National Research University Higher School of Economics
Faculty of Computer Science
HSE and University of London Double Degree Programme in Data Science and Business Analytics

GROUP BACHELOR'S THESIS

Software Project

Development of a system for analyzing large text data

Prepared by the students: of Group 212, Year 4 (year of study),

Levin Maxim Alekseevich, Group 212, Year 4,

Malikov Arsenii Ivanovich, Group 212, Year 4,

Sagidulaeva Amina Zeybulaevna, Group 212, Year 4,

Talagaeva Polina Vladislavovna, Group 212, Year 4

Thesis Supervisor:

Senior Lecturer, Department of Data Analysis and Artificial Intelligence,
Parinov Andrei Andreevich

Moscow
2025

Table of content

1. Introduction

1.1. Abstract

This paper examines modern techniques and technologies for processing large volumes of text data, using the example of analyzing news content from the RBC website as a case study. The aim of the study is to develop a system for classifying news articles, with the subsequent visualization of the resulting data.

During the course of the research, I have explored and applied machine learning algorithms in order to construct and train a BERT model for the purpose of categorizing news articles based on their content.

1.2. Key Words

BERT - Bidirectional Encoder Representations from Transformers — a transformer-based model developed by Google for a wide range of natural language understanding tasks.

DeepPavlov ruBERT - A Russian-language adaptation of the BERT model trained on large corpora of Russian text by the DeepPavlov team.

Machine Learning - A subfield of artificial intelligence that enables systems to learn patterns from data and make predictions or decisions without explicit programming.

Text Classification - The process of assigning predefined categories to textual data based on its content using statistical or machine learning methods.

Big Text Data - Large-scale collections of unstructured textual data, often used in NLP tasks for training models or extracting insights.

Transformers - A deep learning architecture based on self-attention mechanisms, commonly used in NLP models like BERT and GPT.

RBC News - News content from the Russian media source RBC (РБК), used as a real-world dataset for training and evaluation.

Data parsing - The process of extracting and structuring relevant information from raw data sources.

NLP - A field at the intersection of linguistics and machine learning, focused on the interaction between computers and human language.

Tokenization - The process of breaking down text into smaller units (tokens), such as words or subwords, for model input.

Model Training - The process of optimizing a machine learning model's parameters using training data to minimize prediction error.

Class Imbalance - A situation where some classes have significantly more examples than others, potentially biasing the model.

Accuracy - A performance metric indicating the proportion of correct predictions made by a model.

Macro F1-score - A performance metric that calculates the F1-score independently for each class and then takes the average, treating all classes equally.

Topic Modeling - A type of statistical model used to discover abstract topics within a collection of documents.

BERTTopic - A topic modeling technique that uses BERT embeddings and clustering to identify semantically meaningful topics.

Visualization - The graphical representation of data or model results to facilitate interpretation and analysis.

Train/Test split - A common practice of dividing data into training and testing subsets to evaluate model performance.

Data augmentation - Techniques used to increase the diversity of training data without collecting new data, such as synonym replacement or paraphrasing.

Synonymization - The process of replacing words in text with their synonyms to generate variant data samples.

WordNet - A lexical database of the English language that groups words into sets of synonyms and records their relationships.

HuggingFace Transformers - A widely used open-source library that provides pretrained transformer models and tools for NLP tasks.

CrossEntropyLoss - A loss function commonly used in classification tasks that measures the difference between predicted and actual distributions.

AdamW - An optimizer that combines the Adam algorithm with weight decay regularization, often used in training transformer models.

BertForSequenceClassification - A pretrained BERT model extended with a classification head, used for sequence-level classification tasks.

1.3. Relevance

The rapid growth of modern information flow makes the task of efficient processing and analysis of large volumes of textual data increasingly important. News content has a significant impact on shaping opinions and decisions in economic, business, and governmental spheres. However, the manual processing of vast amounts of information requires a considerable amount of time and human resources. Therefore, automated text data analysis systems have become essential for fast and accurate decision-making processes. The implementation of such systems has the potential to significantly accelerate and improve analytical procedures, ensuring high accuracy and relevancy of conclusions, while minimizing the influence of human error. This can lead to more efficient and informed decision-making, ultimately benefiting economic, business, and administrative processes.

1.4. Team Structure

The team consisted of students from the DSBA212 program: Arsenii Malikov, Polina Talagaeva, Maxim Levin, and Amina Sagidulaeva. The project tasks were distributed as follows:

- Polina Talagaeva was responsible for extracting data from the RBC website and Telegram news channels, which would be used to train and evaluate machine learning models. She also created a CatBoost machine learning model for classification purposes, in order to compare its performance with that of other models. Additionally, she used LDA (Latent Dirichlet Allocation) for topic modeling purposes.

- Sagidulaeva Amina was involved in the following activities: studying the Citus distributed database management system based on PostgreSQL; deploying storage infrastructure; participating in the development of system architecture; designing and creating databases; preparing database structures for receiving paired data from external sources (RBC and Telegram); and preparing an analytical review on "Modern Approaches to Internal Representation and Storage of Texts in NLP".
- Malikov Arsenii was responsible for developing, training, and evaluating the performance of a BERT machine learning model using accuracy and F1 metrics, as well as preparing data for model training, analyzing initial dataset data, and identifying possible solutions to issues related to initial data (such as an imbalance in class distribution within the initial dataset).
- Maxim Levin was responsible for working with the database and further visualizing the results of machine learning models using the Streamlit framework.

1.5. Mathematical Problem Statement

The general mathematical problem:

$$\mathcal{I} \xrightarrow{\text{сбор данных}} \mathcal{D} \xrightarrow{P} \mathcal{S} \xrightarrow{DB} \mathcal{S}_{DB} \xrightarrow{f} Y \xrightarrow{V} \mathcal{G}$$

I - a set of source information resources.

D is a set of source documents (raw data) from all sources.

The P - parser.

S is a set of structured data.

DB is a database.

SDB is a subset of S stored in DB.

f is the classification model.

Y is the set of predicted classes.

V - visualization.

G is a graphical representation.

The mathematical problem of news classification

$$X = \{x_i = (x_{i1}, x_{i2}, \ldots, x_{iT}) \mid i = 1, 2, \ldots, n\}$$

$$Y = \{y_1, y_2, \ldots, y_n\} = \{0, 1, \ldots, n\}$$

$$K = n + 1$$

$$X^m = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$$

Training dataset

$$s_i \in S = \left\{ s \in \mathbb{R}^K \mid s_j \geq 0, \sum_{j=1}^{K} s_j = 1 \right\}$$

Probability vector for object x_i

$$\hat{y}_i = \arg\max_j (s_i)_j$$

Predicted class for object x_i

$$CM[p, q] = \sum_{i=1}^{m} I[y_i = p \wedge \hat{y}_i = q]$$

Confusion matrix

$$I = \begin{cases} 1 & \text{if condition is true} \\ 0 & \text{otherwise} \end{cases}$$

Indicator function

$$p \text{ — true class,} \quad q \text{ — predicted class}$$

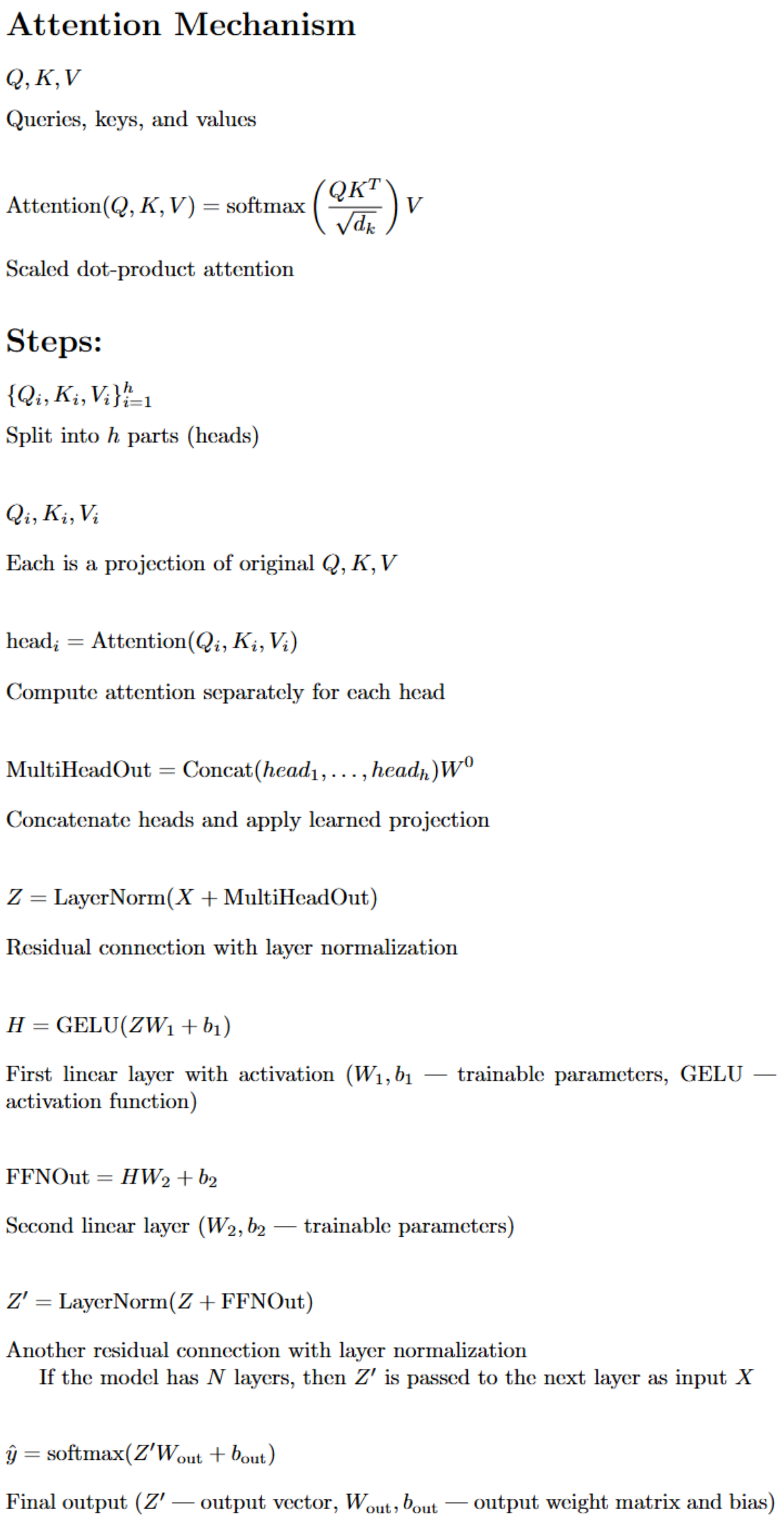
$$CM[p, q] \text{ — number of times class } p \text{ was predicted as } q$$

$$\sum_{k=0}^{n} CM[k, k]$$

Number of correct predictions across all classes

$$\text{Accuracy} = \frac{1}{m} \sum_{i=1}^{m} I[\hat{y}_i = y_i] = \frac{1}{m} \sum_{k=0}^{n} CM[k, k] \to \max$$

The BERT machine learning model was also used to solve this problem.



Attention Mechanism

Q, K, V

Queries, keys, and values

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Scaled dot-product attention

Steps:

$\{Q_i, K_i, V_i\}_{i=1}^{h}$

Split into h parts (heads)

Q_i, K_i, V_i

Each is a projection of original Q, K, V

$\text{head}_i = \text{Attention}(Q_i, K_i, V_i)$

Compute attention separately for each head

$\text{MultiHeadOut} = \text{Concat}(head_1, \ldots, head_h)W^0$

Concatenate heads and apply learned projection

$Z = \text{LayerNorm}(X + \text{MultiHeadOut})$

Residual connection with layer normalization

$H = \text{GELU}(ZW_1 + b_1)$

First linear layer with activation (W_1, b_1 — trainable parameters, GELU — activation function)

$\text{FFNOut} = HW_2 + b_2$

Second linear layer (W_2, b_2 — trainable parameters)

$Z' = \text{LayerNorm}(Z + \text{FFNOut})$

Another residual connection with layer normalization

If the model has N layers, then Z' is passed to the next layer as input X

$\hat{y} = \text{softmax}(Z'W_{\text{out}} + b_{\text{out}})$

Final output (Z' — output vector, $W_{\text{out}}, b_{\text{out}}$ — output weight matrix and bias)

In general, the task is to train the BERT machine learning model to classify news into specified classes.

1.6. Report Structure

The aim of this report is to provide a technical overview of our project and discuss any technical challenges that have arisen throughout its lifecycle. The report is structured as follows:

1. **Project Overview -** This section provides an overview of the project, including its goals and objectives.
2. **Dataset Description -** This section describes the dataset that was used for the project and the process of collecting and preparing it.
3. **Model Description -** This part details the BERT model that was employed in the project, as well as any issues that were encountered in its implementation.
4. **Testing Results -** This section presents the results of testing the model and evaluating its performance, as well as an analysis of the findings.

2. Project Overview

2.1. Usage Of The Project

For some companies, it is essential to analyze statistics on news from various sources and visualize them using dashboards in order to improve the perception of information. However, due to the vast amount of information available, it has become challenging to do so quickly and accurately in recent times. Our project aims to address this challenge.

By utilizing data parsing techniques, we aim to replace the need for human resources in reading news. Machine learning algorithms have the capability to classify data with accuracy. Well-designed dashboards allow for a better understanding of the information presented in an easy-to-digest manner.

2.2. Future Of The Project

In the future, this project could be utilized as a fully-fledged tool, for instance, to notify company executives about new developments that they may need to be aware of. For example, concerning legislation passed by the government that could affect the operations of the company. Naturally, it would be necessary to retrain the classification models for each individual user, but this would significantly free up their time.

3. Dataset Description

3.1. Selected Resource

For this project, we chose the RBC news website, which contains a large number of diverse news items organized into pre-defined categories. News updates are frequent, which facilitates the use of the most current information to train the machine learning model.

The convenience of this source is that the data is categorized into three types: title, overview/body, and text. This structure makes the data easy to understand and analyze.

3.2. Parsed Data version 1

The data parser was run multiple times in order to determine which method would be most efficient and provide the most relevant information. Ultimately, we selected these speakers based on the results of the analysis.

id	project	project_nick	type	category	title	publish_date	fronturl	picture	overview	text

id - News ID (677c3c699a79471ec81f21c2)

project - The name of the resource (RBC)

project_nick - The nickname of the project (Cyst)

type - The size of the news (Article, Short News)

category - The class to which the news belongs (Society, Politics)

title - The headline of the news

publish_date - Date of publication (2025-04-01 21:47:59)

fronturl - Link to the news

(https://www.rbc.ru/society/06/01/2025/677c3c699a79471ec81f21c2)

picture - Presence of pictures (0, 1)

overview - A small summary of the news

text - The full text of the news

This dataset contained 13 columns and 15,599 rows.

All of these columns contain important information for various aspects of the final product. For instance, for the BERT machine learning model, a news text is crucial, and for visualization, the date and link to the news are significant.

However, there were some data issues with parsed data, so data preprocessing was performed.

3.3. Data Preparation for version 1

3.3.1. Nan Values

Initially, I decided to examine the full dataset to get a better understanding of its structure. Upon visual inspection, it became apparent that there were some gaps in the data. For example, some news items lacked an overview, while others lacked information altogether. In order to ensure an adequate analysis, I determined that it would be necessary to exclude those news items from the dataset that contained omissions in any of the three columns. After eliminating 19 items that were largely devoid of content, a total of 15,580 news items remained. For news items that lacked any of the required columns, no further action was taken.

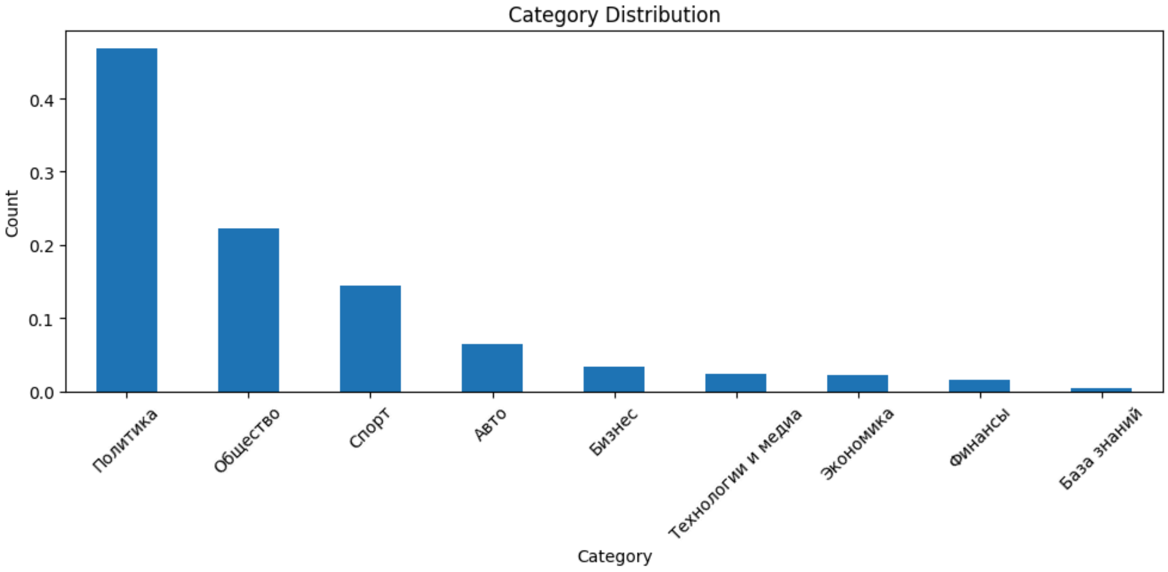
3.3.2. New Column

- To use only one column containing text, I created a single "full-text" column that combines information from all text-based columns (title + overview + text), due to the model's requirements.
- This approach ensures the integrity of the context, as BERT learns best from complete data, analyzing the context and relationships between words. By combining all relevant data into a single column, we significantly improve the accuracy and completeness of feature extraction.

- An important aspect of the tokenization process is the use of a special tokenizer by BERT, which breaks text into subwords. This allows BERT to accurately capture the context and positional encodings of each word and subword in a sentence. If data were split across multiple columns, the tokenizer might not be able to perform this task effectively.
- Ease of data processing: Combining data can significantly simplify the data pipeline and reduce the complexity of the processing process. Thanks to the use of a single column, there is no need for running data through a tokenizer three times, creating three separate inputs, and developing a custom aggregation process.
- Reduction of computational redundancy in the architecture: If you need to process three texts, there are two options. You can either make three passes through the BERT model and then concatenate or average the results, or you can build a more complex multimodal architecture. However, this latter approach is either significantly slower or more challenging to optimize.

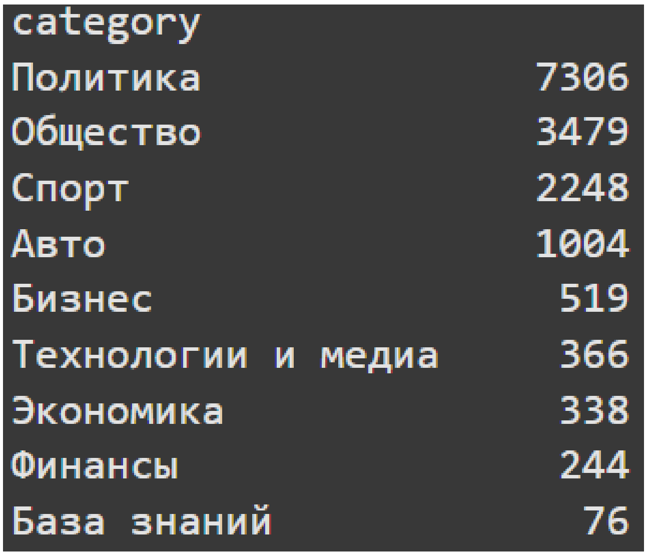
3.3.3. Data Analysis version 1

Next, I decided to investigate whether the data was well-balanced across categories. Using matplotlib for graph plotting, I observed the following:



This graph demonstrates a significant data imbalance across categories. There is a much higher volume of news related to politics compared to any other category.

Additionally, it can be observed that there are a total of 9 distinct categories that will require classification.



```
category
Политика               7306
Общество               3479
Спорт                  2248
Авто                   1004
Бизнес                  519
Технологии и медиа      366
Экономика               338
Финансы                 244
База знаний              76
```

This screenshot illustrates the number of news articles in each category. There is a significant difference in the number of articles in the "Politics" category compared to the "Knowledge Base" category, with the latter having approximately 100 times fewer articles. This observation led me to hypothesize that the model may have difficulty accurately classifying certain categories when using the standard train-test data split.

3.3.4. Train And Test Datasets Creation version 1

When creating the training and test datasets, I considered ways to reduce data imbalance in order to more accurately train the model. I proposed two ideas:

- Creating a data slice to ensure that the difference in training data size is not as significant as in the original dataset.
- Generating examples for underrepresented classes using synonymization.

Initially, I attempted to generate examples, as this would provide more diverse content for the training set, potentially leading to improved accuracy. Through research, I identified several methods:

- Utilizing a lexical database (WordNet) to identify synonyms.
- Employing large language models (LLMs) such as Llama3.

When using a lexical database, a problem was encountered where news was being duplicated instead of replaced with synonyms. Upon further investigation, it was discovered that the database only contains information on English vocabulary and therefore, it is not possible to find synonyms for Russian words.

Attempts to use analogues in the Russian language were unsuccessful, as none were available or free. Using a machine translator was also not feasible, as it would result in the loss of news integrity and the possibility of producing inaccurate translations that could negatively impact the model's performance.

Using a large language model may have been beneficial, but it would have required a significant amount of time and effort. Given the need to generate a large number of applications, and considering that the dataset may not yet be final and there may be further training required for the model, it was impractical to expend such resources at this time. Therefore, I have placed this idea on the backlog in order to consider it for implementation in a future version of the product when the required resources are available.

In light of these considerations, I have decided to attempt to create a data subset in order to minimize the difference in news volume. Based on an analysis of the number of news items in each category, I have determined the following cross-sectional approach.

Category	№ of news
Политика	1000
Общество	500
Спорт	500
Авто	500
Бизнес	100
Технологии и медиа	100

Экономика	100
Финансы	100
База знаний	50

We received 2,950 news items for the train dataset. Additionally, the imbalance in the number decreased from 100 to 20 times between the most frequent and rare categories. Furthermore, a significant advantage of this method is to reduce the training time for the model by several times due to a reduction in the amount of data. For the test dataset, we decided to utilize all remaining applications. In this regard, the evaluation turned out to be fairly balanced, as for the "Knowledge Base" category, using the typical train_test_split would result in approximately 7-10 applications remaining, and the evaluation would be highly reliant on which applications were included in the training dataset. Therefore, this method has numerous benefits, and we have chosen to continue using it.

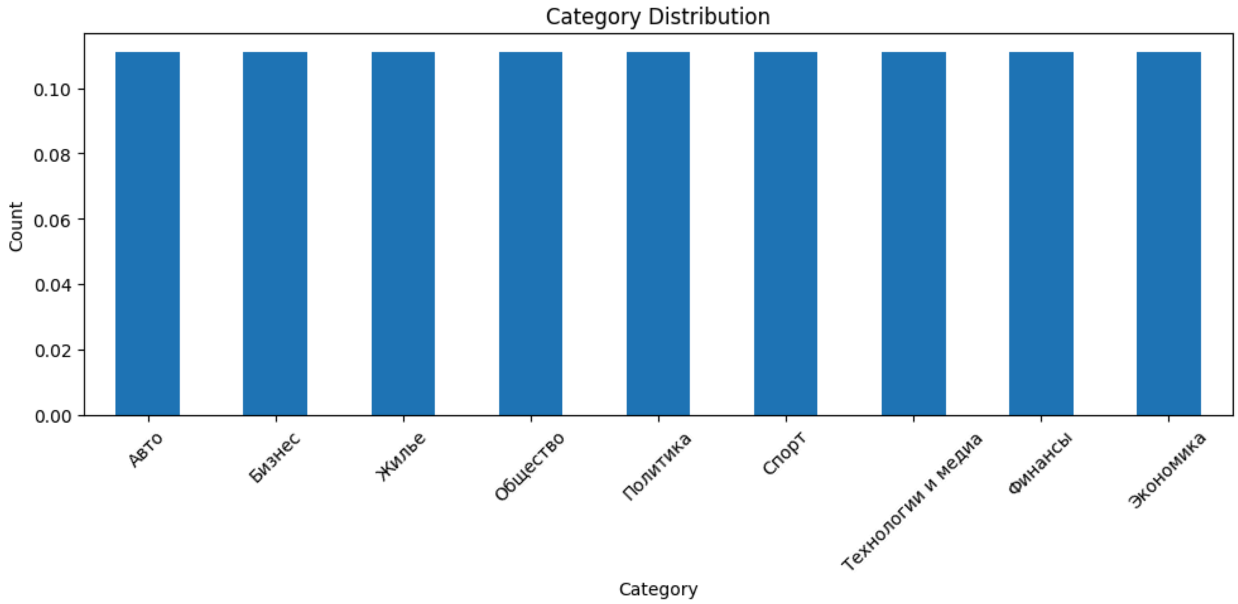
3.3.5. Tokenization

Tokenization within the DeepPavlov/rubert-base-case model is carried out using a pre-trained BertTokenizer based on the WordPiece algorithm. The text input is converted into a sequence of 256 tokens using this tokenizer, which includes special tokens [CLS] and [SEP]. If the input text exceeds the maximum length, it will be truncated, and if it is shorter than the minimum length, it will be padded. This process ensures compatibility with the BERT architecture and allows for efficient training of the model on a GPU.

Currently, data preparation is nearing completion, and the BERT model will be trained to classify news into categories.

3.4 Parsed Data version 2

Following the initial working prototype, we endeavored to create a more harmonious dataset. The data source from the RBC website was employed as well, but adjustments were made to the quantity of news articles and their categorization.



The screenshot illustrates the distribution of news articles across various topics relative to the overall number, expressed as a percentage. In order to evaluate the performance of the model against its previous iteration, we have opted to allocate an equal quantity of news content to each category. Specifically, we have determined to utilize 600 articles for each category. Additionally, we have eliminated the "База знаний" category, which lacked substantial data, and opted to incorporate a new "Жилье" category, as it contains a greater volume of information, potentially enhancing the accuracy of predictions.

3.5 Data Preparation version 2

For this dataset, owing to the balanced nature of the data, it was not necessary to manually specify the number of news items in each category for training the machine learning model. This significantly accelerated the process. Additionally, there were no missing nan values, indicating that the data parsing process is functioning effectively.

Furthermore, I created a new column similar to version 1. To generate the training and testing datasets, I employed the train_test_split functionality provided by the scikit-learn library with proportion 80% train, 20% test

3.6 Tokenization version 2

Following the training of the first model, I chose to examine the average token count based on the news data, revealing a significantly higher number than 256. Therefore, for the second model, I opted to utilize the maximum token limit compatible with the model, which is 512. My rationale behind this decision was the possibility that the model may discard crucial information, potentially leading to issues. The tokenization process was executed in a manner similar to version 1, employing a pre-trained BertTokenizer.

4. Model Description

For this project, the DeepPavlov/rubert-base-case model was used. This is a Russian-language adaptation of the BERT model and it was trained on a labelled dataset (train). The model is capable of automatically classifying news articles based on their content.

4.1. Overview Of BERT Architecture

The BERT architecture, proposed by Google in 2018, is a bidirectional neural network that utilizes the self-attention mechanism. Unlike previous models, BERT processes the context of a word from both the left and right sides simultaneously, allowing it to better understand the semantic meaning of a word in relation to the entire sentence.

The fundamental BERT model consists of:

- 12 layers of transformer blocks
- 12 self-attention heads in each layer
- Positional encoding to preserve word order
- A special [CLS] token, which represents the aggregated representation of the entire input sequence
- Word, segment, and position embeddings

4.2. Russian Language Adaptation: DeepPavlov ruBERT

The DeepPavlov/rubert-base-case model is a fully-trained version of the BERT model, adapted for use with the Russian language. The model was trained using large-scale Russian language data, such as from Wikipedia, news websites, and other open sources.

The "cased" version of the model means that it preserves the distinction between upper- and lower-case letters, which is significant for the Russian language because case can carry important information, such as proper names.

Adaptation of the model to the Russian language ensures that it takes into account the morphological features of Russian, including support for cases, conjugations, and declensions. This, in turn, improves the model's understanding of syntactic and other relationships between words in Russian.

4.3. Model Head: Sequence Classification

To address the classification challenge, the ruBERT model has been augmented with an output layer for classification. The resulting architecture is implemented via the BertForSequenceClassification class and incorporates the following components:

- A dropout layer, which prevents overfitting
- An nn.Linear layer, which transforms the hidden representation of the [CLS] token into a vector with a dimension equal to the number of classes
- A Softmax layer, which converts the output into probabilities for each class
- The CrossEntropyLoss learning loss function, which is typical for multiclass classification problems.

4.4. Training Strategy version 1

The learning process is facilitated through the use of the Trainer class from the Hugging Face library, which provides a high-level interface for the training of transformer models. This class automatically handles batching, optimizer initialization, logging, and model saving.

Key training parameters include:

- Total number of epochs: 20
- Batch size: 8
- Optimizer: AdamW
- Loss function: CrossEntropyLoss
- Metrics: Accuracy and Macro F1-score

At the moment, the model is in the stage of development and testing methods of working with data and hyperparameters. In order to observe changes in metrics of the model during training, the model was run for a large number of epochs. Based on the results, it was decided which number of epochs would be most appropriate for this model.

A cycle of epoch training was implemented, with classification_report output after each epoch and metrics saved for further analysis. The best model, if accuracy and F1 have improved, was also saved. The final version of the model was preserved throughout training.

The preservation of the model is justified by the following factors:

- To avoid repetitive training. If a model meets all metrics and can be deemed successful, it should be saved to prevent the need for re-training. Training a BERT model involves significant computational and time investment. Saving the results will allow for the preservation of all learning outcomes, enabling their future use for further work, thereby avoiding wastage of time and resources.
- The saved model can then be utilized for predictions. As it can be loaded anytime, it can classify new texts, integrate with existing systems, and serve as the basis for building an API.
- Additionally, the ability to compare various versions of a model, as well as different models in general, becomes possible through saving. This allows for a quick analysis of each model's performance and understanding of how they behave on new data or in real-world scenarios.
- Furthermore, storing the trained model in file format facilitates sharing of work results and uploading to necessary servers without the need for additional training.

4.5 Training Strategy version 2

I employed the same training methodology, but I opted to incorporate the early_stopping criterion, thereby eliminating the need to continue training the model in the event of no improvement in metrics for two epochs. My decision also involved setting the number of training epochs at ten, and upon reaching epoch four, it became evident that there was no further improvement, rendering any additional training beyond twenty epochs futile.

4.6. Evaluation Metrics And Model Selection

The following metrics were used to evaluate the model during training:

- Accuracy
- Macro F1-score

Accuracy is a metric that represents how many objects are correctly classified by the model, compared to the total number in the sample. It provides a quick way to evaluate the performance of the model.

Macro F1-Score is the average of F1 measures calculated separately for each class, and then averaged together. This metric considers all classes equally, regardless of how many examples they have in the training set. It is resistant to imbalanced class distributions and reflects the ability of the model to recognize both common and rare classes.

By combining these two metrics, we can assess both the overall performance of the model and its fairness across all classes. If the accuracy is high but the F1 score is low, it may indicate that the model is ignoring some classes.

4.7. Justification For Model Choice

Reasons for choosing the ruBERT model:

- High performance of transformers in natural language processing tasks
- Bidirectional attention mechanism, enabling the model to fully consider the context of a word
- Pre-training on a significant amount of data in the Russian language

- Ease of adaptation for specific tasks
- Integration with modern natural language processing libraries (e.g., Transformers)

Compared to classical models (logistic regression, SCM), the BERT and its Russian language variations demonstrate significantly higher accuracy in text classification tasks.

The study "Topical Text Classification of Russian News: A Comparison of BERT with Standard Models" evaluates the performance of the BERT model against traditional algorithms such as logistic regression and SVM for thematic classification of Russian news. The findings indicate that BERT outperforms classical models in terms of accuracy and F1-score.

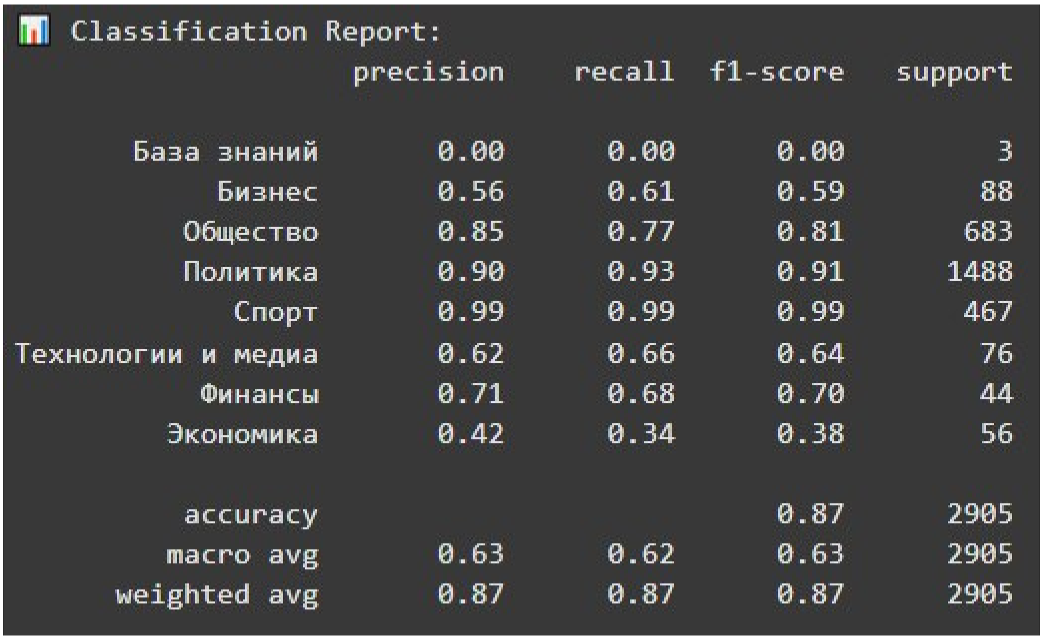
4.8. Summary

DeepPavlov/rubert-base-cased model is a powerful tool for text classification in Russian. It achieves this by pre-training on large datasets and utilizing a transformer architecture, which effectively captures complex semantic relationships between words and phrases. This allows the model to adapt to the specific requirements of a given task, ensuring high levels of accuracy and consistency in predictions.

5. Testing Results

5.1. First Version

Initially, the data presented some differences, and I used a relatively simple BERT model without extensive analysis of the dataset. The data was simply divided into training and testing sets. Despite this, the model produced relatively positive results, indicating that BERT is well-suited for this task.

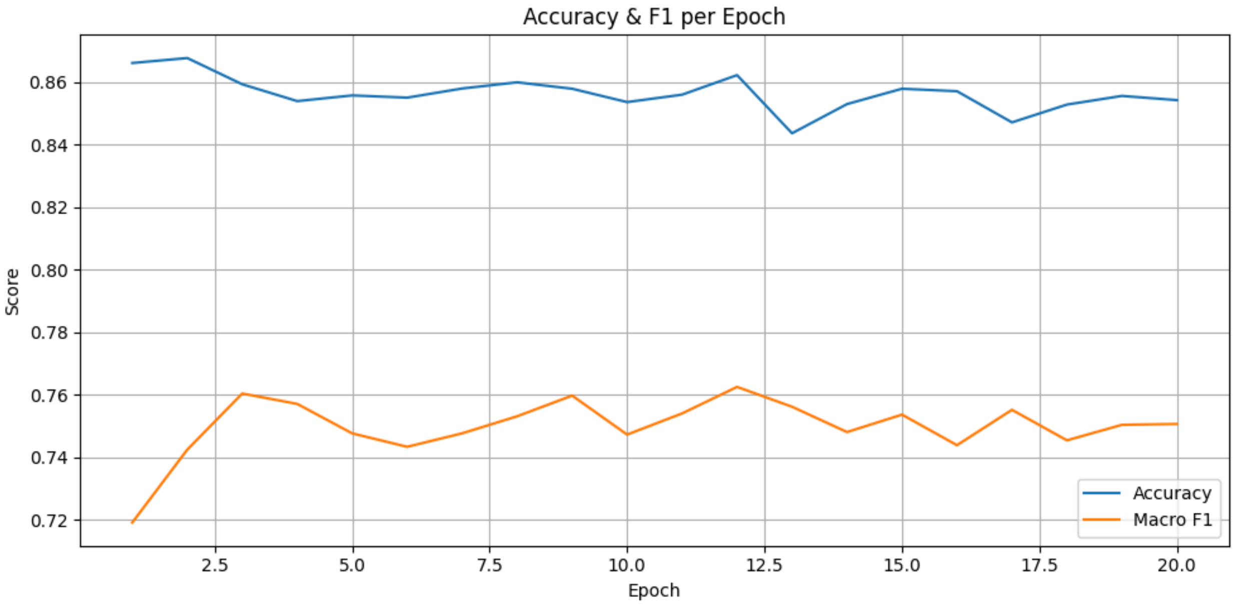


```
📊 Classification Report:
                     precision    recall  f1-score   support

        База знаний       0.00      0.00      0.00         3
             Бизнес       0.56      0.61      0.59        88
           Общество       0.85      0.77      0.81       683
           Политика       0.90      0.93      0.91      1488
              Спорт       0.99      0.99      0.99       467
Технологии и медиа       0.62      0.66      0.64        76
            Финансы       0.71      0.68      0.70        44
          Экономика       0.42      0.34      0.38        56

           accuracy                           0.87      2905
          macro avg       0.63      0.62      0.63      2905
       weighted avg       0.87      0.87      0.87      2905
```

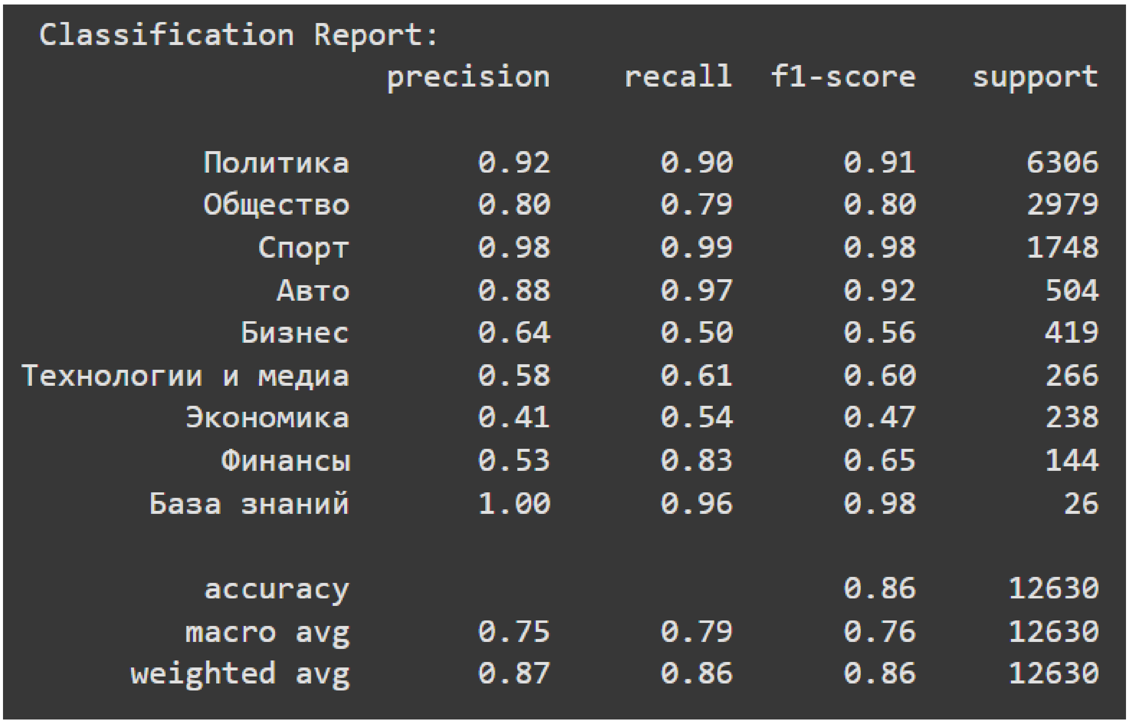
Here is an example of the results. There are fewer classes in this case, as the imbalance was even more significant, and there were also issues with parsing, resulting in no data for the "Авто" class. If we examine the results, the overall accuracy of the model has turned out to be 0.87, which is quite good for such a significant imbalance. However, there is a concern that the model is ignoring the "База знаний" category. The support column indicates that there were only three examples of this category in the test dataset, hence no results for it. Considering the other categories, we can see a notable dominance of "Политика," "Общество," and "Спорт." This is due to the relatively large number of news articles in these categories, which allowed the model to predict them with greater accuracy. As in the test dataset, there are numerous examples of these categories, helping to provide a realistic assessment. For other categories, the results may not be as promising as we would prefer. However, the most concerning aspect is that the model fails to attend to one of the input classes, which is crucial for our task.

5.2. Second Version

After reviewing the results of the initial attempt, it was determined that preprocessing of the data was necessary, and, in general, there were slight changes to the data, which provided more instances for certain categories. Consequently, upon launching the model, the following occurred:



I decided to train the model for a total of 20 training epochs, after which I would select the number of training epochs that produced the best metric results. Based on the visual analysis, the model exhibited its "strongest" performance at the 12th training epoch. While the accuracy improved with additional training, the F1 score decreased, indicating that the model was not able to distinguish between classes with smaller dimensions as well.

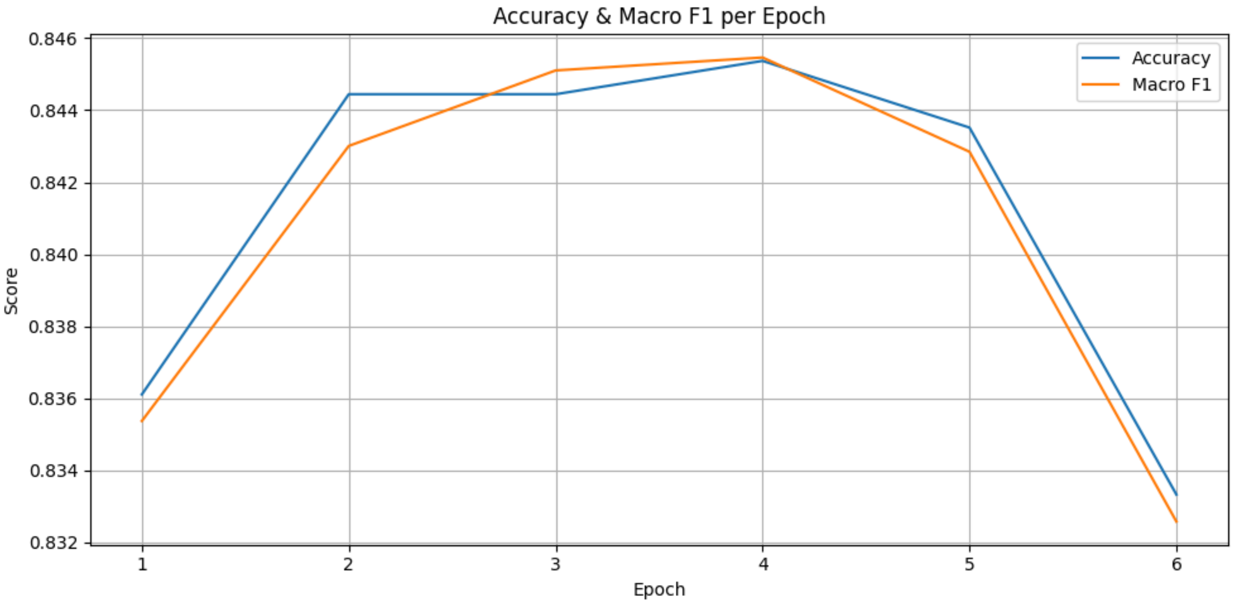


```
Classification Report:
                   precision    recall  f1-score   support

          Политика       0.92      0.90      0.91      6306
          Общество       0.80      0.79      0.80      2979
             Спорт       0.98      0.99      0.98      1748
              Авто       0.88      0.97      0.92       504
            Бизнес       0.64      0.50      0.56       419
Технологии и медиа       0.58      0.61      0.60       266
         Экономика       0.41      0.54      0.47       238
           Финансы       0.53      0.83      0.65       144
       База знаний       1.00      0.96      0.98        26

          accuracy                           0.86     12630
         macro avg       0.75      0.79      0.76     12630
      weighted avg       0.87      0.86      0.86     12630
```

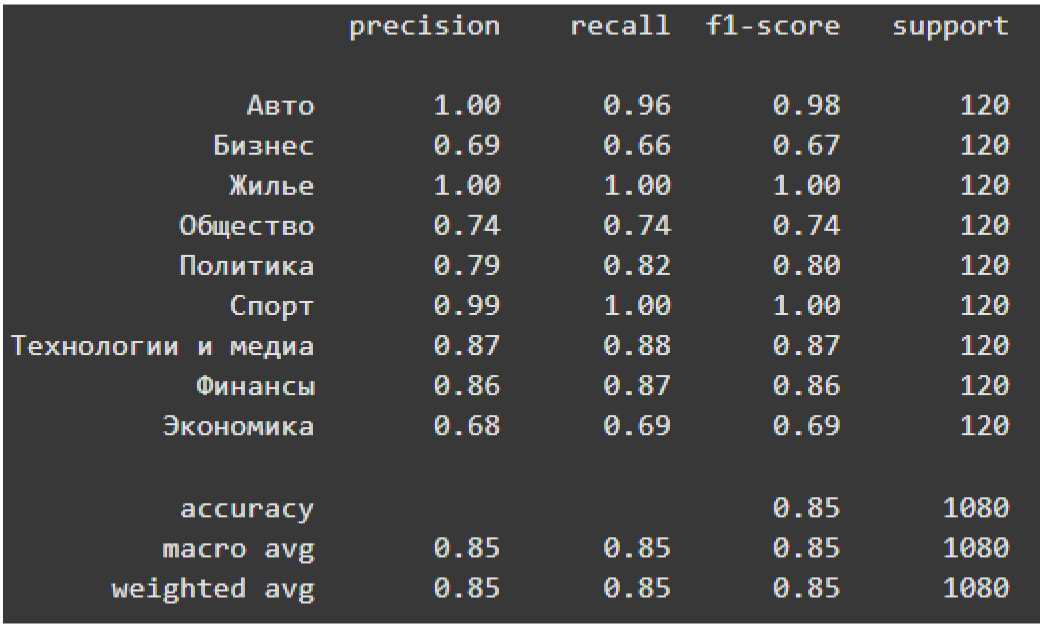
Here are the results for each category. While some categories still have room for improvement, they are significantly better than they were previously. The model identifies all categories and has an overall accuracy rate of 0.86. While the difference in accuracy may not be significant, the ability of the model to recognize all categories is promising. To further improve the accuracy of individual category recognition, I will continue to add more data for these categories (Бизнес, Технологии и медиа, Экономика, Финансы). This should provide the model with a better understanding of these categories, and therefore enhance the evaluation. Another potential issue is overfitting. Scores for the "Спорт" and "База знаний" categories are too high, which could indicate that the model has overtrained, or that there is a significant difference in the texts of news stories in these categories, which contributes to improved learning and prediction.

5.3 Final Version

I kept the learning logic, and that's why I was able to deduce metrics by epoch.



Here it is noticeable how after 4 epochs, the metrics began to fall sharply, so the choice of 4 epochs of learning was correct.



```
                  precision    recall  f1-score   support

             Авто       1.00      0.96      0.98       120
           Бизнес       0.69      0.66      0.67       120
            Жилье       1.00      1.00      1.00       120
         Общество       0.74      0.74      0.74       120
         Политика       0.79      0.82      0.80       120
            Спорт       0.99      1.00      1.00       120
Технологии и медиа       0.87      0.88      0.87       120
          Финансы       0.86      0.87      0.86       120
        Экономика       0.68      0.69      0.69       120

         accuracy                           0.85      1080
        macro avg       0.85      0.85      0.85      1080
     weighted avg       0.85      0.85      0.85      1080
```

This screenshot illustrates the performance metrics of the model during the fourth epoch of training. The overall accuracy of the model appears to be lower compared to that of the second model iteration. However, this discrepancy can be attributed to the fact that the test dataset employed in this iteration was smaller, resulting in a more pronounced impact of model errors on the overall accuracy. It is worth noting that when a dataset is well-balanced, such effects become more evident.

It is evident that there are no severe discrepancies in the metrics for certain categories, as was the case previously, for instance, in the "Economics" category.

The model now exhibits a more balanced performance, albeit with some categories still being predicted less accurately than others. Nonetheless, it is encouraging to see that none of the metrics fall below 0.65, indicating a high level of comprehension of the data by the model. Moreover, this suggests that the model has the capacity to accurately predict almost all categories presented during training.

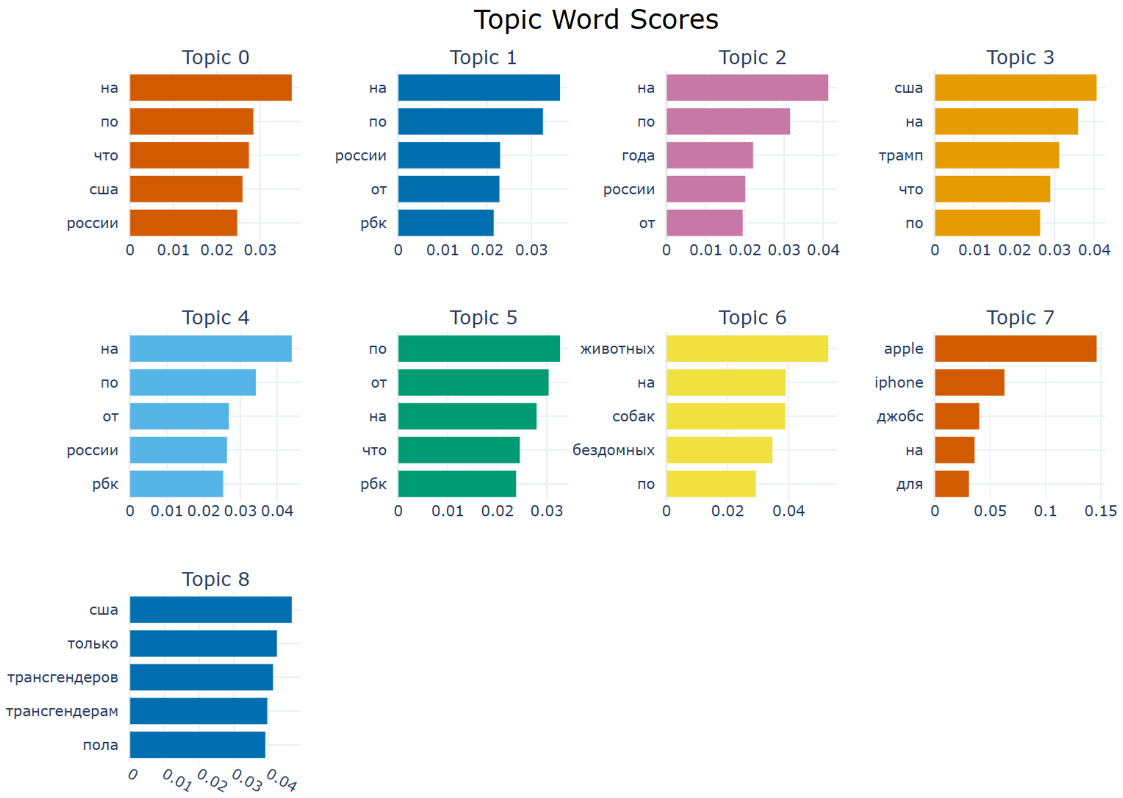
5.4. What Should Be Improved

To ensure that the model functions more accurately, the quality of the input data is of significant importance. Therefore, when constructing a working version of the model, it will be necessary to address the existing errors. For instance, a significant data imbalance may need to be addressed. It may be beneficial to collect more data for less frequent classes, which could aid in training the model. Additionally, the possibility of re-training the model should be considered. To better understand which classes significantly differ from the rest and warrant higher priority, it is essential to delve deeper into data analysis during training.

5.5. Troubled Model

I also attempted to address the topic modeling challenge using the BERT.Topic model, but I was unable to fully understand how to utilize it in a more effective manner. The results were not satisfactory, and there is room for improvement. In general, the model assists in identifying key words for each class, which should aid in creating subcategories for applications. However, due to the original assignment being a classification task, this task has been assigned to the backlog.

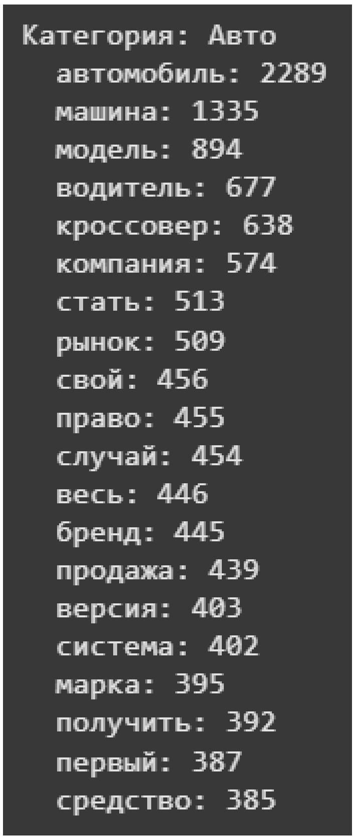
Here is an example of how BERT.Topic works:



The issue with the content in terms of topic relevance is quite evident. There is a significant number of prepositions and other words that do not contribute to the identification of a specific subcategory, therefore, it will be necessary to conduct a more comprehensive review of the text. This includes the removal of prepositions and the use of lemmatization in order to accurately identify popular words related to the various topics.

5.6. Frequency Words

I also compiled lists of the most prevalent words in each category. To accomplish this task, I first lemmatised all the words in the news articles, after which I counted the frequency of each word occurrence in each category. The initial intention was to attempt to categorise news articles based on these words, but this proved unsuccessful, resulting in a list of the twenty most frequently occurring words in the dataset. This information can be utilised for the future development of the project and the creation of subcategories. Here is the example:



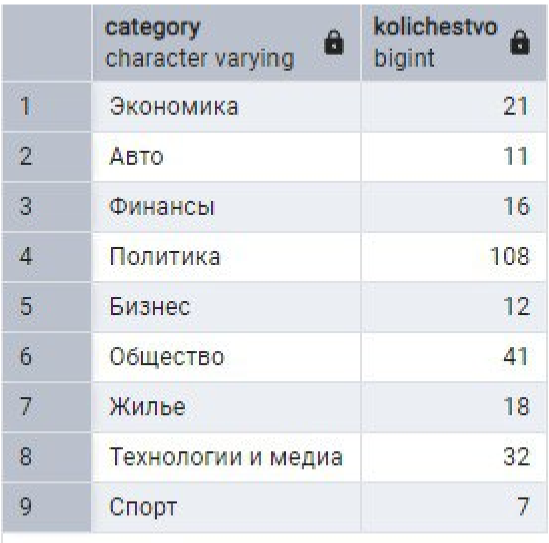
```
Категория: Авто
  автомобиль: 2289
  машина: 1335
  модель: 894
  водитель: 677
  кроссовер: 638
  компания: 574
  стать: 513
  рынок: 509
  свой: 456
  право: 455
  случай: 454
  весь: 446
  бренд: 445
  продажа: 439
  версия: 403
  система: 402
  марка: 395
  получить: 392
  первый: 387
  средство: 385
```

In general, the most logical words “автомобиль” and “машина” are found most often, which means that the way to count such words works as it should.

5.7. Telegram Predictions

Following the completion of the final model's saving, I proceeded to upload its weights to a designated server. Subsequently, I employed the model to forecast the news I had retrieved from the RBC Telegram channel, aiming to assess its performance. The results indicated that all categories employed during training were incorporated in the predictions. This suggests that the model is capable of accurately predicting all categories based on the provided data.

In a previous attempt, I encountered an issue during testing the model's ability to predict Telegram news, where only two categories were identified, while the remaining ones remained undetected.



	category character varying	kolichestvo bigint
1	Экономика	21
2	Авто	11
3	Финансы	16
4	Политика	108
5	Бизнес	12
6	Общество	41
7	Жилье	18
8	Технологии и медиа	32
9	Спорт	7

It is an example of distribution of predicted data from telegram channel.

References

1. **Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.** https://arxiv.org/abs/1810.04805
2. **Kuratov, Y., & Arkhipov, M. (2019). Adaptation of BERT for Russian language: DeepPavlov Library.** https://arxiv.org/abs/1905.07213
3. **Hugging Face Transformers Documentation.** https://huggingface.co/docs/transformers
4. **Ksenia Lagutina. Topical Text Classification of Russian News: a Comparison of BERT and Standard Models** https://www.researchgate.net/publication/360545974_Topical_Text_Classification_of_Russian_News_a_Comparison_of_BERT_and_Standard_Models
5. **Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.** https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html
6. **WordNet: A Lexical Database for English. Princeton University.** https://wordnet.princeton.edu/
7. **DeepPavlov Documentation – ruBERT.** https://docs.deeppavlov.ai/en/master/features/models/bert.html
8. **Hugging Face – Model Hub (ruBERT).** https://huggingface.co/DeepPavlov/rubert-base-cased

Applications

GitHub - https://github.com/aparinov/25_HSE_BigTextsAnalysis