SRBerta—A Transformer Language Model for Serbian Cyrillic Legal Texts

Miloš Bogdanović *, Jelena Kocić and Leonid Stoimenov

Faculty of Electronic Engineering, University of Niš, Aleksandra Medvedeva 14, 18000 Niš, Serbia; tosic.jelena@elfak.rs (J.K.); leonid.stoimenov@elfak.ni.ac.rs (L.S.)

* Correspondence: milos.bogdanovic@elfak.ni.ac.rs; Tel.: +38-118-529-642

Abstract: Language is a unique ability of human beings. Although relatively simple for humans, the ability to understand human language is a highly complex task for machines. For a machine to learn a particular language, it must understand not only the words and rules used in a particular language, but also the context of sentences and the meaning that words take on in a particular context. In the experimental development we present in this paper, the goal was the development of the language model SRBerta—a language model designed to understand the formal language of Serbian legal documents. SRBerta is the first of its kind since it has been trained using Cyrillic legal texts contained within a dataset created specifically for this purpose. The main goal of SRBerta network development was to understand the formal language of Serbian legislation. The training process was carried out using minimal resources (single NVIDIA Quadro RTX 5000 GPU) and performed in two phases—base model training and fine-tuning. We will present the structure of the model, the structure of the training datasets, the training process, and the evaluation results. Further, we will explain the accuracy metric used in our case and demonstrate that SRBerta achieves a high level of accuracy for the task of masked language modeling in Serbian Cyrillic legal texts. Finally, SRBerta model and training datasets are publicly available for scientific and commercial purposes.

Keywords: large language model; legislation; Serbian; BERT

1. Introduction

The proofreading of formal language in official documents is a specific challenge that requires domain knowledge regarding not only the grammatical, lexical, and orthographic rules, but also formal rules used within a domain-specific language. A separate part of the previous problem refers to checking the correct use of formal language when writing legislative texts. Such tasks are delegated to specialists, and domain experts, whose daily work could be facilitated by the development of software tools for these specific purposes.

Language is a unique ability of human beings. Although relatively simple for humans, the ability to understand human language is an extremely complex task for machines. In response, researchers have been working on language modeling for decades. The models created incrementally ranged from statistical language models to neural language models. The research results generated approaches that we use today through artificial intelligence applications. Most applications, as we see them today, rely on only two abilities—generating the probability of the occurrence of a sequence of tokens and generating the probability of the occurrence of the next and/or missing token. The latest ability is a foundation of a particular group of language models, called transformer models, and has demonstrated great potential in solving traditional natural language processing problems and offers a breakthrough that fascinates the audience. Transformer models, based on deep neural networks and trained on huge text corpora, have shown that by scaling the model size, it is possible to achieve additional model capabilities that were not previously available, such as learning in context.
The ultimate goal, which is also the biggest challenge, is for a machine to understand a language. For a machine to learn a particular language, it must understand not only the words and rules used in a particular language but also the context of sentences and the meaning that words take on in a particular context. In the experimental development we carried out, the goal of the language model we developed—SRBerta—was to understand the formal language of Serbian legal documents. SRBerta was envisioned and developed as a part of the preparation for a project that aims to help the digitalization of the archive of legislation in the Republic of Serbia. A significant part of historical legislation material in the Republic of Serbia is currently in the form of scanned documents (an image of the original document), which makes it difficult to process, and almost impossible to query efficiently. In such a situation, SRBerta will be combined with document image analysis (DIA) tools and help automatic digitization in cases of OCR inaccuracy of Cyrillic texts.

In 2018, the Google Research AI team presented the BERT (Bidirectional Encoder Representation from Transformers) artificial neural network architecture [1], setting 2 goals: masked language modeling and next-sentence prediction. Starting from the BERT architecture, in 2019, the Facebook AI team presented RoBERTa (Robustly optimized BERT pretraining approach), a network optimized for the task of masked language modeling [2]. SRBerta was created on the basis of the RoBERTa architecture, whereby the training of the SRBerta network for the task of understanding the formal language of Serbian legislation was carried out in two phases—base model training and fine-tuning. In the rest of this paper, we will present the structure of the model, the structure of the training datasets, the training process, and the evaluation results. For evaluation purposes, we have defined an accuracy metric for our use case, which will also be described. Further, we will prove that our model achieves a high level of accuracy for the task of masked language modeling in Serbian Cyrillic legal texts.

2. Related Work—Resources and Models

Knowing the tasks for which the models are designed, it is not difficult to assume that the basic resource for training language models is text. Given that the size of models is constantly increasing, heterogeneous and mostly publicly available texts are used for model training. Ideally, it is necessary to use as large a corpus of high-quality texts as possible. BookCorpus [3] is a dataset that has been frequently used in the past for model training. It consists of over 11,000 books covering a wide range of topics and genres. Another similar and significant large-scale corpus is Project Gutenberg [4], which consists of over 70,000 literary publications (novels, essays, poetry, drama, history, science, etc.). Project Gutenberg is one of the largest publicly available collections of books. CommonCrawl [5] is a collection of texts whose size is measured in petabytes. This collection was created by indexing and processing content available on the Internet and is probably the most widely used set of texts for training different language models. Since it represents the result of gathering content from the Internet, CommonCrawl contains texts of lower quality, so development teams often resorted to processing texts before training and extracting a subset of texts from the CommonCrawl set that corresponds to a specific purpose. Development teams most often use a mixture of texts in the initial phase of training language models. In practice, this process includes the selection of texts and the processing of texts to create sets that are prepared for a specific model.

Since the text corpora were at hand, the research and development community started creating pre-trained language models. As an example of early models, ELMo (Embeddings from Language Models) [6] was proposed. ELMo used word type embeddings such that each token is assigned a representation that is a function of the entire input sentence. Consequently, it gained the ability to capture context-aware word representations from a pre-trained bidirectional LSTM network. The next big step was made with the introduction of Transformer architecture [1] with self-attention mechanisms—BERT [7]. The success of BERT in pre-training tasks on large-scale unlabeled corpora inspired a large number of subsequent works. Novel transformer-inspired architectures and models, such as GPT-2 [8]
and BART [9], started emerging and proving their efficiency when exploiting their general-purpose semantic features for various NLP tasks. The era of fine-tuning the pre-trained language models and creating large language models was about to begin.

The research community continued experimenting with larger models by scaling model size and data size. Once the number of parameters used grew over 10 billion, models started exhibiting behavior and abilities different from the smaller models. Comparing the 330-million-parameter BERT and the 1.5-billion-parameter GPT-2 to the 175-billion-parameter GPT-3 and the 540-billion-parameter PaLM [10], larger models could perform in-context learning and showed surprising conversation abilities. The development of these models is what made the research community start addressing language models as large language models (LLMs).

The parameter scale of contemporary language models varies from a few hundred million parameters to a few hundred billion parameters. For the research presented in this paper, smaller models, in terms of the number of parameters, were of interest. Many of these models have multiple versions varying in the number of parameters. Flan-T5, hosting 11 billion parameters, was designed for instruction-tuning purposes [11]. CodeGen, also hosting 11 billion parameters, is an autoregressive language model designed for generating code [12]. The family of CodeGen models is trained sequentially on three datasets: THEPILE, BIGQUERY, and BIGPYTHON. MTF was applied to the pre-trained multilingual BLOOM and mT5 model families to produce finetuned variants called BLOOMZ and mT0 [13]. The authors found that fine-tuning large multilingual language models on English tasks with English prompts facilitates task generalization to non-English languages that appear only in the pre-training corpus. Further, PanGu-α [14] is a large-scale autoregressive language model with up to 200 billion parameters developed under the MindSpore [15] and trained on a cluster of 2048 Ascend 910 AI processors. The authors claim superior capabilities of PanGu-α in performing various tasks under few-shot or zero-shot settings, including text summarization, question answering, dialogue generation, etc. Being open source, the LLaMA model [16], hosting 65 billion parameters and trained using 2048 A100-80GB, has attracted significant attention from the research community. This model was fine-tuned to achieve abilities similar to ChatGPT or instruction-following abilities. Another recent model comes from The Technology Innovation Institute (TII) and is called Falcon-40B [17]. Falcon-40B is a 40-billion-parameter causal decoder-only model trained on 1000 B tokens of RefinedWeb enhanced with curated corpora. It is made available under the Apache 2.0 license, and its authors claim it outperforms previously described models.

Significant effort has been put into creating multilingual language models, and the authors of relevant studies mostly claim excellent results. Researchers have studied the ability of multilingual transformer-based language models to encode linguistic features of different languages. Research results reported in [18] indicate that even simple syntactic tasks vary in difficulty across languages, which imposes a hard limit on how well cross-lingual projection can perform, compared to the single-language modeling approach. These findings are supported by the results reported in [19], in which authors study cross-linguistic comparison of linguistic feature encoding in BERT models. Their results show that the structure of the model, in terms of the number of layers, should correspond to the complexity of the encoded features. As an example, authors have proven that a model developed for the Russian language needs more layers to achieve its optimal performance, while Korean and English models showed their best results at much earlier layers, due to the more complicated morphology and syntax of the Russian language.

3. SRBerta Base Model—Training and Evaluation

To achieve the goals defined during this research, whereas the final goal includes the processing of formal language in Serbian legislation, it was necessary to build a base model. The base model is a deep BERT-type neural network capable of understanding natural language. The natural language that needs to be learned is Serbian, and the network we
trained is based on a network model called RoBERTa. There are several processing steps that need to be carried out when training the network to learn the Serbian language, where the processing flow can be summarized as follows:

- Preparation of input data;
- Tokenizer training and text tokenization;
- Preparation of input tensors, and initialization of parameters and training;
- Network testing.

It should be emphasized that for the implementation of all solutions within this research, Python 3.9.5, PyTorch 1.11.0, and CUDA 11.3 were used. The implementation of tokenization algorithms and the RoBERTa network were used as ready-made architectures created by the HuggingFace community. Each of these implementations is a variant of version 4.17.0 of the library called Transformers.

In the first phase, the OSCAR dataset was used to train the SRBerta network. OSCAR is a large set of open data created using linguistic classification over data from the Common Crawl corpus. The dataset we used consisted of 645,747 texts (approximately 150 million words), where the total size of the stored texts, using UTF8 encoding, occupies slightly more than 2 GB. A sample of text used for SRBerta base model training is shown in Figure 1.

![Figure 1. A sample of text used for SRBerta base model training.](image_url)

The available dataset was preprocessed to make it suitable for base model training. The process of preprocessing the obtained data involved minimal changes to the text, and consisted of the following processing steps:

- The newline character “\n” was removed from each of the texts.
- Ten thousand texts were concatenated to form a single file, whereas sentences were separated using new line characters from one another.
- A total of 65 large files were created, as previously described, using the entire OSCAR unshuffled deduplicated sr corpus.
- Ten percent of previously prepared texts were randomly selected for the testing phase.

Once data preparation was complete, the next step was to train a tokenizer and perform text tokenization. Inspired by the RoBERTa network architecture and according to the principles of the same network, SRBerta used the ByteLevelBPETokenizer [20] tokenization algorithm and its implementation by the HuggingFace library [21]. The byte-level tokenizer relies on an existing implementation of the data compression Byte-Pair Encoding algorithm and uses the sub-word level tokenization principles. Such tokenization can be considered a balance between word-level and character-level tokenization and is designed to overcome the problems these methods encounter. The configuration of the tokenizer used for SRBerta is shown in Table 1.
Table 1. SRBerta Tokenizer configuration.

<table>
<thead>
<tr>
<th>Vocabulary Size</th>
<th>Minimum Frequency</th>
<th>Special Tokens</th>
</tr>
</thead>
<tbody>
<tr>
<td>30,522</td>
<td>2</td>
<td>&lt;s&gt; &lt;/s&gt; &lt;pad&gt; &lt;unk&gt; &lt;mask&gt;</td>
</tr>
</tbody>
</table>

As shown in Table 1, the size of the vocabulary was chosen to be 30,522 tokens and a few special tokens were defined:
- `<s>` and `</s>`—sentence delimiters;
- `<pad>`—used to add padding to sequences shorter than the fixed network input length;
- `<unk>`—serves to mark rare words, which were not covered during the vocabulary creation process or were not part of the input corpus;
- `<mask>`—masks randomly selected tokens in the network training process.

SRBerta tokenizer was created using 60% of preprocessed text, and its training lasted less than 15 min. The sample output obtained when we apply the tokenizer over the input sentence “Ово је српски Роберта токенизатор!” comprises the following tensors:

Input_ids_tensor: [0, 5115, 302, 2247, 8986, 19,992, 1076, 297, 393, 1721, 5, 2]

Attention_mask_tensor: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]

It should be noted that the output consists of two tensors, where the first tensor stores corresponding token ids from the dictionary and the second stores mask, which is later used to avoid performing attention on padding token indices.

The process of training the SRBerta network was logically divided into the following steps:
- Prepare input tensors using the previously trained tokenizer;
- Initialize network hyperparameters;
- Define the number of training epochs and perform the training loop.

As used within the reference RobertaForMaskedLM network model, SRBerta also uses three types of input tensors: token ID tensor, attention mask tensor, and label tensor. Token ID tensor is preprocessed and contains 15% of mask tokens for each input sequence. The attention mask tensor is used to deflect the attention mechanism away from padding tokens, while the label tensor is used for the calculation of loss function during the network training.

Before initiating the SRBerta network training process, various hyperparameters had to be defined to set the configuration of the SRBerta network architecture. Since SRBerta is based on the RobertaForMaskedLM model, we have followed recommended values and ranges when selecting hyperparameters, while all additional settings were made according to testing and evaluation results. In particular, the AdamW optimizer was used to update the network weights since it includes improved weight decay methods. The number of training epochs was determined empirically—a series of SRBerta models were trained with the goal of improving model accuracy at each subsequent training epoch. The decision regarding the optimal number of training epochs is based on the changes in the loss function value during the training. Once the loss function value starts oscillating in low increments, the training process should be stopped because the network model is considered to be in danger of becoming overfitted. Further, the size of the mini batch in the case of SRBerta was set to eight due to hardware limitations. The rest of the SRBerta network configuration parameters are shown in Table 2.

Table 2. SRBerta model configuration in comparison to RoBERTa base configuration.

<table>
<thead>
<tr>
<th>Model</th>
<th>Vocabulary Size</th>
<th>Hidden Layers</th>
<th>Attention Heads</th>
<th>Hidden Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SRBerta</td>
<td>30 K</td>
<td>6</td>
<td>12</td>
<td>768</td>
</tr>
<tr>
<td>RoBERTa base [2]</td>
<td>50 K</td>
<td>12</td>
<td>12</td>
<td>768</td>
</tr>
</tbody>
</table>
SRBerta used a slightly modified RoBERTa base configuration that consists of six hidden layers, with 12 attention heads, and reduced vocabulary size. Vocabulary size determines the size of the result vector. The dimensionality of the hidden state vector, which corresponds to each token from the input sequence, is set to 768. In short, the resulting SRBerta model can be summarized as follows: the network starts with an embedding layer comprising word-, position-, and token-type embeddings, followed by normalization and dropout layers connected to six encoder layers, whereas each of them contains self-attention mechanism, and outputs results to final language modeling head.

The process of training the SRBerta network in the first stage was carried out through 19 epochs and lasted a total of 6 days, 17 h, and 30 min on a GPU card Nvidia QUADRO RTX 4000. The evaluation of the SRBerta network was performed using 10% of the previously extracted input data from the OSCAR dataset. The testing dataset consisted of 60,000 input sequences, i.e., small texts in the Serbian language. A random masking of 15% of the tokens in each test sequence was performed.

During the training process, the value of the loss function was observed to determine the duration of each training epoch. The training epochs are zero-indexed. As shown in Figure 2, during the first training epoch (epoch index 0), the value of the loss function decreased significantly, from the starting value of 10 at the very beginning to an average value of about 0.26, which is the loss function value determined after the last step within the given epoch.

![Figure 2. The value of loss function during the first training epoch.](image)

In the next few epochs, the value of the loss function continued to decrease, with typical oscillations, although more slowly, taking values between 0.26, at the beginning of the second epoch (epoch index 1), and 0.17, at the end of the fifth epoch (epoch index 4). It should be noted that the model extremely effectively lowered the value of the loss function after only a couple of epochs of training, after which this value started to decrease slowly, albeit still significantly improving its results.

The value of the loss function during the training epochs 15 and 17 (epoch indexes 14 and 16) is shown in Figure 3 and indicates the trend of the loss function value during the last six training epochs of the SRBerta network. The range of loss function values continued to decrease, again more slowly, with oscillations from the range of [0.148, 0.139] during epochs with indexes 13 and 14, to the range of [0.144, 0.133] during epochs with indexes 15 and 16. The value of the loss function during the epoch with index 17 was in the range [0.14, 0.133], while the loss function value during the epoch with index 18 decreased to a value of 0.13 and then rose to a value of 0.142. Since the SRBerta network model failed to make significant progress in the last few training epochs and the value of the loss function started increasing during the last training epoch, the first stage of SRBerta network model training was concluded using nineteen training epochs. It should be emphasized that SRBerta models were stored after each training epoch, which resulted in 19 network models.
that SRBerta models were stored after each training epoch, which resulted in 19 network models.

Figure 3. The value of loss function during the training epoch 15 and 17 (epoch indexes 14 and 16).

The SRBerta network evaluation was performed using 10% of input data previously extracted from the OSCAR dataset. This part of input data consists of 60,000 input sequences, i.e., smaller texts in the Serbian language, and it was not used during the training epochs. The evaluation process was conducted for all network models generated after each training epoch, while the model quality is determined in the following way:

- 60,000 input sequences were used to evaluate each model.
- 15% of words within each input sequence were randomly chosen and masked.
- For each masked word, only the top five output scores of the network were considered.
- Only predictions of the network that absolutely matched the label behind the masked word were considered correct.

The results of the evaluation are shown in Table 3. The presented results confirm that the training process was successful and that the network managed to adjust its weights during the training on the language modeling task of the Serbian language. After only two training epochs, a remarkable result of 63.6% accuracy was obtained. The model quality measured for the models generated within the final training stages indicates that the SRBerta model converges around an accuracy value of 73% and slightly increases to a value of 73.7%.

Table 3. Determined quality of SRBerta models for each training epoch.

<table>
<thead>
<tr>
<th>SRBerta Model (Training Epoch Index)</th>
<th>Model Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>56.2%</td>
</tr>
<tr>
<td>1</td>
<td>63.6%</td>
</tr>
<tr>
<td>2</td>
<td>66.4%</td>
</tr>
<tr>
<td>4</td>
<td>69.3%</td>
</tr>
<tr>
<td>6</td>
<td>70.8%</td>
</tr>
<tr>
<td>8</td>
<td>71.9%</td>
</tr>
<tr>
<td>10</td>
<td>72.5%</td>
</tr>
<tr>
<td>12</td>
<td>73.1%</td>
</tr>
<tr>
<td>14</td>
<td>73.3%</td>
</tr>
<tr>
<td>16</td>
<td>73.5%</td>
</tr>
<tr>
<td>17</td>
<td>73.7%</td>
</tr>
<tr>
<td>18</td>
<td>73.7%</td>
</tr>
</tbody>
</table>

The results should be interpreted as follows: if we have a text in Serbian whose words are masked in 15% of cases, SRBerta will be able to predict the identical word (token) hidden behind it in 73.7% of cases. Further, in many other cases, the SRBerta model will be able to suggest other potential substitutes for the masked word. It should be noted that a softmax function was applied over the network outputs to generate probabilities for each token from the dictionary.

Another interesting finding emerged during the model quality assessment. A very significant characteristic of the obtained models in terms of the ability to understand
the structure of the Serbian language is its ability to correctly use cases in the Serbian language. Since the Serbian language has seven cases, the appropriate usage of cases can be challenging even for native speakers. Such an ability is demonstrated in the following example, where the sentence in Serbian reads as follows: Са годину сто гледали <MASK> у позоришту “Боско Буха”. When the above sentence is translated into English language, it means the following: Last night we saw <MASK> in the theater “Bosko Buha”.

When we feed the previous sentence into the SRBerta network, we obtain the next top five generated results which are here presented sorted by score, starting from the highest:

- 'представу',
- 'премијеру',
- 'концерт',
- 'Литургију',
- 'филм'

Each output, when obtained from the network, contains a token, score, and token-string that the network suggested should be used instead of the masked token. Previously listed string suggestions in Serbian language, when translated into English, mean play, premiere, concert, liturgy, and movie. One should note that the above nouns, suggested by the SRBerta network, represent the correct usage of accusative case in the Serbian language, which can sometimes be discerned by comparing case suffixes to nominative case, for example:

- nominative case—представа;
- accusative case—представу.

Another ability SRBerta demonstrates is that it can distinguish grammatical gender and number as well as verb tense in Serbian Cyrillic texts, which will be demonstrated using following two examples.

In the first example, the network is provided with a masked input sentence in Serbian language: Он <MASK> да једног дана постане новинар. When translated into English, the masked input means the following: He <MASK> to become a journalist one day. Again, the best-ranked scores are selected from the network’s output, namely жели and планира, meaning wants and plans, respectively. The key feature demonstrated by this example is the correct usage of the verb’s number and verb tense in the Serbian language. The difference can be seen, for example, by looking into the verb’s form in the first person singular—желим.

In the last example SRBerta is provided with a masked sentence: Она је некада <MASK> да постане глумица. Translated into English, it means: She once <MASK> to become an actress. This time, the outputs from the network, namely волела, желела, and могла—desired, wanted, could—proves, above all, the ability of the SRBerta network to distinguish between verb’s gender. The difference can be seen if the female gender of verbs’ forms, obtained from the network’s output, would be compared to male gender forms: волео, желео, могао.

4. SRBerta Fine-Tuning

In the second phase, SRBerta was fine-tuned using a larger number of available legal texts. These data were gathered from the Legal Information System of the Republic of Serbia. The texts used during the fine-tuning of the SRBerta network include legislation related to the Constitution of the Republic of Serbia and state organization, judiciary, defense, army and internal affairs, public revenues, monetary system, financial organizations, and business. These legislative texts, each of which is between 12 and 15 MB of data, had to be prepared, that is, preprocessed, in a slightly more specific way, to generate as many input sequences as possible. This process aims to reduce the number of data that could be lost due to the maximum size of the input sequence defined as 512 tokens. Therefore, all the texts were split into smaller units of 512 tokens, which were concatenated using the newline character. This resulted in generating new texts that will be optimally used by the
network during its training. At the end of the data preprocessing and after the creation of input sequences (tensors), a total of 10,266 masked input training sequences were created in a similar way as in the process of initial training of the network over the Serbian language.

When performing the process of model fine-tuning, the experiments were performed using different base models, pre-trained over different numbers of epochs, with the aim of determining which model would result in achieving the best accuracy. Experimentation during fine-tuning included the careful selection of AdamW optimizer parameters. On one hand, it was important to minimize model oscillations, so that the network weights are not harmed under the influence of new data. On the other hand, if the learning rate is set too low, this setup will not lead to the necessary changes in the network weights and their adaptation to the specifics of the legislative texts. Therefore, the experiments were performed using the learning rate values of $1 \times 10^{-4}$, $3 \times 10^{-5}$, $2 \times 10^{-5}$, and $1 \times 10^{-5}$.

The first group of experiments involved fine-tuning several SRBerta models picked from the different epochs of the previous training process. The experiment used a part of the input data with the learning rate value set to $1 \times 10^{-4}$. This learning rate value was considered the default value for the fine-tuning process since SRBerta had previously been trained using the same learning rate value. It should be noted that the fine-tuning process duration was only 30 min per model, within five epochs, with the best results achieved after the end of the third epoch for each of the selected models. The achieved accuracy values for selected models are shown in Table 4.

Table 4. Achieved accuracy values for selected SRBerta models.

<table>
<thead>
<tr>
<th>Epoch</th>
<th>Base Model Accuracy</th>
<th>FT-Model Accuracy, Learning Rate $1 \times 10^{-4}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>71.9%</td>
<td>80.7%</td>
</tr>
<tr>
<td>10</td>
<td>72.5%</td>
<td>81.2%</td>
</tr>
<tr>
<td>12</td>
<td>73.1%</td>
<td>81.5%</td>
</tr>
<tr>
<td>14</td>
<td>73.3%</td>
<td>81.6%</td>
</tr>
<tr>
<td>16</td>
<td>73.5%</td>
<td>81.7%</td>
</tr>
<tr>
<td>17</td>
<td>73.7%</td>
<td>81.9%</td>
</tr>
<tr>
<td>18</td>
<td>73.7%</td>
<td>81.9%</td>
</tr>
</tbody>
</table>

For further fine-tuning process, we have decided to continue experimenting with the model generated during epoch 18, since it demonstrated the lowest value of the loss function, e.g., the highest accuracy. In the next fine-tuning stage, we set the learning rate value to $3 \times 10^{-5}$, included the whole training set, and performed five training epochs. The resulting loss function value is shown in Figure 4, and the model reached 84.7% accuracy.

![Figure 4. The value of loss function for model from epoch 18 with learning rate value $3 \times 10^{-5}$.](image.png)
Next, we decreased the learning rate value to $3 \times 10^{-5}$ and performed an additional five training epochs resulting in the values of loss function shown in Figure 5 and a maximal accuracy level of 84.8%. In the last step, we decreased the learning rate value to $1 \times 10^{-5}$ and performed an additional five training epochs resulting in an accuracy level of 84.5%, which concluded the training process of the model using previously described hardware resources.

Briefly summed, the SRBerta network was created as a result of a training process that lasted a total of 6 days and 19 h, using a large corpus of texts collected from the Internet, in the Serbian language. It was fine-tuned using another smaller, but many times better corpus of Serbian laws, and achieved an accuracy of as much as 84.8% on the task of masked language modeling of legislative texts. The achieved results proved the feasibility of creating this kind of model based on the previously defined principles of natural language processing.

5. Conclusions

Motivated by previous works and conducted experiments, followed by the fact that the various architectures of BERT transformer networks became open source, we were inspired to define an approach for training a network of this type to understand the Serbian language and the context of its sentences. Based on all the above, the development and testing conclude that it is possible to achieve a high level of accuracy (industrially acceptable of over 90% accuracy), where the only prerequisite is having a sufficiently high-quality and large set of data and an appropriate physical architecture of the system on which we perform the training process. Training the network to understand the Serbian language would enable a wide range of problems to be solved by creating fine-tuned models based on natural language processing.

Our motivation became even greater once promising results were achieved by our model after the first training phase, e.g., after the process of training on Serbian language texts. These results provided us with the foundation to proceed and show a potential solution to the problem of analyzing and correcting texts legislation—SRBerta model. SRBerta united all our goals and provided real evidence for the possibility of the development and usage of such technological solutions in the Republic of Serbia. SRBerta network was trained in two stages. The process of training the SRBerta network in the first stage was carried out through 19 epochs and lasted a total of 6 days, 17 h, and 30 min on a GPU card Nvidia QUADRO RTX 4000. Testing results conducted after the first training stage have shown that SRBerta will be able to predict the identical token in 73.7% of cases. In the second stage, it was fine-tuned using another smaller, but many-times-better corpus of...
Serbian laws, and achieved accuracy of as much as 84.8% on the task of masked language modeling. The fact that the SRBerta network has achieved an accuracy of almost 85% when testing masked legislation texts, undeniably confirms the initially set assumption that it is possible to facilitate and automate the traditional way of doing business of this type in the Republic of Serbia, considering not only the specificities and rules of the Serbian language, but also its formal texts, while written in Cyrillic script.

Future improvements of the presented model will include the use of a larger and better physical system, on which it is possible to implement a higher quality training process, and which would allow the use of large mini-series of data (in the case of the RoBERTa network, the used series size was as many as 8000 training samples, while in our case, the highest number of series that the given GPU processor could support within the pre-training process was a maximum of 8).

**Author Contributions:** Conceptualization, M.B.; Methodology, M.B.; Software, M.B. and J.K.; Data curation, J.K.; Writing—original draft, M.B.; Writing—review and editing, J.K. and L.S.; Supervision, L.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was funded by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The model presented in this study is openly available at https://huggingface.co/JelenaTosic/SRBerta (accessed on 20 January 2024).

**Acknowledgments:** Authors would like to thank the Ministry of Science, Technological Development and Innovation of the Republic of Serbia for funding this research.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**


Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.