### 1.2. Our Contribution

In this paper, we present a novel approach to sentence embedding compression by utilizing a pre-trained sentence embedding model, specifically Sentence-BERT (SBERT), as the sentence encoder. Our method involves training a non-linear mapper on top of the encoder using the Maximal Coding Rate Reduction (MCR$^2$) as the training objective. This allows us to learn discriminative low-dimensional features that preserve all essential information from the high-dimensional data. Compared to standard training objectives like cross-entropy, our approach offers enhanced robustness and results in well-defined clusters in the embedding space. The primary contribution of our work lies in the development of a sentence embedding compression technique that achieves comparable results to the baseline sentence encoder, even with smaller sentence embedding sizes, on semantic NLP benchmarks.

Building upon the insights and findings presented in our earlier work, where we experimented only with a projection of word vectors, in this research, we delve deeper into the exploration of effective sentence representation in natural language inference, *i.e.* we expand on the concept of knowledge distillation and introduce a novel approach, utilizing the Maximum Coding Rate Reduction (MCR$^2$) objective, to distill Sentence-BERT into a smaller, more practical model, while maintaining comparable performance on semantic retrieval benchmarks.

The paper's organization is as follows: In Section 2, we delve into the details of the Maximum Rate Coding Reduction training objective, which is instrumental in computing the subspace embedding space. Additionally, we provide a description of the SBERT architecture as the sentence encoder and introduce the projection layer. Moving on to Section 3, we present the results of our experimental evaluation and engage in a detailed discussion of the outcomes. The code for our approach is publicly available and can be found on GitHub repository[1].

## 2. Methodology

For a given set of sentences $S$ of size $n$ and for each sentence $x = w_1 w_2 ... w_{|x|} \in S$ where $w_i$ are tokens (words) from a predefined vocabulary, our task is to construct a lower dimensional embedding $z \in \mathbb{R}^d$ that captures essential semantic information specific to that sentence. In this context, the embedding ($z$) serves as a compact representation of the sentence's meaning, facilitating downstream tasks such as document classification, sentiment analysis, or information retrieval. Our idea is to extend SBERT and, from its embedding, compute a small projector to reduce the dimension, *i.e.*, given the set of SBERT's embeddings $Z \in \mathbb{R}^{(d \times n)}$ of the dataset $S$, find a $\hat{Z} \in \mathbb{R}^{(\hat{d} \times n)}$ that preserves semantic information extracted by SBERT.

### 2.1. Learning a Subspace Representation with MCR$^2$

Following the idea from Li *et al.* [20] we aim to minimize the angle between similar sentences and maximize the entropy of the whole dataset. More formally, for two representations of two sentences $\hat{z}_1, \hat{z}_2 \in \mathbb{R}^{\hat{d}}$ we measure how similar they are by cosine similarity function:

$$D(\hat{z}_1, \hat{z}_2) = \frac{\cos \hat{z}_1^{\mathsf{T}} \hat{z}_2}{\|\hat{z}_1\|_2 \|\hat{z}_2\|_2} \tag{1}$$

For two sets of representations $\hat{Z}_1, \hat{Z}_2 \in \mathbb{R}^{(\hat{d} \times b)}$ we define this function as:

$$D(\hat{Z}_1, \hat{Z}_2) = \frac{1}{b} \sum_{i=1}^{b} D(\hat{z}_{1,i}, \hat{z}_{2,i}), \tag{2}$$

where $\hat{z}_{1,\, i}$ is the $i$-th element of $\hat{Z}_1$ and $\hat{z}_{2,\, i}$ is the $i$-th element of $\hat{Z}_2$. Given pairs of similar sentences we want them to have the $D$ score as large as possible.

For a set of representations $\hat{Z} \in \mathbb{R}^{(\hat{d} \times n)}$ with $n$ elements, its entropy is defined as:

$$R_\varepsilon(\hat{Z}) = \frac{1}{2} \log\det\left( I + \frac{d}{n\varepsilon^2} \hat{Z}\hat{Z}^{\mathsf{T}} \right), \tag{3}$$

for a given parameter $\varepsilon$ and identity matrix $I$. The function we employ closely resembles the Shannon coding rate function for a multivariate Gaussian distribution, with an average distortion $\epsilon$ (for further details, refer to Cover *et al.* [21]). Our aim in maximizing this function, as denoted by (3), is to increase the volume of the embedding-packed ball. The theoretical background supporting this notion is beyond the scope of this paper, but it can be found in the work of Ma *et al.* [22]. In their paper, they explore rate distortion, $\varepsilon$-ball packing, and lossy encoding utilizing normally distributed data. In our approach, we optimize this function in

[1] https://github.com/tomo61098/compsentrepMCR2

parallel with (2) to create significant separation between each sentence, except for the similar pairs that we aim to keep in close proximity. Additionally, given cluster assignments, we can measure the entropy of each cluster with:

$$R_\varepsilon\left(\hat{Z}, \Pi_k\right) = \frac{n_k}{2n} \log\det\left(I + \frac{d}{n_k \varepsilon^2} \hat{Z} \Pi_k \hat{Z}^\mathsf{T}\right), \quad (4)$$

where $\Pi_k$ is a diagonal matrix with $i$-th entry being 1 if the $i$-th sentence belongs to cluster $k$, otherwise 0, and $n_k = tr(\Pi_k)$, trace of matrix $\Pi_k$, *i.e.*, number of points in this cluster. Combining functions (1), (2) and (3) into one we get the MRC$^2$ loss function defined as follows:

$$\begin{aligned} L\left(\hat{Z}, \Pi\right) = &-R_\varepsilon\left(\hat{Z}\right) \\ &+ \sum_{i=1}^{k} R_\varepsilon\left(\hat{Z}, \Pi_i\right) \quad (5) \\ &- \lambda D\left(\hat{Z}_1, \hat{Z}_2\right), \end{aligned}$$

for some hyperparameter $\lambda$ and pairs of similar sentences respectively divided into two sets $\hat{Z}_1$, $\hat{Z}_2$. Matrix $\Pi$ is the clustering of data given by the user or learned by the architecture. The selection of $\lambda$ values is crucial in determining the proximity of similar sentences in our projection. When using larger $\lambda$ values, the network tends to merge similar pairs into the same vector, which, if not carefully managed, can result in collapsing all vectors into a single representation. Conversely, with smaller $\lambda$ values, the network has greater flexibility in deciding which vector embeddings to keep close. However, this can lead to an undesirable vector representation that maximally distances vectors from each other. Striking the right balance in $\lambda$ is essential to achieve the desired outcome in our model. Therefore, by minimizing (4) we:

- maximize the volume of all embeddings $R_\varepsilon(\hat{Z})$,
- minimize the sum of volumes of clusters $\sum_{i=1}^{k} R_\varepsilon\left(\hat{Z}, \Pi_i\right)$,
- maximize the cosine similarity of pairs of similar sentences $\lambda D(\hat{Z}_1, \hat{Z}_2)$.

The consequence of this is that after the minimization we have an embedding in which different clusters are orthogonal to each other (see Yu *et al.* [19] for more details), *i.e.*

$$i \neq j \Rightarrow \hat{Z}_i \hat{Z}_j^\mathsf{T} = 0. \quad (6)$$

## 2.2 Architecture

Our model takes as input a batch of sentences $S$, produces an encoding of a sentence representation $Z$ and outputs projected sentence representations $\hat{Z}$ together with cluster assignments $\Pi$ for $S$. The overall architecture is shown in Figure 1.

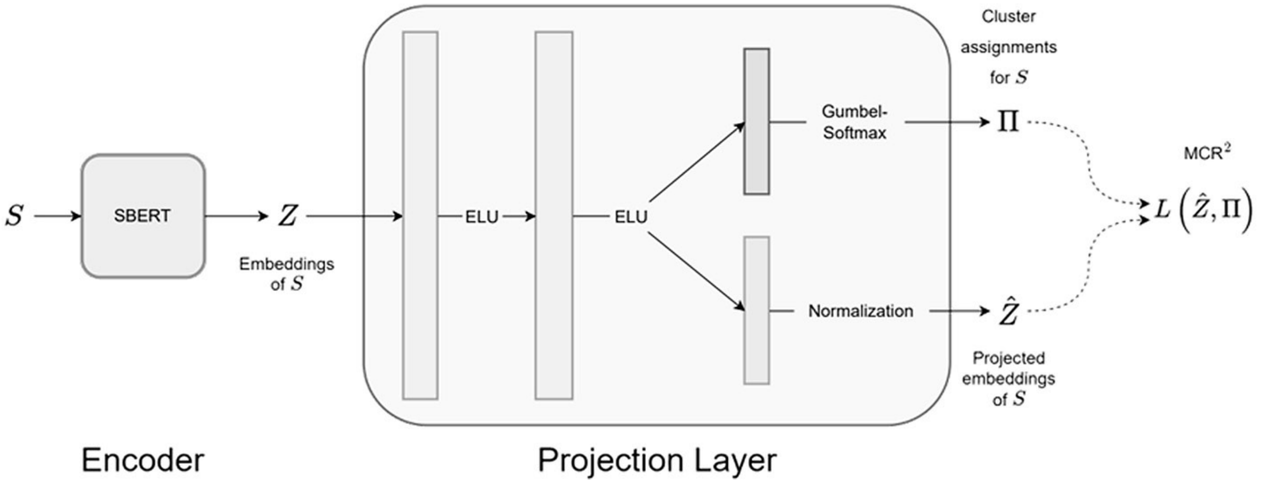

*Figure 1.* The SBERT augmented with MCR$^2$ projection layer.

### 2.2.1. Sentence encoder

BERT and its variants have achieved a new state-of-the-art performance in sentence pair regression and classification tasks [4]. However, one drawback is that BERT requires both sentences to be processed through the network, resulting in a computation overhead. This can make simple tasks, such as identifying similar sentence pairs in large datasets, a costly process. To address this, SBERT [7] was introduced as a modification of the BERT network. SBERT employs a Siamese network that can derive semantically meaningful sentence representations.

The SBERT model incorporates BERT as a pre-trained encoder and utilizes a pooling layer to compute the sentence representation as an average of hidden states from the last layer of BERT. It is trained on a combination of the SNLI [23] and MultiNLI [24] datasets, enabling it to capture the semantic nuances present in sentence pairs effectively. This approach significantly enhances the efficiency of finding similar sentence pairs while still maintaining strong performance.

### 2.2.2. Projection Layer

Based on Li *et al.*'s approach [20], we utilize the aforementioned SBERT as the backbone of our architecture, which incorporates two linear heads responsible for generating features and cluster logits. The features produced by the first head are further normalized to the unit sphere, while the clusters are learned from the provided pairs of similar sentences. Refer to Figure 1 for a detailed representation of the entire architecture, which comprises the SBERT model as the encoder and a feed-forward neural network serving as the projection layer. The projection layer consists of two heads. The first head is a single linear layer responsible for gathering cluster information and applying Gumbel-Softmax, as explained in Huijben *et al.* [25]. Meanwhile, the second head, also a single linear layer, generates features, which are then normalized to have zero mean and unit variance. For this purpose, we employ the ELU activation function due to its advantageous properties, including negative value support, speeding up learning, and noise-robust deactivation state, as demonstrated in Clevert *et al.* [26]. In the smaller-scale computer systems and embedded systems, ELU can act as a compute-effective alternative to batch normalization.

## 2.3. Knowledge Distillation with Compression

Using the idea from Zhao *et al.* [18], we finetune a smaller pre-trained sentence encoder with a projection layer to produce compact representations while mimicking a large pre-trained language augmented with MCR$^2$ trained projection layer to retain the sentence representation quality. More formally, let $x \in S$ denote input to the teacher model $f_t$, and the projection layer be defined as a (learnable) function $\pi_t$: $\mathbb{R}^{d_t} \rightarrow \mathbb{R}^{\hat{d}}$ that takes the output from $f_t$ of dimension $d_t$ and outputs compressed sentence representation $\hat{z}^t = (\pi_t ( f_t(x)) \in \mathbb{R}^{\hat{d}}$. Moreover, we consider a student model, that is a pre-trained smaller sentence encoder as $f_s$ augmented with projection layer $\pi_s$, and we want to distill knowledge from compressed sentence representations of a teacher model to compressed sentence representations of student model. To be more precise, we want to compute compressed sentence representation $\hat{z}^s =(\pi_s( f_s(x)) \in \mathbb{R}^{\hat{d}}$ such that the $\hat{z}_s$ is close to $\hat{z}_t$. To do so, we fine-tune student model $f_s$ (together with $\pi_s$) so that the MSE loss is minimized. The MSE loss is represented by the following equation:

$$\mathcal{L} = \frac{1}{M}\sum_{i=1}^{M}\left\|\hat{z}_i^s - \hat{z}_i^t\right\|_2^2, \tag{7}$$

where $M$ is the total number of sentences during fine-tuning. Note that during the fine-tuning of $f_s$ the parameters of $f_t$ and $\pi_s$ are fixed. To total process of knowledge distillation is shown in Figure 2.

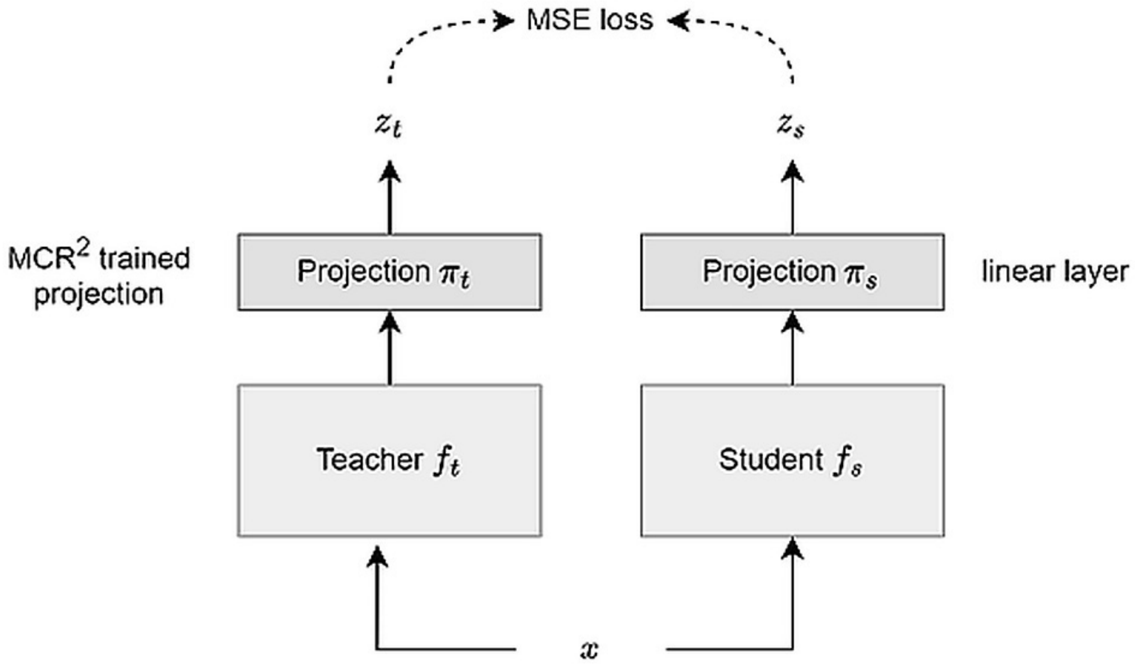

*Figure 2.* Compressing sentence representation by knowledge distillation and MCR$^2$ trained projection using the loss function defined in (7).

## 3. Experiments

Our experiments were conducted in two ways: first, we tested how much information can be preserved by compressing only the sentence representations, and then, we did the same by compressing both model (knowledge distillation) and sentence representations.

### 3.1. Experimental Setup

We utilized StackExchange duplicate questions from CQADupStack [27] to train our model as title/title pairs. The Sentence Transformers pipeline (https://github.com/UKPLab/sentence-transformers) was employed with default settings, 256 batch size, and 50 epochs for SBERT and the projection layer. The pre-trained SBERT models used were the frozen backbone *all-mpnet-base-v2* and distilled model *all-MiniLML6-v2* [28], referred to as MPNET and MiniLM, respectively. The MPNET model maps sentences and paragraphs to a 768-dimensional dense vector space. Despite its 420 MB model size, it consistently provides high-quality embeddings. In contrast, The MiniLM model efficiently maps sentences to a 384-dimensional dense vector space. Despite being 5 times faster than MPNET, it maintains high-quality embeddings with a compact model size of 80 MB. Both models are suitable for semantic search and clustering. The only trained part was the projection layer, where we tried several values of hyperparameter $\lambda$ from equation (4) and found that $\lambda = 2000$ gives best results for dimensions 50 and 100, and $\lambda = 4000$ for all other dimensions. Our model's performance was evaluated on various downstream NLP tasks, starting with clustering-oriented semantic retrieval tasks. Additionally, we demonstrated the effectiveness of the computed low-dimension sentence representations on other semantic benchmarks. The dimension sizes were motivated by experimental observations of suitable word vector sizes from Patel *et al.* [29] and Li *et al.* [17], establishing a connection between word vectors and sentence embeddings. For standard textual similarity, sentiment analysis, and question-type classification tasks, we used available datasets from the SentEval evaluation toolkit [15] for sentence embeddings. Dataset descriptions can be found in Conneau *et al.* [15] and the references therein. During knowledge distillation, we employed *nli-mpnet-base-v2* as the teacher model and *all-MiniLM-L6-H384-uncased* as the student model.

All experiments were conducted on the following hardware setup: AMD Ryzen Threadripper 3990X 64Core Processor @ 4.3GHz, Nvidia GeForce RTX 3090 GPU, CUDA 11.6 with PyTorch implementation 1.9.1.

### 3.1.1. Semantic Retrieval (SR) Task

The semantic retrieval (SR) task aims to identify all sentences in the retrieval corpus that share semantic similarity with the query sentence. The fundamental approach involves computing sentence embeddings for both the retrieval corpus and the query sentence. The objective is to locate the nearest points in the retrieval corpus embedding space to the given query. To expedite this process, Johnson *et al.* [30] propose the clustering of sentences in the retrieval corpus embedding space into $k$ clusters. This allows for query sentences to efficiently find the closest cluster of sentences, thus speeding up the retrieval process.

For the evaluation of our method, we employ the Quora Duplicate Question Dataset (https://www.kaggle.com/datasets/sambit7/first-quora-dataset). This dataset contains 500k sentences with over 400k annotated question pairs, indicating whether they are duplicates or not. By utilizing this dataset, we can effectively assess the performance of our approach in the SR task.

### 3.1.2. Semantic Textual Similarity (STS) Task

In the natural language processing, the semantic textual similarity (STS) task stands as one of the fundamental baseline benchmarks. This task involves qualitatively assessing the semantic similarity between two sentences, or text snippets. To evaluate our model's performance, we utilize the cosine similarity (1) between the embeddings of sentence pairs. The evaluation is carried out on standard STS tasks, including STS 2012-2016 and STS Benchmark from SentEval. These datasets are annotated with similarity scores ranging from 0 to 5, indicating the level of semantic relatedness between sentence pairs. The evaluation process utilizes Spearman rank correlation, which measures the quality of correlation between the calculated similarity scores and the human-labeled similarity scores. The Spearman rank correlation value falls between $-1$ and 1, and it will be high if the ranks of the predicted similarities closely align with the ranks of human labels.

### 3.1.3 Sentence Classification (SC) Task

Sentiment classification tasks involve the assignment of sentiment scores to text snippets, categorizing them into two or more sentiment classes, which typically include negative, positive, neutral, or variations in-between. Datasets like SST, SUBJ, CR, and MR serve as standard benchmarks for sentiment analysis. Another example of a sentence classification task is the assignment of a question type to a given question, as seen in the TREC task. On the other hand, the paraphrase detection problem, such as MRPC, involves classifying whether one sentence is a paraphrase of another. The MPQA dataset exemplifies an opinion classification task. For these benchmarks, the performance metric is measured in terms of accuracy. All these datasets are readily available in the SentEval toolkit.

## 3.2. Knowledge Distillation

For the last part, we compare how MCR$^2$ trained projection approach compares to the results of Homomorphic Projective Distillation from Zhao *et al.* [18] by recreating their experiments and evaluation.

# 4. Results

In this section, we present a comparison of our method as both a clustering and compression algorithm. Table 1 displays the performance of our clustering approach, where we benchmark it against the $k$-means algorithm (implemented in the *scikit-learn* Python package). The comparison includes time performance metrics, such as the time taken for encoding vectors, clustering, and the overall processing time. Moving on to the second part, we evaluate our sentence representation compression in semantic-relatedness tasks. The results are reported in Tables 1, 2, and 3. The model names in these tables follow a structured format, comprising the sentence encoder model name, the projection method used (MCR$^2$ or PCA), the projection dimension, and optionally, whether k-means is employed. The default embedding size is denoted in parentheses.

## 4.1. Results on SR Tasks

We assess the clustering capability of our projection layer in comparison to k-means clustering within the retrieval space of sentence embeddings. For this evaluation, we assign the query sentence to a cluster of semantically related sentences and determine whether the ground truth duplicate belongs to that cluster. The results are reported as accuracy scores. In our experiments, we tested out several number of clusters (32,64, 128, 512, 1024 and 2048) and compared them to the overall benchmark results. Smaller cluster numbers had performance degradation, but greater cluster numbers incurred longer computational time. In the end, we choose cluster number upper boarder of 128, which has good performance and acceptable computational time. Table 1 and Figure 3 present accuracy scores and overall computation time (including encoding of sentence embeddings and clustering) based on various embedding sizes and model types (MCR$^2$ with implicit clustering or $k$-means). Our method demonstrates comparability to the $k$-means algorithm up to a certain dimension. For dimensions less than 200, $k$-means performs slightly better as we did not extensively optimize $\lambda$ values (suggested values of $\lambda$ are from [19]). Notably, our method computes clusters during inference, significantly speeding up the process compared to using the $k$-means algorithm. Additionally, our projection layer, when used as a non-linear mapper without dimensionality

*Table 1*. Semantic Retrieval (SR) tasks.

| model | accuracy | encoding | time<br>clustering | total |
|---|---|---|---|---|
| all-mpnet-base-v2 + MCR50 | 0.562 | 00:04:15 | – | 00:04:15 |
| all-mpnet-base-v2 + MCR100 | 0.545 | 00:04:15 | – | 00:04:15 |
| all-mpnet-base-v2 + MCR200 | 0.645 | 00:04:16 | – | 00:04:16 |
| all-mpnet-base-v2 + MCR300 | 0.632 | 00:04:17 | – | 00:04:17 |
| all-mpnet-base-v2 + MCR50 + kmeans | 0.671 | 00:04:15 | 00:07:08 | 00:11:23 |
| all-mpnet-base-v2 + MCR100 + kmeans | 0.650 | 00:04:15 | 00:07:22 | 00:11:37 |
| all-mpnet-base-v2 + MCR200 + kmeans* | 0.635 | 00:04:16 | 00:09:08 | 00:13:24 |
| all-mpnet-base-v2 + MCR300 + kmeans | 0.631 | 00:04:17 | 00:11:09 | 00:15:26 |
| all-mpnet-base-v2 + kmeans (768)** | 0.648 | 00:04:17 | 00:59:57 | 01:04:12 |
| all-mpnet-base-v2 + MCR768 + kmeans | 0.630 | 00:04:17 | 00:18:15 | 00:22:32 |

reduction, retrofits the sentence embeddings, resulting in faster convergence of the *k*-means algorithm, as shown in the last row of Table 1. In the context of Semantic Retrieval (SR) tasks, the use of the *all-mpnet-base-v2* SBERT model with MCR$^2$ projection to dimension 200 yielded the highest accuracy without requiring additional clustering time, as seen in the same setup with *k*-means (denoted with $*$). In contrast, clustering baseline sentence embeddings from SBERT with *k*-means (denoted with $**$) took almost an hour.

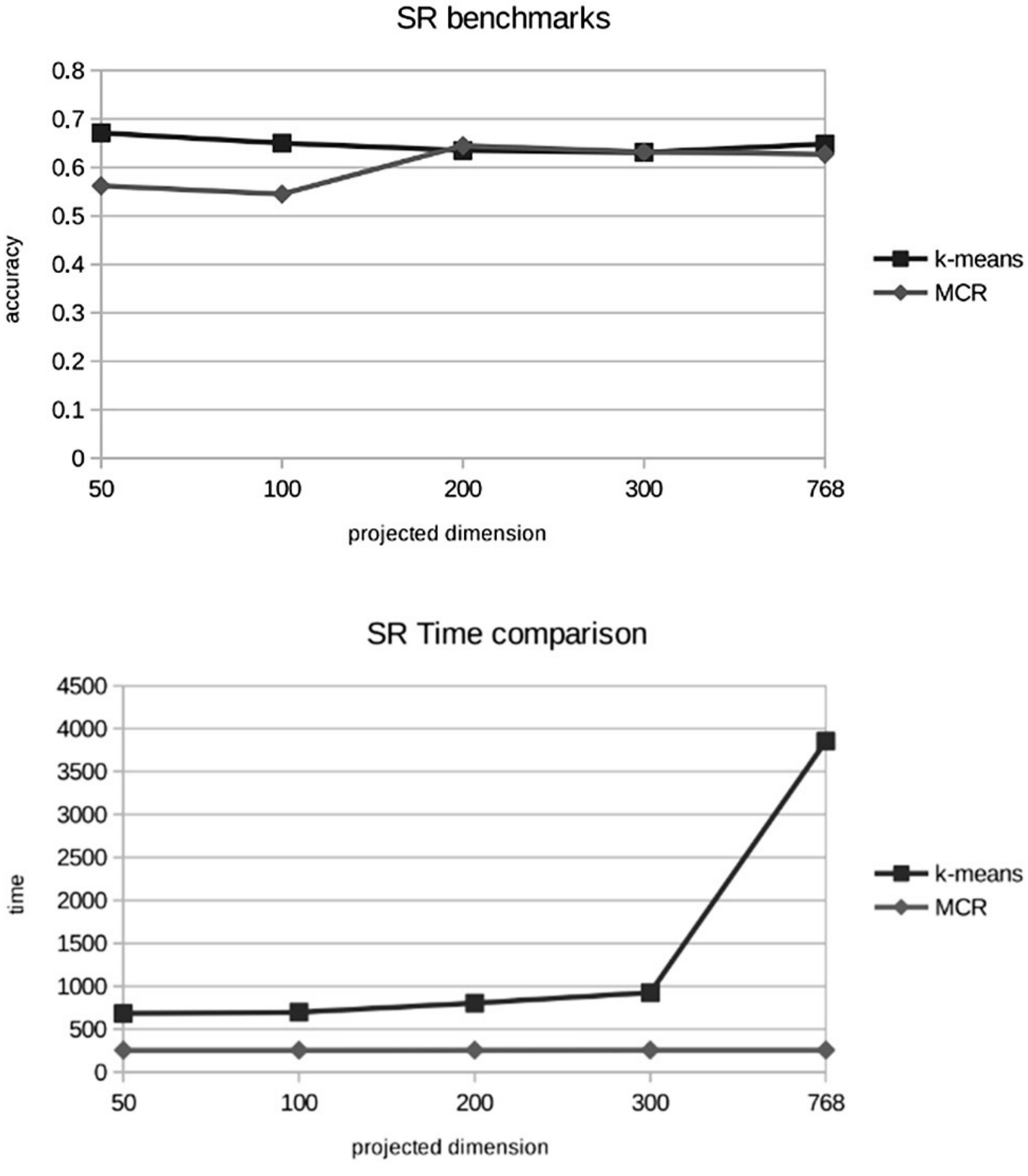

*Figure 3.* Performance comparison on SR tasks.

### 4.2. Results on STS Tasks

Results in Table 2 present the outcomes for the baseline model (MPNET) and the distilled model (MiniLM) when combined with the projection layer (MCR$^2$) at different embedding sizes. A remarkable finding is that reducing the sentence embedding dimension to as low as 6% of the original size results in a relative error of up to 13% in Spearman rank correlation. This observation indicates the efficacy of the projection layer in preserving the cosine distance in the lower-dimensional space, thereby retaining the neighborhood of points and minimizing performance degradation. This trend remains consistent across all STS benchmarks for both the baseline and distilled models. For a visual representation of the relative error in Spearman rank correlation coefficient concerning the projection dimension, please consult the first column of Figure 4, which provides insights for both model variants. Across multiple Semantic Textual Similarity (STS) tasks, the baseline models exhibit the best results (indicated in bold) for the Spearman rank correlation coefficient. However, it's worth noting that we observe comparable outcomes with other models as well.

*Table 2*. Semantic Textual Similarity (STS) tasks.

| **model** | **STSb** | **STS12** | **STS13** | **STS14** | **STS15** | **STS16** |
|---|---|---|---|---|---|---|
| all-mpnet-base-v2 + MCR50 | 0.749 | 0.666 | 0.739 | 0.713 | 0.754 | 0.768 |
| all-mpnet-base-v2 + MCR100 | 0.788 | 0.696 | 0.782 | 0.753 | 0.791 | 0.793 |
| all-mpnet-base-v2 + MCR200 | 0.818 | 0.712 | 0.812 | 0.779 | 0.819 | 0.816 |
| all-mpnet-base-v2 + MCR300 | 0.821 | 0.718 | 0.817 | 0.783 | 0.827 | 0.823 |
| **all-mpnet-base-v2 (768)** | 0.836 | 0.722 | 0.821 | 0.790 | 0.838 | 0.831 |
| all-MiniLM-L6-v2 + MCR50 | 0.752 | 0.654 | 0.690 | 0.682 | 0.741 | 0.737 |
| all-MiniLM-L6-v2 + MCR100 | 0.778 | 0.685 | 0.742 | 0.721 | 0.780 | 0.777 |
| all-MiniLM-L6-v2 + MCR200 | 0.810 | 0.705 | 0.773 | 0.751 | 0.813 | 0.792 |
| all-MiniLM-L6-v2 + MCR300 | 0.813 | 0.710 | 0.780 | 0.759 | 0.826 | 0.800 |
| **all-MiniLM-L6-v2 (384)** | 0.824 | 0.711 | 0.790 | 0.772 | 0.838 | 0.812 |

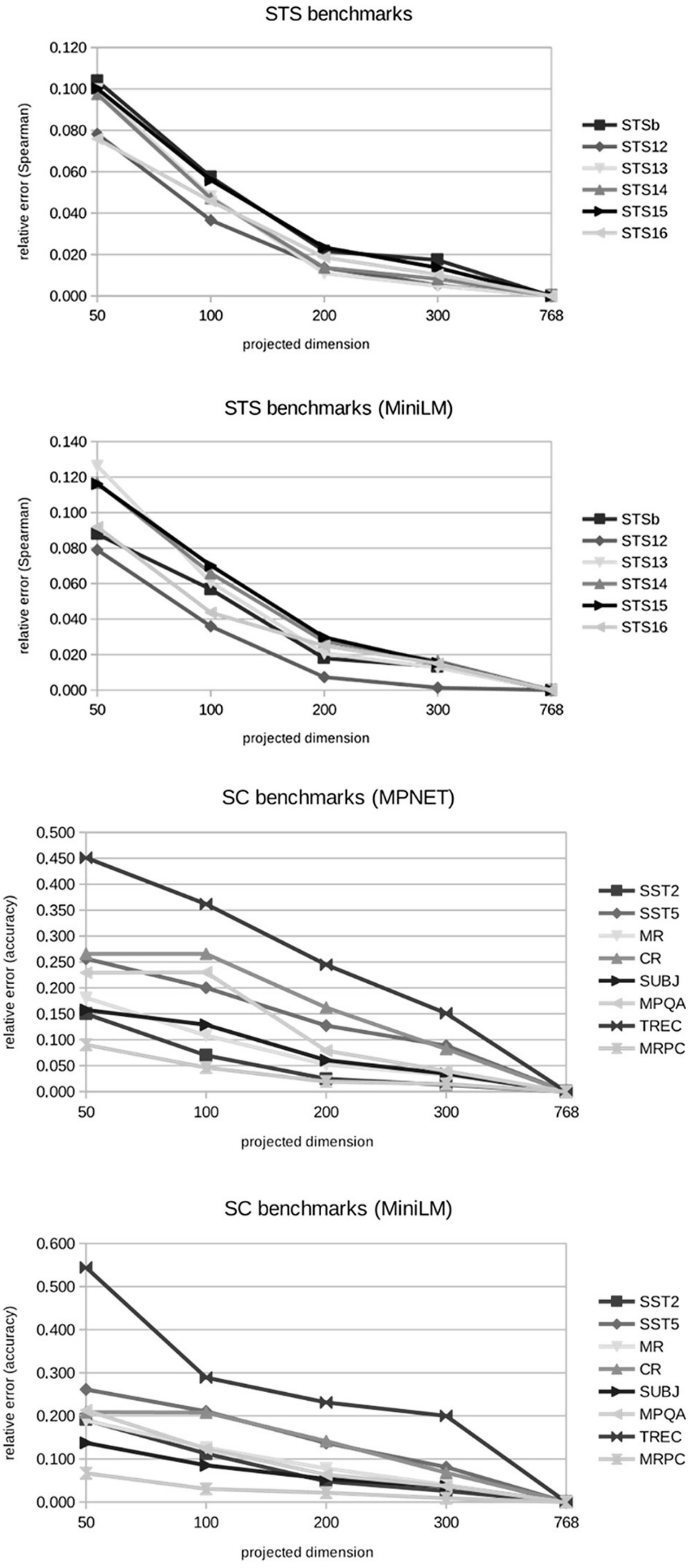

*Figure 4.* Comparing the performance of Semantic Textual Similarity (STS) and Sentence Classification (SC) tasks..

### 4.3. Results on SC Tasks

As presented in Table 3, across multiple Sentence Classification (SC) tasks, the baseline models achieve the best results (indicated in bold) in terms of accuracy on various benchmarks. Nevertheless, it is essential to note that our method also produces comparable results in these tasks. Per-sentence classification tasks, such as SST2 and MRPC, demonstrate relatively less performance degradation compared to per-token sentence classification problems like MPQA and per-sentence multi-classification tasks like TREC. The reason behind this difference lies in the fact that fine-grained semantics in per token and multi-classification tasks are not preserved as effectively during projection. In the most challenging scenarios, the performance degradation reached up to 45% for the baseline model and up to 60% for the distilled model when the projected dimension was reduced to 6% of the original embedding size. For a visual representation of the relative error in accuracy concerning the projected dimension, please refer to the second column of Figure 4, which illustrates the results for both the baseline and distilled models.

*Table 3*. Sentence Classification (SC) tasks.

| **model** | **SST2** | **SST5** | **MR** | **CR** | **SUBJ** | **MPQA** | **TREC** | **MRPC** |
|---|---|---|---|---|---|---|---|---|
| all-mpnet-base-v2 + MCR50 | 75.45 | 36.43 | 69.67 | 63.76 | 79.16 | 68.84 | 51.6 | 68.29 |
| all-mpnet-base-v2 + MCR100 | 82.54 | 39.19 | 75.85 | 63.76 | 81.86 | 68.77 | 60.0 | 71.59 |
| all-mpnet-base-v2 + MCR200 | 86.55 | 42.76 | 80.62 | 72.77 | 88.28 | 82.27 | 71.0 | 73.62 |
| all-mpnet-base-v2 + MCR300 | 87.59 | 44.66 | 82.33 | 79.71 | 90.73 | 85.76 | 79.8 | 73.97 |
| **all-mpnet-base-v2 (768)** | 88.74 | 49.00 | 85.05 | 86.84 | 93.97 | 89.32 | 94.0 | 73.16 |
| all-MiniLM-L6-v2 + MCR50 | 65.95 | 31.76 | 61.61 | 63.76 | 79.19 | 68.77 | 41.0 | 66.49 |
| all-MiniLM-L6-v2 + MCR100 | 72.27 | 33.94 | 66.38 | 63.82 | 83.97 | 76.58 | 64.0 | 67.13 |
| all-MiniLM-L6-v2 + MCR200 | 77.54 | 37.10 | 70.06 | 69.17 | 86.87 | 81.83 | 69.2 | 66.67 |
| all-MiniLM-L6-v2 + MCR300 | 79.35 | 39.50 | 72.95 | 75.07 | 88.47 | 84.13 | 72.0 | 68.06 |
| **all-MiniLM-L6-v2 (384)** | 81.44 | 42.99 | 75.98 | 80.56 | 91.80 | 87.38 | 90.0 | 72.12 |

## 4.4. Results with Knowledge Distillation with Compression

For comparison, we used code from [18] to evaluate the results when using MPNET and MiniLM as teacher and student models, respectively. The same training regime and evaluation as in [18] was followed. Following the ideas described earlier in Section 2.3 we have that projection layer $\pi_t$ is MCR$^2$ or PCA[2] and it is trained and used to compute compressed sentence embeddings. On the other hand, projection layer $\pi_s$ is just one linear layer followed by tanh(.) activation function. Comparison of our methods is given in Table 4. On all STS benchmarks MCR$^2$ showed a slight improvement of up to 1% on average. We recreated experiments with the available code from [18] using only projection and not whitening due to the missing implementation.

The presented results demonstrate that our method outperforms the PCA approach, albeit by a narrow margin. While PCA is a more straightforward technique, our method offers the advantage of ad-hoc data clustering. In other words, we believe that users can fine-tune the pre-trained MCR$^2$ projection layer with the student model to compute sentence clusters during the compression process. This capability opens up new possibilities for practical applications, as it allows for tailored clustering of sentence representations to meet specific needs. Exploring this aspect will be the subsequent phase of our future research efforts.

*Table 4*. Results of knowledge distillation from the *nli-mpnet-v2* teacher model to the *MiniLM-L6-H384-uncased* student model using either the MCR$^2$ or PCA projection technique.

| **model** | **STS12** | **STS13** | **STS14** | **STS15** | **STS16** | **STS-B** | **SICK-R** | **Avg.** |
|---|---|---|---|---|---|---|---|---|
| MCR$^2$ (64 dim) | 73.05 | 84.09 | 79.4 | 85.43 | 80.04 | 83.52 | 77.46 | 80.43 |
| MCR$^2$ (128 dim) | 73.57 | 83.9 | 79.88 | 85.63 | 81.05 | 83.22 | 76.1 | 80.48 |
| MCR$^2$ (256 dim) | 73.1 | 83.35 | 79.13 | 85.04 | 82.15 | 82.91 | 76.79 | 80.35 |
| MCR$^2$ (384 dim) | 72.57 | 81.83 | 77.86 | 84.17 | 81.19 | 82.45 | 76.92 | 79.57 |
| PCA (64 dim) | 73.04 | 83.73 | 78.93 | 83.33 | 79.07 | 83.3 | 78.28 | 79.95 |
| PCA (128 dim) | 71.39 | 82.45 | 78.24 | 84.65 | 78.85 | 82.33 | 78.42 | 79.48 |
| PCA (256 dim) | 71.36 | 82.65 | 78.2 | 84.65 | 79.21 | 82.55 | 78.36 | 79.57 |
| PCA (384 dim) | 70.94 | 82.06 | 77.6 | 84.41 | 78.7 | 82.04 | 78.31 | 79.15 |

[2] It stands for Principal Component Analysis and is a statistical technique used to reduce the dimensionality of data while preserving its variance; see [30] for more details.

## 5. Conclusion

SBERT (bi-encoders) is a practical choice for real-life scenarios due to its simplicity and ease of use, but it's essential to recognize that its fixed-sized sentence embeddings may not always be optimal. In situations where variable-length context matters, such as document summarization or long-text understanding, the loss of context information in fixed-sized embeddings could be detrimental. Additionally, tasks requiring fine-grained positional awareness, like question answering with context, may benefit more from BERT's cross-encoder approach, which considers the entire input sequence. Furthermore, for complex tasks demanding more than sentence-level features—such as coreference resolution or discourse analysis—BERT's cross-encoder, with its ability to capture inter-sentence dependencies, remains a robust choice.

Despite the interesting properties, the computational cost of MCR$^2$ grows with the number of clusters. Specifically, evaluating and differentiating the log-determinant terms for each cluster can become prohibitively expensive when dealing with large-scale datasets or a high number of clusters. This complexity can slow down training and inference times significantly. MCR$^2$ relies on accurate covariance estimates, *i.e.* in scenarios where the data is noisy or the covariance matrices are ill-conditioned, the performance of MCR$^2$ may degrade. While MCR$^2$ aims to improve discriminative representations, it doesn't guarantee better generalization. Overfitting can still occur if the model focuses too much on the training data's specific structure rather than capturing more robust features. MCR$^2$ introduces additional hyperparameters (*e.g.*, regularization terms) that need careful tuning. Finding the right balance between maximizing coding rate differences and preventing overfitting can be challenging. The effectiveness of MCR$^2$ can vary across different domains and tasks. It might work exceptionally well for certain types of data (textual data, like in our case) but less effective for others.

This paper showcased the efficacy of the MCR$^2$ technique in obtaining lower-dimensional embeddings for sentence representation, resulting in faster semantic retrieval tasks and reducing the size by up to 70% of the original. Importantly, we established that these embeddings are comparable to SBERT results on standard semantic NLP benchmarks. The projection layer's clustering ability allowed us to efficiently cluster sentences without incurring additional time costs, further reducing the sentence representation to a reasonable dimension size without significantly compromising important semantic features. Moreover, we employed MCR$^2$ as a method to distill and compress knowledge from large sentence encoders to smaller ones, achieving comparable results to state-of-the-art techniques. This highlights the potential of our approach in deploying AI models on smaller-scale computer systems. By leveraging the MCR$^2$ technique, our research offers valuable insights into enabling efficient and effective applications of AI models in various scenarios, particularly on constrained computing platforms. We hope that our findings will inspire new possibilities and applications in the field of natural language processing and encourage further advancements in the realm of sentence embedding compression and knowledge distillation.

## References

[1] Y. Bengio *et al.*, "A Neural Probabilistic Language Model", J. Mach. Learn. Res., vol. 3, pp. 1137–1155, 2003.

[2] T. Mikolov *et al.*, "Distributed Representations of Words and Phrases and their Compositionality", arXiv, 2013. http://dx.doi.org/10.48550/ARXIV.1310.4546

[3] J. Pennington *et al.*, "GloVe: Global Vectors for Word Representation", in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar: Association for Computational Linguistics*, 2014, pp. 1532–1543. http://dx.doi.org/10.3115/v1/D14-1162

[4] J. Devlin *et al.*, "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), Minneapolis, Minnesota: Association for Computational Linguistics*, 2019, pp. 4171–4186. http://dx.doi.org/10.18653/v1/N19-1423

[5] M. E. Peters *et al.*, "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), New Orleans, Louisiana: Association for Computational Linguistics*, 2018, pp. 2227–2237. http://dx.doi.org/10.18653/v1/N18-1202

[6] H. Aldarmaki and M. Diab, "Evaluation of Unsupervised Compositional Representations", arXiv, 2018. http://dx.doi.org/10.48550/ARXIV.1806.04713

[7] N. Reimers and I. Gurevych, "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), K. Inui, J. Jiang, V. Ng, and X. Wan, Eds., Hong Kong, China: Association for Computational Linguistics*, 2019, pp. 3982–3992. http://dx.doi.org/10.18653/v1/D19-1410

[8] S. S. Saha *et al.*, "Machine Learning for Microcontroller-Class Hardware: A Review", *IEEE Sensors Journal*, vol. 22, no. 22, pp. 21362–21390, 2022. http://dx.doi.org/10.1109/jsen.2022.3210773

[9] Inc. Modzy, "Build Edge AI Solutions Faster." [Online]. Available: https://www.modzy.com/

[10] D. Ševerdija *et al.*, "Compressing Sentence Representation with Maximum Coding Rate Reduction", in *Proc. of the 2023 46th MIPRO ICT and Electronics Convention (MIPRO)*, 2023, pp. 1096–1101. http://dx.doi.org/10.23919/MIPRO57284.2023.10159709

[11] R. Kiros *et al.*, "Skip-Thought Vectors", in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, in NIPS'15. Cambridge, MA, USA: MIT Press*, 2015, pp. 3294–3302.

[12] L. Logeswaran and H. Lee, "An Efficient Framework for Learning Sentence Representations", *CoRR*, vol. abs/1803.02893, 2018, [Online]. Available: http://arxiv.org/abs/1803.02893

[13] M. Pagliardini *et al.*, "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)*, M. Walker, H. Ji, and A. Stent, Eds., New Orleans, Louisiana: Association for Computational Linguistics, 2018, pp. 528–540. http://dx.doi.org/10.18653/v1/N18-1049

[14] P. Bojanowski *et al.*, "Enriching Word Vectors with Subword Information", arXiv preprint arXiv:1607.04606, 2016.

[15] A. Conneau and D. Kiela, "SentEval: An Evaluation Toolkit for Universal Sentence Representations", in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan: European Language Resources Association (ELRA), 2018. [Online]. Available: https://aclanthology.org/L18-1269

[16] T. Gao *et al.*, "SimCSE: Simple Contrastive Learning of Sentence Embeddings", arXiv, 2022. http://dx.doi.org/10.48550/arXiv.2104.08821

[17] B. Li *et al.*, "On the Sentence Embeddings from Pre-trained Language Models", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), Online: Association for Computational Linguistics*, 2020, pp. 9119–9130. http://dx.doi.org/10.18653/v1/2020.emnlp-main.733

[18] X. Zhao *et al.*, "Compressing Sentence Representation for Semantic Retrieval via Homomorphic Projective Distillation", arXiv, 2022. http://dx.doi.org/10.48550/arXiv.2203.07687

[19] Y. Yu *et al.*, "Learning Diverse and Discriminative Representations via the Principle of Maximal Coding Rate Reduction", in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, in NIPS'20. Red Hook, NY, USA: Curran Associates Inc., 2020.

[20] Z. Li *et al.*, "Neural Manifold Clustering and Embedding", arXiv, 2022. http://dx.doi.org/10.48550/arXiv.2201.10000

[21] T. M. Cover and J. A. Thomas, *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*, USA: Wiley-Interscience, 2006.

[22] Y. Ma *et al.*, "Segmentation of Multivariate Mixed Data via Lossy Data Coding and Compression", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 9, pp. 1546–1562, 2007. http://dx.doi.org/10.1109/TPAMI.2007.1085

[23] S. R. Bowman *et al.*, "A Large Annotated Corpus for Learning Natural Language Inference", in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Lisbon, Portugal: Association for Computational Linguistics*, 2015, pp. 632–642. http://dx.doi.org/10.18653/v1/D15-1075

[24] A. Williams *et al.*, "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), New Orleans, Louisiana: Association for Computational Linguistics*, 2018, pp. 1112–1122. http://dx.doi.org/10.18653/v1/N18-1101

[25] I. A. M. Huijben *et al.*, "A Review of the Gumbel-max Trick and its Extensions for Discrete Stochasticity in Machine Learning", arXiv, 2021. http://dx.doi.org/10.48550/ARXIV.2110.01515

[26] D.-A. Clevert *et al.*, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)", arXiv, 2015. http://dx.doi.org/10.48550/ARXIV.1511.07289

[27] D. Hoogeveen *et al.*, "CQADupStack: A Benchmark Data Set for Community Question-Answering Research", in *Proceedings of the 20th Australasian Document Computing Symposium (ADCS)*, in ADCS '15. New York, NY, USA: ACM, 2015, p. 3:1–3:8. http://dx.doi.org/10.1145/2838931.2838934

[28] N. Reimers and I. Gurevych, "Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation", in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics*, 2020. [Online]. Available: https://arxiv.org/abs/2004.09813

[29] K. Patel and P. Bhattacharyya, "Towards Lower Bounds on Number of Dimensions for Word Embeddings", in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers), Taipei, Taiwan: Asian Federation of Natural Language Processing*, 2017, pp. 31–36. [Online]. Available: https://aclanthology.org/I17-2006

[30] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, 1st ed. Springer, 2007.

*Received:* July 2023
*Revised:* February 2024
*Accepted:* February 2024

*Contact addresses*:
Domagoj Ševerdija
School of Applied Mathematics and Computer Science
University of Osijek
Croatia
e-mail: dseverdi@mathos.hr

Tomislav Prusina
Universität Hamburg
Department of Informatics
Germany
e-mail: tomislav.prusina@uni-hamburg.de

Luka Borozan
School of Applied Mathematics and Computer Science
University of Osijek
Croatia
e-mail: lborozan@mathos.hr

Domagoj Matijević
School of Applied Mathematics and Computer Science
University of Osijek
Croatia
e-mail: domagoj@mathos.hr

Domagoj Ševerdija, Ph.D., is an Assistant Professor and head of the Computer Science and Machine Learning Research Group at the School of Applied Mathematics and Computer Science, J. J. Strossmayer University of Osijek, Croatia. His academic journey began at the same university, where he earned his doctorate in Electrical Engineering, Computer Science, and Information Technology (2007–2013). His scholarly pursuits encompass discrete and combinatorial optimization, machine learning, and computational linguistics, contributing to the advancement of these disciplines through his research and teaching.

Tomislav Prusina holds an MSc in mathematics and is currently pursuing a PhD at the Department of Informatics at the University of Hamburg. He kickstarted his academic path at the University of Osijek, obtaining both his bachelor's and master's degrees between 2018 and 2022. His research interests lie in convex optimization, machine learning, and data structures and algorithms. Outside of his studies, he dedicates his free time to tutoring high school students in competitive programming.

Luka Borozan obtained his PhD in computer science from the University of Zagreb, Faculty of Science, Department of Mathematics, Croatia in 2021. At present, he holds the position of a postdoc at the School of Applied Mathematics and Computer Science, J. J. Strossmayer University of Osijek, Croatia where he is involved in both scientific research and teaching. His primary research area focuses on using combinatorial optimization algorithms in computational molecular biology, alongside contributions to econometrics, theoretical mathematics, and machine learning through various publications.

Domagoj Matijević received his PhD in computer science from the Universität des Saarlandes, Germany in 2007. He is now an Associate Professor and vice dean for teaching and students at the School of Applied Mathematics and Computer Science, J. J. Strossmayer University of Osijek, Croatia. Since 2023 he has been the CEO of Meet Intelligent Innovations, a spin-off company specializing in cutting-edge AI solutions. His research interests range broadly from theoretical to applied algorithmics, including areas like machine learning, combinatorial optimization, and application of machine learning in bioinformatics.