Transformer-based Model for Text Classification in Ukrainian

Larysa Katerynych, Maksym Veres and Eduard Safarov

Taras Shevchenko National University of Kyiv, Academician Glushkov Avenue 4d, Kyiv, 03680, Ukraine

Abstract

The purpose of this paper is to find a solution for printed text classification in Ukrainian, as well as to choose the means for its implementation. The paper considers the problem of identification of short texts by their scientific topic. A model, built for classification, is described. The current state of NLP and transfer learning is studied too. In practice, the effectiveness of the implemented methods is proven, which allows to obtain good results of text classification. These methods and approaches include concepts such as transfer learning, NLP, BERT. The model is built using Python programming language and some of its machine learning libraries. A multilanguage BERT model by Google is additionally trained on Ukrainian texts from school subjects. After that, the questions from the external independent evaluation are submitted to the input, and the model classifies them.

Keywords [1]

Text classification, deep learning, recurrent neural networks, long short-term memory, convolutional neural networks, natural language processing, bidirectional encoder representations from transformers, local interpretable model-agnostic explanations.

1. Introduction

Searching for scientific papers and articles related to a particular field of knowledge can be difficult for students, scientists, researchers, etc. Thematic classification of works facilitates the search process. Having a list of subjects in the research field, it is needed to find out which subject area a particular study is more related to. However, manually categorizing a large collection of resources is a time-consuming process. Usually, the strategy is to search based on keywords, certain terms in the title of the article and even in the whole text. However, processing the entire text of the article takes a long time. In this case, neural network (NN) text classification can be applied. With deep learning evolving, NN architectures such as recurrent NN (RNN), long short-term memory (LSTM) and convolutional NN (CNN) demonstrated good performance in natural language processing (NLP) tasks such as text classification, machine translation and more. However, the effectiveness of deep learning models in NLP depends on large sets of labeled text data. Most labeled datasets are not sufficient for deep NN training, because these networks have many parameters, and training such networks on small datasets leads to overtraining. Another reason that held NLP progress back was the lack of transfer learning. It was not used in NLP until 2018, when Google introduced a transformer model. Since then, transfer learning in NLP has helped to solve complex text processing problems.

2. Transfer learning in NLP

Classic machine learning technique is depicted in the Figure 1. As can be seen, each separate task demands training of a separate NN with its own model and data. If a new problem arises for NN to solve, it could be difficult to build an effective system for this purpose. Transfer learning is depicted in the Figure 2. It is a technique, where a deep learning model, taught on a large dataset, is used to perform similar tasks on another dataset [1]. This model of deep learning is known as a pre-trained model [2]. Most tasks in NLP, such as text classification, machine translation, etc., are sequence modeling tasks. Classic machine learning models and NN cannot fixate the consecutive information, present in the text. Therefore, RNN have been used, as these architectures can model the sequential information.

Information Technology and Implementation (IT&I-2021), December 01–03, 2021, Kyiv, Ukraine
EMAIL: katerynych@gmail.com (L. Katerynych); veres@ukr.net (M. Veres); edward.saf99@gmail.com (E. Safarov)
ORCID: 0000-0001-7837-764X (L. Katerynych); 0000-0002-8512-5560 (M. Veres); 0000-0001-9651-4679 (E. Safarov)

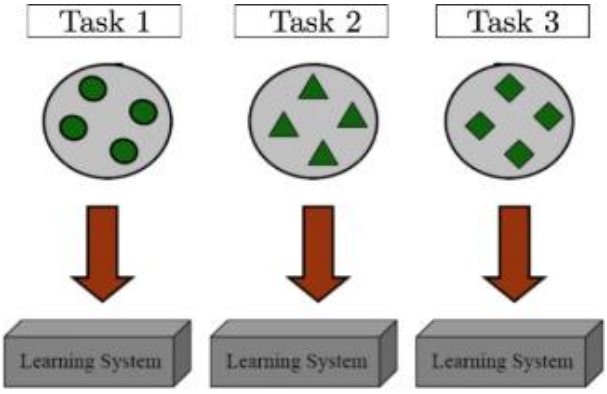



Figure 1: Classic machine learning

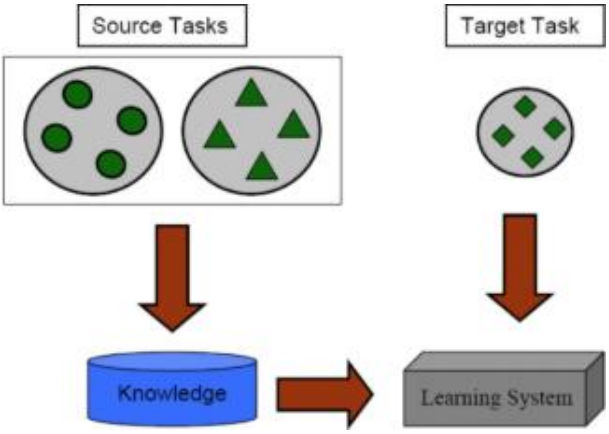


Figure 2: Transfer learning

However, these periodic NN have their drawbacks. One of the main problems is that RNN cannot be parallelized (as opposed to a linear NN [3]) because it accepts one input at a time. In the case of a text sequence, RNN or LSTM accept one token at a time as input. Therefore, training such model on a large dataset will take a long time. As mentioned above, in 2018, the transformer was introduced by Google, which gave a significant impulse to NLP systems [4]. Soon, a wide range of models, based on transformers, were offered for various NLP tasks. There are many advantages to using transformer-based models, but the most important are the following: [2]

1. These models do not process input sequence token-by-token. They take the entire sequence at once, which is a significant improvement over RNN-based models, as the model can now be accelerated by graphics processing unit (GPU).
2. Labeled data is not required for the preparation of these models. It is needed to provide a huge amount of unlabeled text data to prepare a model based on the transformer. This trained model can be used for other NLP tasks. They can include text classification; named-entity recognition (NER), for example, people, geographical, company names; text generation; etc.

Bidirectional encoder representations from transformers (BERT) and the second version of generative pre-trained transformer (GPT-2) are the most popular NLP models based on transformers. For example, it is possible to use the previously trained BERT model to classify Ukrainian text.

3. Model fine-tuning

BERT is a large NN architecture with big number of parameters that can range from 100 to 300 mln. Therefore, training the BERT model from scratch on a small dataset will lead to overtraining.

It is better to use a pre-trained BERT model as a starting point. These pre-trained models are usually trained on big datasets. There are several options for BERT. All of them are listed on the official page [4]. A universal multilingual BERT for 104 languages was chosen. It uses a dictionary of whole words as well as the most common syllables. A part of the dictionary for multilingual version of BERT can be seen in the Figure 3. This dictionary shows which words NN uses as input. These are whole words, for example, кілька. But most Ukrainian words are broken down into syllables. So, the word прийшов will be split into прий and ##шов.

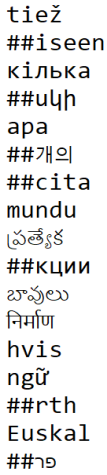


Figure 3: Some words from BERT dictionary

The training of the model can be continued on another, relatively smaller new dataset. This process is known as fine-tuning of the model [5]. Fine-tuning strategies depend on various factors, but the most important are the size of the new dataset and its similarity to the original dataset. Given that the nature of a typical NN for NLP is more universal in the early layers and becomes more closely related to a specific dataset on subsequent layers, four main scenarios can be identified:

1. The new dataset is smaller and similar in content to the original dataset. If the amount of data is small, then it makes no sense to fine-tune the NN due to overfitting. Since the data is similar to the original, it can be assumed that the distinguishing features in NN will be relevant for this dataset as well. Therefore, the optimal solution is to train the linear classifier as a distinctive feature of NN.
2. The new dataset is relatively large and similar in content to the original dataset. Since there is more data, the overfitting does not take place, if the entire NN is being fine-tuned.
3. The new dataset is smaller and significantly different in content from the original dataset. Since the amount of data is small, only a linear classifier will be sufficient. Since the data is significantly different, it is better to train the classifier not from the top of NN, which contains more specific data. Instead, it is better to train the classifier by activating it on earlier layers of NN.
4. The new dataset is relatively large and differs significantly in content from the original dataset. Since the dataset is very large, it is possible to train the entire NN from scratch. Nevertheless, in practice, it is often still more advantageous to use it to initialize weights from a pre-trained model. In this case, there is enough data to fine-tune the entire NN.

4. BERT architecture

First, BERT is based on the transformer architecture as stated above.

Second, BERT is pre-trained on a large set of unlabeled text, including the entire Wikipedia (that is 2.5 bln words) and the BooksCorpus (800 mln words). This process took 4 days for 16 tensor processing units (TPU). The pre-training step is half the success of BERT. The reason is when a model trains on a large text body, it begins to gain a deeper understanding of how the given language "works".

Third, BERT is a deep bidirectional model. Bidirectionality means that BERT learns information from both left and right side of the token context at the training stage. A similar method of bidirectional processing is also used by embeddings from language model (ELMo) system developed by Paul Allen Institute of Artificial Intelligence. However, BERT demonstrates a more complex relationship between the layers of language representation, so it is considered deeply bidirectional and ELMo is superficially bidirectional. For comparison, the visualization of NN architectures of different types is displayed in the Figure 4. They include an example of one-way processing method – OpenAI GPT.

5. ktrain library

ktrain library [6] for Python programming language can be used for BERT fine-tuning. This is a wrapper for Keras framework that helps to build, train, and deploy NN models with minimum amount of code. ktrain provides means for: [7]

- learning speed regulation, which will help to find the initial level of learning for the model;

- visual graphs of learning speed to increase productivity;
- pre-trained models for text data (e.g., text classification, NER), images (e.g., image classification), graphs (e.g., link prediction);
- methods that allow downloading and pre-processing text and images in various formats;
- verification of data that have been misclassified to improve the model;
- an application programming interface (API) for saving and deploying models and pre-processing data.

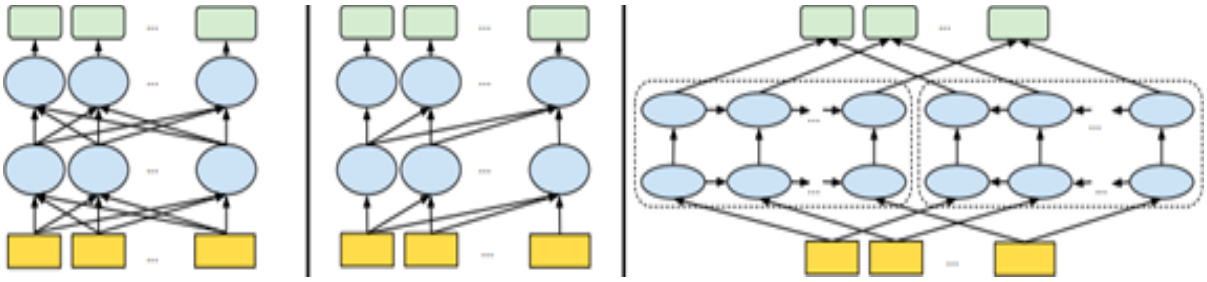


Figure 4: BERT, OpenAI GPT and ELMo respectively

6. Problem statement

As an example of a practical solution to the problem of Ukrainian text classification, BERT model can be considered and taught to classify scientific topics of given texts. These texts will cover the following 7 school subjects:

- history of Ukraine,
- physics,
- geography,
- biology,
- mathematics (algebra and geometry),
- Ukrainian (language and literature),
- chemistry.

The task is to create a system that determines automatically to which subject the question relates. The dataset used for this case consists of electronic textbooks on the specified subjects. To test the model, the questions from the tests, that were offered to school graduates at the external independent evaluation (also known as "3HO") of 2021, were considered. A sample can be seen in the Figure 5. It should also be noted that only 11th (final) grade textbooks are included in the dataset, while the external evaluation questions cover several years of study and relate to a wider range of knowledge. Google Colab was used for development.

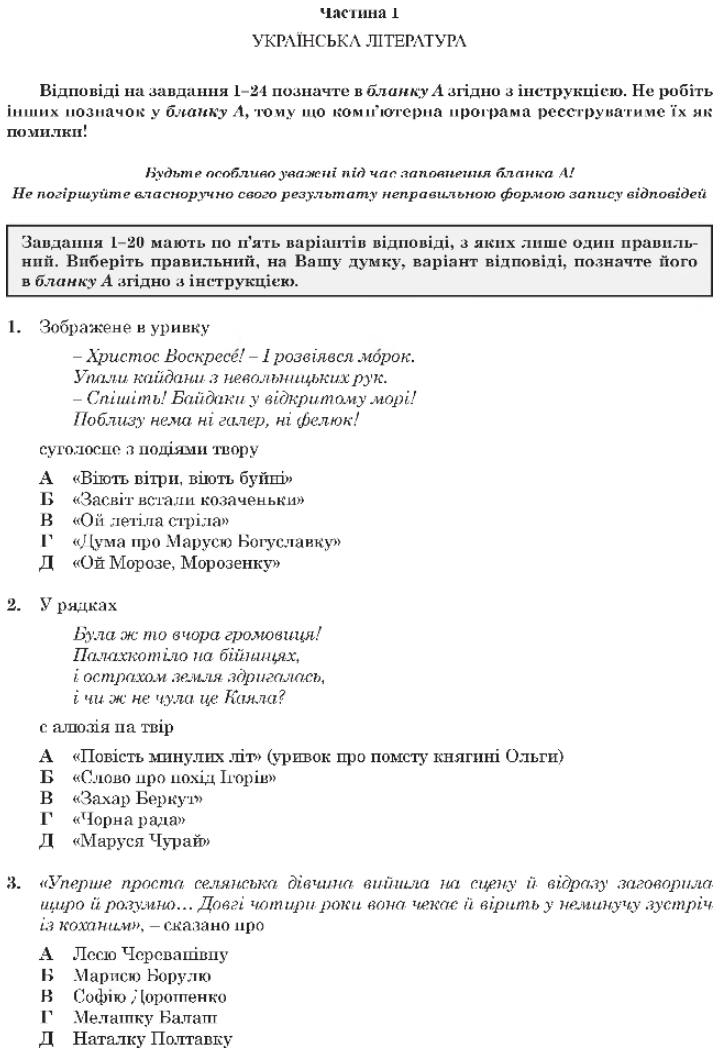
6.1. Download and process input data

ktrain library is needed to get started:

```
!pip install ktrain
import ktrain
from ktrain import text
```

Publicly available [9] textbooks for remote studying are considered as input data. The textbooks were converted to TXT format and divided into smaller files. ktrain automatically detects natural language and character encoding, processes the data, and sets up the model:

```
(x_train, y_train), (x_test, y_test), preproc = text.texts_from_folder(
  '/content/drive/MyDrive/dataset/',
  maxlen=75,
  max_features=10000,
  preprocess_mode='bert',
  train_test_names=['train', 'test'],
  val_pct=0.1,
  classes=['history', 'physics', 'math', 'geography', 'biology', 'Ukrainian',
    'chemistry'])
```



Частина 1

УКРАЇНСЬКА ЛІТЕРАТУРА

Відповіді на завдання 1–24 позначте в *бланку А* згідно з інструкцією. Не робіть інших позначок у *бланку А*, тому що комп'ютерна програма реєструватиме їх як помилки!

Будьте особливо уважні під час заповнення бланка А!
Не погіршуйте власноручно свого результату неправильною формою запису відповідей

Завдання 1–20 мають по п'ять варіантів відповіді, з яких лише один правильний. Виберіть правильний, на Вашу думку, варіант відповіді, позначте його в *бланку А* згідно з інструкцією.

1. Зображене в уривку

– Христос Воскресé! – І розвіявся мрок.
Упали кайдани з невольницьких рук.
– Спішіть! Байдаки у відкритому морі!
Поблизу нема ні галер, ні фелюк!

суголосне з подіями твору

А «Віють вітри, віють буйні»
Б «Засвіт встали козаченьки»
В «Ой летіла стріла»
Г «Дума про Марусю Богуславку»
Д «Ой Морозе, Морозенку»

2. У рядках

Була ж то вчора громовиця!
Палахкотіло на бійницях,
і острахом земля здригалась,
і чи ж не чула це Кайла?

є алюзія на твір

А «Повість минулих літ» (уривок про помсту княгині Ольги)
Б «Слово про похід Ігорів»
В «Захар Беркут»
Г «Чорна рада»
Д «Маруся Чурай»

3. *«Уперше проста селянська дівчина вийшла на сцену й відразу заговорила щиро й розумно… Довгі чотири роки вона чекає й вірить у неминучу зустріч із коханим»*, – сказано про

А Лесю Українку
Б Марусю Борулю
В Софію Яворівну
Г Мелашку Балаш
Д Наталку Полтавку

Figure 5: A page from the external independent evaluation test on Ukrainian language and literature

The first argument is the path to the dataset folder. The `maxlen` argument specifies the maximum number of words (512 for BERT, but it is better to use less to reduce memory usage and increase speed) in each file, with extra words being cut off. `maxlen=75` because the input text files are small.

The text must be pre-processed for usage with BERT. This is achieved by setting the `preprocess_mode` value to `'bert'`. The BERT model and vocabulary will be loaded automatically, if necessary. `val_pct=0.1` means automatically selecting 10% of the data for validation.

Finally, the `texts_from_folder` function expects the following directory structure:

```
<folder>
  train
    <subject_1>
    <subject_2>
    <subject_3>
    ...
  test
    <subject_1>
    <subject_2>
    <subject_3>
    ...
```

So, the dataset folder with corresponding content for 7 classes `'history'`, `'physics'`, `'math'`, `'geography'`, `'biology'`, `'Ukrainian'`, `'chemistry'` is created. The result may look like this:

```
detected encoding: UTF-8-SIG
downloading pretrained BERT model (multi_cased_L-12_H-768_A-12.zip)...
extracting pretrained BERT model...
```

```
done.

cleanup downloaded zip...
done.

preprocessing train...
language: uk
done.

Is Multi-Label? False
preprocessing test...
language: uk
done.
```

6.2. Use BERT learner object for content in Ukrainian

Creating a model and wrapping it with the learner:

```
model = text.text_classifier('bert', (x_train, y_train), preproc=preproc)
learner = ktrain.get_learner(model,
                             train_data=(x_train, y_train),
                             val_data=(x_test, y_test),
                             batch_size=32)
```

The first argument of the `get_learner` function uses a pre-trained BERT model with a randomly initialized end dense layer. The second and third arguments are training and verification data, respectively. The last argument of `get_learner` is the packet size. A small `batch_size=32` is used based on Google's recommendations.

6.3. Model training

To train the model, the optimal level of training is found, that corresponds to the problem being solved. ktrain offers an effective method called `lr_find`, which trains a model with different learning metrics and creates a graph of model loss as the learning level increases.

Loss graph is displayed in the Figure 6.

```
learner.lr_find()
learner.lr_plot()
```

The graph of the loss function shows that the classifier provides minimum losses when the training level is 10^{-5}...10^{-4}. Model training:

```
learner.fit_onecycle(2e-5, 1)
```

`fit_onecycle` function from ktrain library is used. This function utilizes the `onecycle` learning speed policy, which linearly increases the learning speed during the first half of learning and then reduces the learning speed for the second half [8].

Result:

```
begin training using onecycle policy with max lr of 2e-05...
542/542 [==============================] - 710s 1s/step - loss: 0.5202 -
accuracy: 0.8294 - val_loss: 0.2498 - val_accuracy: 0.9182
```

Validation:

```
learner.validate(val_data=(x_test, y_test))
```

Result:

```
              precision    recall  f1-score   support

           0       0.91      0.89      0.90       370
           1       0.87      0.93      0.90       313
           2       0.90      0.87      0.89       348
           3       0.85      0.77      0.81       201
           4       0.93      0.95      0.94       792
```

```
              5       0.98      0.97      0.97       541
              6       0.89      0.91      0.90       222

       accuracy                           0.92      2787
      macro avg       0.90      0.90      0.90      2787
   weighted avg       0.92      0.92      0.92      2787
```

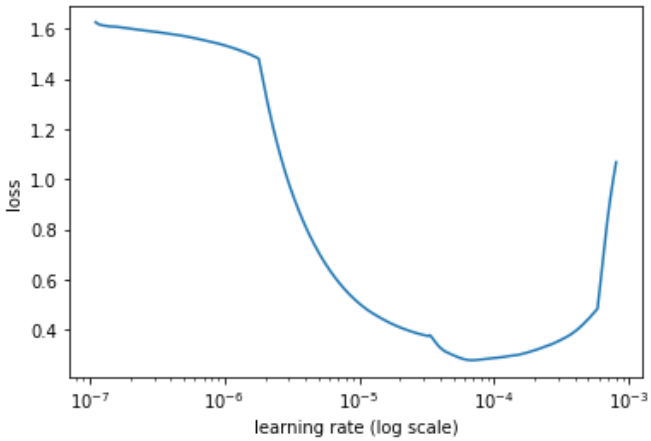


Figure 6: Loss graph

As can be seen from the results of validation, training reaches 85-98% accuracy (precision) in one epoch. Classification errors:

```
learner.view_top_losses(n=10, preproc=preproc)
```

Creating a predictor:

```
p = ktrain.get_predictor(learner.model, preproc)
```

It is better to save the model for later use:

```
p.save('./drive/MyDrive/predictor')
```

After downloading it and trying to offer models of questions from the external independent evaluation of 2021:

```
fin_bert_model = ktrain.load_predictor('./drive/MyDrive/predictor')
p.predict("Вільгельм фон Габсбург-Лотрінген – австрійський архікнязь, полковник армії УНР, поет. Під яким псевдонімом він відомий як полковник Українських січових стрільців (УСС)?")
history

p.predict("Усю воду із широкої посудини перелили у високу вузьку порожню посудину. Якими стануть сила тиску й тиск води на дно вузької посудини після цього порівняно із силою тиску и тиском цієї води на дно широкої посудини? Уважайте, що посудини мають циліндричну форму")
physics

p.predict("Установіть відповідність між графіком (1 – 3) функції, визначеної на проміжку [– 4; 4], та її властивістю (А – Д)")
math

p.predict("Укажіть материк, на якому лежить Україна")
geography

p.predict("Під час експерименту декілька яєць морських їжаків помістили в морську воду, де їх запліднили. У цю воду добавили мічений Тритієм (3Н) тимідиловий нуклеотид (рис. 1), який поглинали клітини ембріонів")
biology

p.predict("Зображене в уривку - Христос Воскресе! – І розвіявся морок. Упали кайдани з невольницьких рук. - Спішіть! Байдаки у відкритому морі! Поблизу нема ні галер, ні фелюк! суголосне з подіями твору")
Ukrainian
```

```
p.predict("Алмаз і графіт – прості речовини")
chemistry

fin_bert_model.predict(Однакові кульки, підвішені на нитках, заряджені так, як це показано на рисунках. У якому з випадків правильно зображено положення цих кульок, зумовлене їхньою взаємодією?")
physics

fin_bert_model.predict("Яка владна інституція звернулася із цитованою відозвою до населення України?")
history

fin_bert_model.predict("Установіть відповідність між виразом (1 – 3) і твердженням про його значення (А – Д), яке є правильним, якщо a = -2")
math

fin_bert_model.predict("З будь-якої точки Світового океану можна дістатися в будь-яку іншу, не перетнувши суходіл. Це доводить, що Світовий океан –")
geography

fin_bert_model.predict("Уключення міченої сполуки в молекули клітин ембріона відбувається під час")
biology

fin_bert_model.predict("Однаковий звук позначають букви, підкреслені в окремих словах речення")
Ukrainian

fin_bert_model.predict("Укажіть формулу вуглекислого газу")
chemistry
```

Taking a closer look at how the model makes conclusions for certain inputs:

```
fin_bert_model.explain("Мічена сполука в клітинах ембріонів потрапляє в молекули")
```

Result:

```
y=biology (probability 0.989, score 5.285) top features
Contribution    Feature
+5.804          Highlighted in text (sum)
-0.519          <BIAS>
мічена сполука в клітинах ембріонів потрапляє в молекули
```

This visualization is generated using a technique called local interpretable model-agnostic explanations (LIME) [10]. It helps to understand the relative importance of different words for the final prediction using a linear interpreted model. “Green” words contribute to the correct classification (biology), “red” words reduce the probability of correct prediction. Shade of color indicates the strength or size of the coefficients in the final linear model.

7. Other approaches

There are a number of solutions representing classification of English texts. Many of them provide algorithms to build text classifiers. But it’s hard to find works describing algorithms to construct a classifier of Ukrainian text. The main problem, however, is not an algorithm itself, but the lack of resources for experiments to train the classifier. A researcher can:

- use Brownian Corps of the Ukrainian Language (BCUL),
- use specific dictionaries,
- create own dataset.

Deep learning tends to use large and robust datasets in order to perform well. One of the attempts to perform Ukrainian text classification is described in the paper [11], where its authors consider random forest classifier, support vector machines (SVM), naive Bayes classifier and logistic regression algorithms as well as BCUL dataset. The best result is shown by SVM model. Its average accuracy is 80%. Another approach to the problem proposes a solution, which can detect sentiments from Ukrainian text (namely hotel reviews) and classify them for positive and negative ones [12]. Besides, it analyzes

reviews about given hotel and summarizes its most important positive and negative properties. The dataset, which contains user reviews in Ukrainian was parsed from Booking.com and TripAdvisor. Following techniques were considered: fastText (created by Facebook Artificial Intelligence Research lab), Seq2Seq, CNN, RNN and recurrent convolution neural network (RCNN). RCNN model gives the best accuracy on available dataset: 85% for text classification and 86% for sentence classification.

8. Conclusions

BERT and ktrain were used to solve the problem of classifying documents in Ukrainian. A complex model like BERT can be applied to any problem in any language (out of 104 most popular ones in the world, including Ukrainian, for which there is a pre-trained BERT) using ktrain.

Testing accuracy of 85-98% was achieved in one learning epoch. Although the effectiveness of BERT was proven, it is relatively slow both in terms of learning and predictions for new data. Therefore, if the training lasts more than one epoch, it may be better to omit the `val_data` argument from `get_learner` and check the accuracy only after training.

This can be done in ktrain using the `learner.validate` method, as shown in code samples above. BERT can be quite demanding on memory. If errors, indicating that the GPU memory limits have been exceeded, are encountered, it is possible to reduce the value of either `maxlen`, or `batch_size` parameters. If the model performs well after training, it should be saved for future classifications. When using BERT, Keras's built-in `load_model` function does not work, although `model.save_weights` and `model.load_weights` can still be used to save weights and load them. But to load the model, the `learner.load_model` function from ktrain should be used.

Finally, the proposed BERT solution for Ukrainian text classification was compared to other approaches to this problem. As can be seen, the described model performs well, and its accuracy is high enough.

9. References

[1] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, The MIT Press, 2016.

[2] E. Olivas, J. Guerrero, M. Sober, J. Benedito, A. Lopez, Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques, IGI Publishing, 2009.

[3] L. Katerynych, M. Veres, E. Safarov, Neural Networks' Learning Process Acceleration, in: Proceedings of the 12th International Scientific and Practical Conference of Programming, UkrPROG'2020, Problems in Programming Scientific Journal, Kyiv, 2020, pp. 313–321. doi: 10.15407/pp2020.02-03.313.

[4] J. Devlin, S. Petrov. Multilingual BERT models, 2019. URL: https://github.com/google-research/bert/blob/master/multilingual.md.

[5] J. Devlin, M. Chang, Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing, 2018. URL: https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html.

[6] A. Maiya. ktrain: A Low-Code Library for Augmented Machine Learning, 2020. URL: https://arxiv.org/pdf/2004.10703.pdf.

[7] S. Ravichandiran, Getting Started with Google BERT: Build and Train State-of-the-Art Natural Language Processing Models using BERT, Packt Publishing Ltd, 2021.

[8] L. Smith: A Disciplined Approach to Neural Network Hyper-parameters: Part 1, 2018. URL: https://arxiv.org/pdf/1803.09820.pdf.

[9] Electronic Versions of Textbooks, Institute for Modernization of Educational Content, 2021. URL: https://lib.imzo.gov.ua/yelektronn-vers-pdruchnikv.

[10] M. Ribeiro, S. Singh, C. Guestrin, "Why Should I Trust You?": Explaining the Predictions of any Classifier, in: 22nd International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016, pp. 1135–1144.

[11] K. Bobrovnyk, K. Dukhnovska, M. Piroh, Thematic Classification of Ukrainian Texts, Difficulties of its Introductions, in: Control Systems and Computers, Kyiv, Ukraine. doi:10.15407/usim.2019.01.041.

[12] D. Babenko, Determining sentiment and important properties of Ukrainian-language user reviews, Master's thesis, Ukrainian Catholic University, Lviv, Ukraine, 2020.