SRBedding: Information Retrieval Embedding Model for Serbian

Selena Milutin, Intern
selena.milutin@gmail.com

Teodora Mihajlov, Intern
teodoramihajlov@gmail.com

Milutin Studen, Mentor
milutin.studen@smartcat.io

Abstract

This technical report describes the training process of SRBedding model, an embedding model for Information Retrieval in the Serbian language. The model is a result of *SmartCat.io* 2024 internship. The model and the data is published under a fully permissive licence. You can find code and data to replicate the model at https://github.com/smartcat-labs/SRBedding. The model and the data can be retrieved from https://huggingface.co/smartcat.

1 Introduction

Text embeddings are a fundamental element of NLP applications, encoding the semantic meaning of text for use in various tasks like clustering, classification, and information retrieval. They are also vital for retrieval-augmented generation (RAG) in chatbots and LLMs, as well as in semantic search (Lewis et al., 2020).

The MTEB benchmark (Muennighoff et al., 2022) is the easiest resource for finding leading open-source embedding models, though most listed models are primarily trained for English. While some multilingual models, like OpenAI's *text-embedding-3-small*, *e5-multilingual*, and *paraphrase-multilingual-MiniLM-L12-v2*, perform well on downstream tasks for low-resource languages, they often fail to capture the subtle linguistic nuances that a monolingual model could.

Developing high-quality, production-ready models for low-resource languages like Serbian is difficult due to limited data and resources. Additionally, evaluation benchmarks such as STS (Cer et al., 2017) and BEIR (Thakur et al., 2021) are rarely available for smaller languages, making model comparison challenging. Without predefined benchmarks for these languages, it is hard to determine which models perform best or whether they surpass the performance of their English counterparts. These factors combined make the development of models for low-resource languages especially challenging. The lack of data, limited availability of evaluation benchmarks, and the difficulty in capturing linguistic nuances all contribute to the complexity of creating effective models for languages like Serbian.

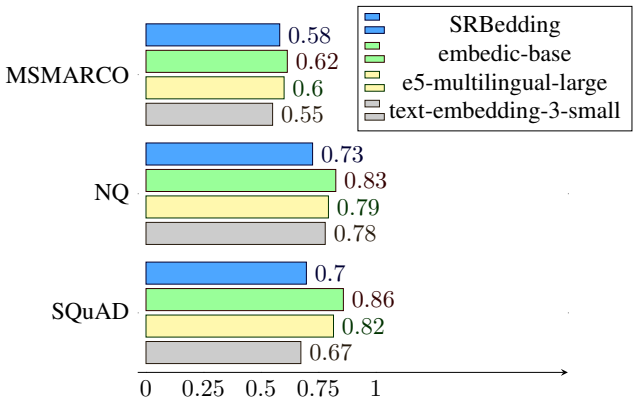

Figure 1: **Text Embedding Models dotNDSG@10.** The comparison of the performance of OpenAI text-embedding-3-small, e5-multilingual-large, embedic-base, and our model, SRBedding on MS MARCO, Natural Quesions (NQ) and SQuAD evaluation benchmarks in the Serbian language. All models display relatively similar performance, with embedic-base outperforming other models on all three evaluations, and SRBedding outperforming OpenAI text-embedding-3-small on two out of three evaluations. The dotNDSG@10 metric values represnted on the x-axis go from 0 to 1, where 1 indicates better performance.

Most Serbian embedding models currently have context lengths of 512 and 768 and are generalist, such as Jerteh-81 and Jerteh-355. Due to linguistic similarities, models like BERTic (Ljubešić and Lauc, 2021) and SRoBERTa have been developed for several South-Slavic languages. Additionally, there are domain-specific models, such as SrBERTa (Bogdanović et al., 2024) for the legal field and BERTic-squad-sr-lat for Question An-

swering (QA).

2 Related Work

State-of-the-art text embedding models are trained by fine-tuning a pre-trained transformer with a contrastive loss objective, often using labeled datasets like MSMarco (Bajaj et al., 2016) and SNLI (Bowman et al., 2015) to generate paired training data. Examples include SBERT (Reimers, 2019), SimCSE (Gao et al., 2022), and SGPT (Muennighoff, 2022). Recent models like E5 (Wang et al., 2022), GTE (Li et al., 2023), BGE (Xiao et al., 2023), and Jina (Günther et al., 2023, 2024) adopt a multi-stage approach, first fine-tuning on large corpora of weakly paired data (e.g., Quora, Reddit) and then on smaller, high-quality labeled datasets, significantly enhancing model quality due to the abundance of weakly paired data (Nussbaum et al.).

The development of Serbian language models encompasses a variety of architectures and training methodologies. Among the notable monolingual models, BERTic (Ljubešić and Lauc, 2021), which is based on the ELECTRA architecture, features 110 million parameters and is trained on a diverse corpus of 8 billion tokens, including 800 million tokens in Bosnian, 5.5 billion in Croatian, 80 million in Montenegrin, and 2 billion in Serbian. The model BERTic-squad-sr-lat is fine-tuned specifically for the Question Answering (QA) task. SRoBERTa, with 120 million parameters, is trained on 18 million tokens from the Leipzig corpus and supports only Cyrillic script. Its variants include SRoBERTa-base, which incorporates the OSCAR corpus, and SRoBERTa-XL, which adds additional corpora like cc-100-hr and cc-100-sr to enhance its training. Other significant models include SrBERT-a, based on the RoBERTa architecture and trained on the OSCAR corpus, and Jerteh-355, a RoBERTa-large architecture with 355 million parameters that supports both Cyrillic and Latin scripts, trained on Jerteh and PDRS 1.0 corpora (Škorić, 2024).

In the realm of generative models, GPT2-srlat is a GPT2-small architecture with 138 million parameters trained on the Serbian Wikipedia, while GPT2-orao, a GPT2-large model with 800 million parameters, is trained on JeRTeH corpora and doctoral dissertations (Škorić et al., 2023). Additionally, the Alpaca-Serbian series includes models with 3 billion and 7 billion parameters, optimized for tasks such as Named Entity Recognition (NER), sentiment analysis, and choice of plausible alternatives (COPA). These models leverage various datasets, including hr500k, ReLDI-sr, ParlaSent, and COPA-sr, to enhance their performance on specific language tasks.

During our internship, *embedic* models for Information Retrieval in Serbian were released on HuggingFace. Currently, we do not know the training data for the *embedic* models. According to HuggingFace, these models are fine-tuned from *multilingual-e5*, come in small, base, and large sizes, and were trained on a 4070ti super GPU. They were evaluated on Information Retrieval, Sentence Similarity, and Bitext Mining tasks. The author translated the STS17 cross-lingual dataset and spent $6,000 on Google Translate for translating four IR datasets into Serbian, far exceeding our data and training costs. Notwithstanding limited time and resources, our model outperforms OpenAI's *text-embedding-3-small* on two of the three benchmarks.

Some notable public datasets for Serbian language include several key corpora. SrpWiki consists of 477,473 articles (70 million words) from Serbian Wikipedia, and SrpKorNews is a corpus of Serbian news texts with 488 million words. SrpKor4Tagging contains literary and administrative texts tagged for PoS, while NARDUS includes 11,624 doctoral dissertations, along with metadata. SrpELTeC provides 100 Serbian novels from 1840-1920, and srWaC is a 2-billion-word web corpus. Additional datasets include OSCAR, mC4, and CLASSLA-sr used for training BERTic, as well as QA datasets like SQuAD-sr and sentiment analysis corpora like ParlaSent.

3 Data Curation

In this section, we outline the procedure for data compilation. The data collection process is conducted separately for each of the three stages in our model development: data for knowledge distillation training, data for supervised training using Multiple Negative Reranking Loss, and data for the final evaluation. As Serbian is a low-resource language, it was challenging to create a representative high-quality dataset. Therefore, compiling data was an iterative, trial-and-error process. Source code for recreating datasets is available here. Datasets are also publicly available on HuggingFace.

3.1 Knowledge Distillation

Knowledge distillation was utilised to transfer a monolingual English embedding model to a multilingual English-Serbian embedding model, as suggested in Reimers and Gurevych (2020). This type of training required parallel English-Serbian data.

The largest portion of the data comes from the Serbian-English parallel corpus MaCoCu-sr-en 1.0, built by crawling the ".rs" and ".srb" top-level domains in 2021 and 2022, with dynamic extensions to other domains (Bañón et al., 2022). The preparation of the MaCoCu-sr-en 1.0 dataset is outlined in Bañón et al. (2022).

The dataset provides both sentence- and document-level parallel data. We used the sentence-level data, with the Serbian text in Latin script. Since no extra metadata was needed, we kept only the parallel English-Serbian sentences, along with their BLEU scores and IDs for traceability. The dataset also contains a title in both Serbian and English for each of Serbian-English sentence pair. To expand the dataset, unique titles were added as sentence pairs. Though BLEU scores for the headlines are not available, they can be readily computed. The final dataset contains 1.8M parallel English-Serbian sentences.

The seond part of the parallel courps is a Wikimatrix dataset, which is part of the OPUS corpus (Tiedemann and Thottingal, 2020), retrieved from HuggingFace, and available here. Since sentences in Serbian were in both Cyrilic and Latin script, we transliterated them to Latin using the serbai library. The final dataset consists of 10k pairs of English-Serbian setences.

We used the NARDUS dataset, the National Serbian repository of doctoral dissertations, accessed via the HuggingFace repository managed by JeRTeh here. The dataset includes abstracts in Serbian and English, which we paired as parallel data, excluding any missing abstracts. To preserve the sentence-level structure across corpora, we extracted only the first sentence from each abstract, assuming it to be a direct translation. This assumption was not verified, but future work could involve validation using BLEU and ROUGE scores, or cosine similarity.

Semantic Text Similarity (STS) dataset was used to evaluate the distilled model. As suggested in Reimers and Gurevych (2020), we translated the second sentence of each sentence pair of the train split of English-English STS dataset. We used the STS bechmark datast available at MTEB HuggigFace. For translation, we used the *gpt-3.5-turbo-0125* model, and translation prompt available here.

3.2 Training Data

The main objective of our training was to finetune an embedding model for information retrieval (IR) in Serbian. Training a model for IR requires a *query-context* pair, where query can be a paragraph, document, sentence, a semantic question, or keyword search, and context can be a document, paragraph, or a sentence. To the best of our knowledge, no such dataset was available in Serbian. Consequently, obtaining training dataset involved several steps - finding the right datasets in Serbian that could be used for IR, creating chunks of the obtained documents, and artificially generating queries.

3.2.1 Retrieving Data

Škorić and Janković (2024) offers an extensive overview of available corpora in the Serbian language. This was our starting point for researching data we will use for model training. First we went through all datasets to filter out the ones that were not publicly available. Next, the size, domain, and format of the dataset. The aim was to find data from different domains that could be sampled for the initial dataset, but that offered enough data that our dataset could be scaled in the future. After going through multiple datasets, ultimately, we opted for:

- **SrpWiki** - dataset comprises 477, 473 articles, i.e. 70 million words of Serbian Wikipedia texts up until 2020. The dataset is a single text file, split into sentences, where each sentence is limited with separators.

- **SrpKorNews** - dataset contains around 468M words. The texts are Serbian news source articles, both automatically and manually post-processed and corrected. The data is organized in 337 documents, where each document contains multiple news articles. Documents are split into sentences and each sentence is delimited.

- **NARDUS** - currently, the NARDUS corpus comprises 13,289 doctoral dissertations. This collection offers extensive information about

each dissertation, including the URL, titles in Serbian and English, and details about the mentors and committee members involved. Additionally, it includes the author's name, publication date, format, publisher (faculty), access rights, and the URI of the license, along with keywords in both languages to enhance searchability.

- **SrpELTeC** - SrpELTeC is a corpus of old Serbian novels for the first time published in the period 1840-1920. After carefully examining the dataset, we established that the language of most novels is too archaic. However, we found that novel "Pop Ćira i pop Spira" written by Stevan Sremac, which is a part of Serbian canon, is high-quality enough to be utilized.

All the aforementioned datasets are curated by the Language Resources and Technologies Society, and available at their HuggingFace profile, JeRTeh.

Initially, all datasets were loaded from Huggingface using the built-in function load_dataset. The next step involved replacing sentence separators to create sentence strings. Once the sentences were extracted, a subset of each dataset was sampled to ensure an adequate number of sentences for chunk creation, with the assumption that each chunk would contain approximately six sentences. To determine the required number of sentences per dataset, we multiplied the desired number of chunks by six. To preserve diversity, a specific number of consecutive sentences were sampled, along with a jump parameter to sample from different sections of the dataset. A complete script for dataset loading and sampling is available here.

The size of the initial training dataset is 10, 000 query-context pairs, distributed as follows: 65% SrpWiki, 10% SrpKorNews, 10% NARDUS, and 5% from a novel in the SrpELTeC collection. Since all data sources provide larger quantities, the dataset can be easily scaled if needed.

3.2.2 Data Chunking

After acquiring the data samples, the subsequent step involved segmenting the texts into appropriately sized chunks. In determining the chunk size, we ensured that each chunk was sufficiently long to preserve semantic information, while remaining compact enough to align with the embedding size limitations of most models, typically 512 tokens.

For this phase, we followed the guidelines outlined in the *Five Levels of Text Splitting* tutorial here. To ensure that the generated chunks conveyed coherent meaning, we employed a Semantic Chunking approach. Initially, sentences are indexed, and a sliding window method with a window size (referred as *buffer_size* in code) of 2 was applied. For each target sentence, we selected the sentence itself, along with the two preceding and two following sentences, to form the initial chunks. This process was carried out separately for each dataset.

However, in the case of Wikipedia and news datasets, articles were not labeled or segmented in a meaningful way. Consequently, if we relied solely on the sliding window chunking step, we risked producing chunks that mixed content from different topics. Since our goal is to train an embedding model for information retrieval, such mixed-topic chunks would introduce noise into the dataset. To mitigate this, the next step involved embedding the chunks. For this, we utilized the OpenAI API with the *text-embedding-3-small* model. Then, cosine distance was measured between the obtained chunk embeddings, a threshold of 0.95 was set, and all similarities above that threshold were taken as the new chunk boundaries. Lastly, not to obtain chunks that are too short or too long, we processed the text chunks, filtering out those with fewer than 50 tokens and splitting those with more than 450 tokens into smaller chunks. Finally, we obtained 11k chunks. The code and an example of data formatting for chunking is available here.

3.2.3 Prompt Engineering

The goal of this step is to create a high-quality dataset of context-query pairs, where the context is a text chunk and the query is a question, statement, or keyword search generated by an LLM. We used OpenAI models and followed their prompt engineering guidelines here for optimal results.

The objective was to generate three queries of varying lengths (short, medium, long) for each chunk, ideally covering different aspects of the text. The model also generated up to 5 keywords per context and assigned a relatedness score from 1 to 5 for each query as a label. We found that keyword generation aided the model to better detect topics in the contexts and improved the quality

of the generated queries. The output was provided in JSON format here.

In the first instance, we used *gpt-3.5-0125* for query generation. The initial prompt framed the task as generating queries for an information retrieval dataset, but this resulted in queries that sometimes included information not stated or implied in the context, a behavior we refer to as ”hallucinations.” Adjusting the prompt to focus on ”fact-checking” by issuing queries to a search engine reduced these hallucinations. However, regardless of the prompt, the query lengths did not fully comply with the specified limits, and the model only produced queries in question form.

In the final iteration, we used *gpt-4o* for query generation to improve results and reduce costs. This switch required prompt adjustments, as we found *gpt-4o* performs better with more direct prompts, while *gpt-3.5-0125* needed more detailed instructions. The main difference observed was the absence of hallucinations in *gpt-4o*. Additionally, *gpt-4o* generated both questions and statements as queries. Scripts with prompts can be found at GitHub.

3.2.4 Query Generation

We first generated a sample of 20 queries from the Wikipedia dataset chunks, to evaluate the query quality and adjust the prompt where needed. Once we were satisfied with the output, we generated queries for the entire 11k chunk dataset. Queries were generated using OpenAI API batch processing. Since we generated queries for 10k examples, we avoided the 24h wait, and retrieved the results in a couple of hours. The code used for batch processing can be found here. To retrieve the batches, use this script. Depending on the size of your data, i.e. the number of requests you send to the model, query generation can last up to 24h. In our case, 11k requests were complete in a little over an hour.

3.3 Evaluation Data

MTEB (Muennighoff et al., 2022) has become the standard benchmark for evaluating embedding models, covering 8 tasks across 56 datasets. It assesses models on tasks such as Classification, Clustering, Reranking, Retrieval, and Semantic Textual Similarity. The MTEB score is a weighted average of the scores for each task. At the moment of the creation of this project, MTEB leaderboard is not available for Serbian.

The most commonly used evaluation benchmark for IR is BEIR (Thakur et al., 2021), which is also incorporated in the MTEB and the MTEB leaderboard. Here, we use only MS MARCO version 1.1 (Bajaj et al., 2016) and Natural Question (NQ) (Kwiatkowski et al., 2019) datasets from BEIR. Due to resource limitaions, we translated the first 8000 samples of the datasets to Serbian. In addition to the two datasets, we use SQuAD (Rajpurkar et al., 2016) for evaluation. SQuAD dataset was already translated to Serbain and available on Kaggle. Note that SQuAD is not a part of the BEIR benchmark.

3.3.1 Retrieving Data

MS MARCO (available here) and NQ (available here) datasets were retrieved from HuggingFace, while SQuAD was downloaded from Kaggle (available here).

- **MS MARCO** - a large-scale resource for training and evaluating information retrieval and question answering models, developed by Microsoft. It contains over 1 million queries and 8.8 million passages. We loaded the dataset from Hugging Face and extracted only the passages, queries, and query IDs, for traceability. Each query is associated with several passages, with one positive passage labeled as 1 and the others as negative, labeled as 0.

- **NQ** - a benchmark for question answering developed by Google. The NQ corpus comprises questions posed by real users and requires QA systems to read and understand complete Wikipedia articles, which may or may not provide the answer. We loaded the dataset, and then extracted long answer candidates from documents based on the provided indices. Finally, we extracted passages, queries, and query IDs, for traceability.

- **SuQAD** - a reading comprehension dataset developed by Stanford. It is comprised of more than 100k Wikipedia articles and crowdsourced questions, where an answer to each question is a segment of text in a corresponding Wikipedia article passage. It was translated to Serbian using an adapted translate-align-retrieve method presented in Cvetanović and Tadić (2023). We

loaded the dataset in Serbian from Kaggle and extracted titles, passages, and queries.

3.4 Prompt Engineering

Model *gpt-3.5-0125* was used for the translation of MSMARCO and NQ datasets, We defined a clear few-shot prompt, where the model was assigned a role of translator, and instructed to first understand the sentence in English, then translate it to Serbian, conveying original context and meaning, and adhering to the rules of Serbian language and grammar. The model is instructed to provide output in JSON format.

3.5 Translation

After obtaining the datasets in the same format, we translated them using the previously outlined prompt. Initially, we translated a sample of 50 examples from the MS MARCO dataset to assess the results and adjust the prompt if necessary. Overall, the model demonstrated strong translation capabilities and adhered well to Serbian language and grammar rules. However, it occasionally misinterpreted polysemous words; for instance, in a passage about squirrels, it translated *arms* as *krila* (eng. *wings*), likely because it recognized the context of animals but overlooked that most squirrel species do not have wings. We were unable to resolve these issues through prompt adjustments. When the translation was assessed as satisfactory, we translated 8000 instances of MS MARCO, and 8000 instances of NQ. Translation was generated using OpenAI API batch processing. The entire pipeline for retrieving datasets, prompt, and generating translations can be found at GitHub repository.

4 Experimental Setup

In this section, we will detail the experimental setup, beginning with an overview of the models employed, followed by a discussion of the various training paradigms explored. The training process followed the best practices given in the HuggingFace blog, available here.

4.1 SBERT

Sentence-BERT (SBERT) is a fine-tuned BERT model using a siamese network architecture to generate fixed-size sentence embeddings (Reimers, 2019). These embeddings enable efficient comparison using cosine similarity or other distance metrics, such as Euclidean distance. SBERT is optimized for tasks like information retrieval, semantic search, and clustering (Reimers, 2019).

SBERT was selected for its ability to significantly reduce the computational cost of sentence-pair comparisons. While traditional BERT models may take hours to identify similar pairs, SBERT completes this in seconds, making it suitable for real-time retrieval of semantically similar documents. This efficiency is achieved by precomputing embeddings for the entire document collection, clustering semantically related documents in vector space.

The SentenceTransformer library was used to implement SBERT. It offers pre-trained models, including SBERT and SRoBERTa, supports task-specific fine-tuning, and simplifies embedding-based tasks like semantic search, clustering, and classification.

4.2 Hard Negative Mining

One of the ways to enhance results is introducing (hard) negatives to your data (overview available here).

In our initial approach, we first embedded all text chunks using *text-embedding-3-small*, then computed cosine similarity to find the most similar ones. From n closest paragraphs we randomly selected 5 as negatives. In the meantime, the Sentence Transformer library released their hard negative mining utility, so we switched out approach to that.

When training the model with negatives, we used TripletLoss function. Here, the data is organized in anchor-positive-negative triplets. In our case, anchor was a search query, positive pair was text chunk corresponding to that query, and negative was a different text chunk. The objective of the function is to get positive examples close to the anchor, and negative examples far away from the anchor. Triplet loss did not provide us meaningful result improvement. Thus, withdrew it from the final training pipeline.

4.3 Cosine Similarity Loss

In Cosine Similarity loss, for each pair of sentences A and B, we obtain embeddings u and v, then compute cosine similarity score between the sentence embeddings, and compare the cosine similarity to a golden, ground-truth score.

Here, we use an SBERT implementation, the CosineSimilarityLoss function, to try and train our model for information retrieval. The model is tasked to find the most similar context for each query. As the golden truth, we normalize the GPT-generated score from 1 to 5 for each query-context pair by dividing it by five. Trainig the model with CosineSimilarity loss yielded poor results, possibly due to the GPT-generated golden scores, which were usually distributed between 3 and 5. Therefore, we gave up on applying this method in our final training setup and will not further discuss its results.

4.4 Knowledge Distillation

In machine learning, knowledge distillation is a process of transferring knowledge from a larger model, referred to as *teacher*, to a smaller model, referred to as *student* (Hinton, 2015) (for a brief overview you can also see here). However, this training paradigm can also be used to transfer a model from one language to another, i.e. to make a monolingual model multilingual, as suggested in Reimers and Gurevych (2020).

This type of training requires a (monolingual) teacher model, and a (multilingual) student model, and a parallel dataset. During the training, the teacher model first embeds the English sentences from the parallel corpus. Then, during inference, the student model embeds both the English and the sentences in another language. MSE loss is computed between the teacher and student embeddings. The aim of the training is two-fold:

1. Aligning vector spaces across languages, i.e. to put the same sentences in different languages close.

2. Adaptation and transfer of vector space properties of the original source language from the teacher model to other languages.

In our case, the teacher model is an English-language embedding model specialized for information retrieval, and the student model is a multilingual embedding model, also specialized for the IR task. The parallel corpus are 1.8M English-Serbian sentence pairs, described in 3.1.

4.4.1 Training

The knowledge distillation training pipeline in this project is based on the sentence transformers pipeline, available here.

First, we tried out the original model setup with best results described in Reimers and Gurevych (2020), where sentence-transformers/paraphrase-distilroberta-base-v2 is the teacher and XML-R is the student model, and to check whether our data is good for distillation training. Since training yielded good results, we moved on to training embedding models focused on information retrieval.

Based on results on MTEB and our hardware, we opted for using the e5 model for knowledge distillation. We tried out the following setups: (a) monolingual intfloat/e5-base-v2 as a teacher and the same model as a student; (b) monolingual intfloat/e5-base-v2 as teacher and a multilingual intfloat/multilingual-e5-base as student. MSELoss was used during the training process in order to reach our goal of bringing closer student and teacher embeddings. As the monolingual-multilingual setup gave better results, we opted for longer training with it.

The final model knowledge distillation training took about 27 hours. It was run on 6 epochs, batch size 16 , gradient accumulation was set to 2, and learning rate to 2e-5. The model was evaluated at every 5000 steps. The distilled model is saved and then trained on the Information Retrieval task. Adding knowledge distillation to our training led to ultimate result improvement on the Information Retrieval (IR) task, in comparison with models that were not distilled, but only trained for IR.

4.5 Supervised Fine-tuning with Multiple Negative Reranking Loss

Once the model was distilled, we fine-tuned it for Information Retrieval with Multiple Negative Reranking Loss (MNR loss). The MNR loss function

4.6 Evaluation

Through model evaluation is crucial for making well-informed design decisions. To evaluate our models after fine-tuning for IR, we utilize commonly used offline Information Retrieval evaluation metrics:

- **Recall@k** - measures how many relevant items were returned (true positives, TP) against how many relevant items exist in the entire dataset, i.e. true positives and false negatives (FN). Order of the returned results do not affect the score.

Table 1: Results of model evaluation on MSMARCO, Natural Questions (NQ) and SQuAD datasets in the Serbian language. The best model overall is presented in underlined and bold, while the results where our model, SRBedding outperformed OpenAI's text-embedding-3-small are presented in bold.

Model name	Dataset	dotRecall@10	dotMRR@10	dotNDCG@10	dot-MAP@100
text-embedding-3-small	MSMARCO	0.936	0.431	0.551	0.434
	NQ	0.876	0.749	0.780	0.753
	SQuAD	0.840	0.622	0.674	0.628
multilingual-e5-large	MSMARCO	0.957	0.487	0.601	0.490
	NQ	0.894	0.761	0.794	0.765
	SQuAD	0.945	0.774	0.816	0.776
embedic-base	MSMARCO	**0.972**	**0.502**	**0.615**	**0.503**
	NQ	**0.917**	**0.796**	**0.826**	**0.799**
	SQuAD	**0.964**	**0.824**	**0.859**	**0.826**
SRBedding	MSMARCO	**0.938**	**0.471**	**0.582**	**0.473**
	NQ	0.835	0.690	0.725	0.695
	SQuAD	**0.860**	**0.646**	**0.698**	**0.652**

- **Mean Reciprocal Rank@k (MRR@k)**- an order-aware metric, which means that, unlike recall@K, returning an actual relevant result at rank one scores better than at rank four. It calculates the average of 1/returned rank of the item for each query.
- **Normalized Discounted Cumulative Gain (NDCG@k)** - an order-aware metric. It calculates a Discounted Cumulative Gain@k (DCG@k) by assigning higher importance to top-ranked items and normalizing it by the Ideal DCG (IDCG), which represents the best possible ranking. The result is a score between 0 and 1, where 1 indicates a perfect ranking.
- **Mean Average Precision@k (MAP@k)** - an order-aware metric. It calculates the average precision for each query, based on the ranked list of retrieved results, and then takes the mean of these averages across all queries. Precision measures how relevant the results are, and MAP rewards systems that return relevant results at higher positions in the ranked list.

More details on the IR evaluation metrics can be found in a Pinecone blog, available here.

5 Results

In Table 1, we present the evaluation results on our benchmark, the MS MARCO, NQ, and SQuAD datasets in Serbian. We compare the performance of our fine-tuned model, SRBedding, against other pretrained models representative for the task. As we can see, all models are outperformed by the *embedic-base* model.

At the point of compiling this report, we do not know what data was used to train the *embedic* models. As per the model description on Huggingface, the models are fine-tuned from multilingual-e5 models, come in three sizes (small, base, large) and were trained on a 4070ti super GPU. The evaluation covered three tasks: Information Retrieval, Sentence Similarity, and Bitext Mining. The author translated the STS17 cross-lingual dataset, and spent $6,000 on the Google Translate API to translate four Information Retrieval evaluation datasets into Serbian, which is significantly more than our data creation and model training costs combined. However, our model outperforms OpenAI's *text-embedding-3-small* on two out of three benchmarks.

Table 2 presents the results of models fine-tuned solely with MNR loss, excluding knowledge distillation. The performance across all models is fairly similar, with *mxbai-rerank-base-v1* slightly outperforming the other two. Additionally, we can see a clear improvement by cca. 10% across all metrics of the *multilingual-e5-base* after knowledge distillation. As previosly mentioned, the *SRBedding* presented in Table 1 is a distilled *multilingual-e5-base* with *e5-base* as teacher.

Table 2: Results of models fine-tuned with MNR loss on the MS MARCO, Natural Questions (NQ), and SQuAD datasets in Serbian. The highest metrics for each benchmark are highlighted in bold, with ties between models indicated by bold and underlining.

Model name	Dataset	dotRecall@10	dotMRR@10	dotNDCG@10	dot-MAP@100
bge-base-en-v1.5	MSMARCO	0.826	**0.393**	**0.496**	**0.398**
	NQ	0.685	0.541	0.576	0.547
	SQuAD	0.694	0.487	0.536	0.495
mxbai-rerank-base-v1	MSMARCO	**0.835**	0.390	**0.496**	0.396
	NQ	0.721	**0.573**	**0.609**	**0.578**
	SQuAD	**0.945**	**0.774**	**0.816**	**0.776**
multilingual-e5-base	MSMARCO	0.830	0.355	0.467	0.363
	NQ	**0.728**	0.519	0.570	0.527
	SQuAD	0.802	0.512	0.581	0.521

6 Training Resources

For query generation, dataset translation, and embeddings, we employed the OpenAI API, specifically using *gpt-3.5-0125* and *gpt-4o* for generation, *gpt-3.5-0125* for translation, and *text-embedding-3-small* for embeddings. We implemented batch processing for query generation and translation, while parallel processing was utilized for embeddings. The OpenAI API usage costs summed up to $153.27.

For training purposes we used JupyterLab in the AWS SageMaker with instance *ml.g5.2xlarge*. One training was also done on *ml.g4dn.12xlarge* with the purpose of multi GPU training. Since multi-GPU training didn't improve results, we opted for using *ml.g5.2xlarge* to lower the training costs. Approximate AWS costs are USD 300.

7 Conclusion

In this project, we fine-tuned an embedding model for the Information Retrieval task in Serbian. Our results demonstrate that, with the right training approach, a model can be trained using relatively small amounts of data and at low cost, yet still outperform models trained with significantly greater resources, such as OpenAI models. We release the model, data, and code under permissive licences to allow for the reproduction of the entire training process.

7.1 Future Work

In the future, we aim to pre-train a generalist Serbian model on the Masked Language Modeling (MLM) task, followed by fine-tuning for Semantic Text Similarity (STS) and Information Retrieval (IR) tasks. We also plan to scale up the data for IR training and explore knowledge distillation using only challenging examples, where translations from English to Serbian involve more complex semantic structures, such as idiomatic expressions or language-specific phrases. Once this training approach is established, we will focus on optimizing hyperparameters and experimenting with optimizers like AdamW.

7.2 Contributions

Selena Milutin contributed to the development of evaluation and training scripts, conducted research related to model training paradigms, contributed to selecting models, loss functions, and evaluation metrics. Teodora Mihajlov contributed to the research, the dataset creation process, and the selection of training and evaluation paradigms and models. Milutin Studen supported the project by mentoring the interns, providing guidance, valuable resources, and ongoing feedback. Special thank you to the rest of *SmartCat.io* (Machine Learning) team, for their support, knowledge sharing, interest in the project, and making us feel at home. We extend our gratitude to *SmartCat.io* for organizing the internship, and for giving us the opportunity and resources to learn and grow, both personally and professionally.

References

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Marta Bañón, Miquel Esplà-Gomis, Mikel L. Forcada, Cristian García-Romero, Taja Kuzman, Nikola Ljubešić, Rik van Noord, Leopoldo Pla Sempere, Gema Ramírez-Sánchez, Peter Rupnik, Vít Suchomel, Antonio Toral, Tobias van der Werff, and Jaume Zaragoza. 2022. MaCoCu: Massive collection and curation of monolingual and bilingual data: focus on under-resourced languages. In *Proceedings of the 23rd Annual Conference of the European Association for Machine Translation*, pages 303–304, Ghent, Belgium. European Association for Machine Translation.

Miloš Bogdanović, Jelena Kocić, and Leonid Stoimenov. 2024. Srberta—a transformer language model for serbian cyrillic legal texts. *Information*, 15(2):74.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada. Association for Computational Linguistics.

Aleksa Cvetanović and Predrag Tadić. 2023. Synthetic dataset creation and fine-tuning of transformer models for question answering in serbian. In *2023 31st Telecommunications Forum (TELFOR)*, pages 1–4. IEEE.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2022. Simcse: Simple contrastive learning of sentence embeddings.

Michael Günther, Louis Milliken, Jonathan Geuter, Georgios Mastrapas, Bo Wang, and Han Xiao. 2023. Jina embeddings: A novel set of high-performance sentence embedding models.

Michael Günther, Jackmin Ong, Isabelle Mohr, Alaeddine Abdessalem, Tanguy Abel, Mohammad Kalim Akram, Susana Guzman, Georgios Mastrapas, Saba Sturua, Bo Wang, Maximilian Werk, Nan Wang, and Han Xiao. 2024. Jina embeddings 2: 8192-token general-purpose text embeddings for long documents.

Geoffrey Hinton. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

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

Nikola Ljubešić and Davor Lauc. 2021. BERTić - the transformer language model for Bosnian, Croatian, Montenegrin and Serbian. In *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, pages 37–42, Kiyv, Ukraine. Association for Computational Linguistics.

Niklas Muennighoff. 2022. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Zach Nussbaum, John X Morris, Brandon Duderstadt, and Andriy Mulyar. Nomic embed: training a reproducible long context text embedder (2024). *arXiv preprint arXiv:2402.01613*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online. Association for Computational Linguistics.

Mihailo Škorić. 2024. Novi jezimodeli za srpski jezik. *Infoteka*, 24.

Mihailo Škorić and Nikola Janković. 2024. New textual corpora for serbian language modeling. *arXiv preprint arXiv:2405.09250*.

Mihailo Škorić, Miloš Utvić, and Ranka Stanković. 2023. Transformer-based composite language models for text evaluation and classification. *Mathematics*, 11(22):4660.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

Jörg Tiedemann and Santhosh Thottingal. 2020. OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, pages 479–480, Lisboa, Portugal. European Association for Machine Translation.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

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