News classification with Logistic Regression, BERT and GPT

Artyom Radchenko

December 2023

Abstract

In this project I investigated several solutions for News classification. A corpus of Russian-language news dedicated to Chines AI was collected within "Китайский Искусственный Интеллект" (`https://t.me/chinese_ai_news`) news aggregation project. These news are divided on two classes: "Published" and "Not published". The problem was solved using three natural language processing approaches: TF-IDF and Logistic Regression, BERT and GPT. A comparative analysis of the results of these approaches was carried out.

Repository of this project is located in `https://github.com/ArtyomR/AI-News-Classification`.

1 Introduction

Text classification problems have been widely studied and addressed in many real applications over the last few decades. Especially with recent breakthroughs in Natural Language Processing (NLP) and text mining, many researchers are now interested in developing applications that leverage text classification methods. Most text classification and document categorization systems can be deconstructed into the following four phases: (1) Feature extraction, (2) dimension reductions, (3) classifier selection, and (4) evaluations. [Kowsari et al., 2019]

Text classification systems in terms of the pipeline illustrated in Figure 1. [Kowsari et al., 2019]

In this document I am going to concentrate on classifier selection stage. I would like to test and compare Logistic Regression and BERT approaches.

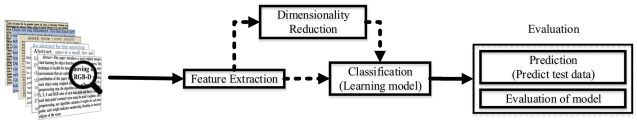

Figure 1: Overview of text classification pipeline.

On another hand last year solutions based on GPT models became very popular for different tasks of Natural Language Processing. It was very interesting for me to check usage of GPT for Text classification task. I didn't find big number of articles regarding implementation of GPT for text classification.

Same time I was looking for News classification solution for "Китайский Искусственный Интеллект" (`https://t.me/chinese_ai_news`) news aggregation project.

So there are two main goals in this project:

- To choose News classification solution (classifier) for "Китайский Искусственный Интеллект" (`https://t.me/chinese_ai_news`) news aggregation project.

- To check GPT model for News classification task

1.1 Team

Artyom Radchenko created and prepared dataset, developed and adopted software and models for implementation of solutions and prepared this document.

2 Related Work

According to [Kowsari et al., 2019] there are big number of solutions for Text classifier. I would like to propose following grouping of Text classifier algorithms basing on [Kowsari et al., 2019] and [Pathak, 2022]: ‘

- **Non-deep learning algorithms:** Rocchio classification, Naive Bayes Classifier, K-nearest Neighbor, **Logistic Regression**, Support Vector Machine (SVM), Decision Tree, Random Forest, Conditional Random Field (CRF)
- **Deep learning algorithms:** Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN), Recurrent Convolutional Neural Networks (RCNN), Random Multimodel Deep Learning (RMDL) etc.
- **Transformers:** ELMO, BERT, ROBERTA, DISTILBERT, XLNet, **rubert-tiny2** etc.
- **GPTs:** GPT-2, GPT-3.5, GPT-4, Mistral, **Saiga** etc.

‘

According to [YULIANTO, 2022] **Logistic regression** shows very good performance among Non-deep learning algorithms.

According to Text Classification on AG News Leaderboard leaders for Text Classification on AG's News Corpus are XLNet and BERT-ITPT-FiT. See Figure 2.

For Russian news corpus (News Dataset from Lenta.ru) there is comparison of Text classifiers in [Chelyshev et al,2022]. Naive Bayes, Logistic regression, Random forest of decision trees and Artificial neural network were compared in

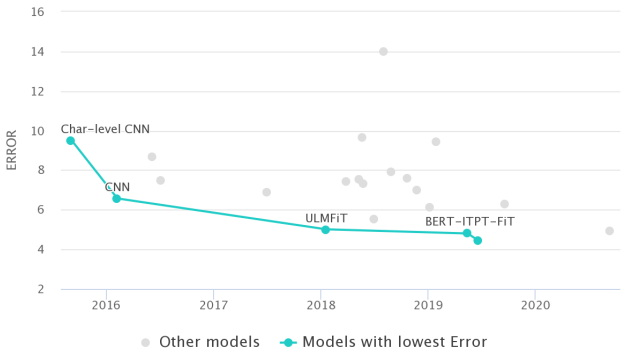

Figure 2: Text Classification on AG News Leaderboard

this document. Logistic regression and Artificial neural network shown the best performance.

There is an article regarding implementation of BERT classifier for Russian texts "BERT для классификации русскоязычных текстов"

I did not find any comparison or leaderbord for GPT-based text classifiers. I decided to use Question Answering solution described in the article "Русский LLM-помощник (saiga) с кэшем, используя RAG (Retrieval-Augmented Generation)"

3 Dataset

The Dataset was created within "Китайский Искусственный Интеллект" (`https://t.me/chinese_ai_news`) news aggregation project. It could be downloaded from github. Data are combined from titles and first paragraphs of news dedicated to Chinese high technologies. Originally these news were in English or Chinese. Then they were translated into Russian by Google translation engine. Raw data are shown on Figure 3.

	publication	url	title	title_ru	paragraph_one	paragraph_one_ru	publish_date
0	Published	https://www.scmp.com/tech/article/3242683/brit...	British AI chip darling Graphcore pulls out of...	Британский разработчик чипов искусственного ин...	The British graphics processing unit maker Gra...	Британский производитель графических процессор...	2023-11-24 15:00:09
1	Published	https://www.scmp.com/week-asia/people/article/...	Asia-Pacific is new 'ground zero' for cybercri...	Азиатско-Тихоокеанский регион стал новым «эпиц...	The 'attack rate' in the Asia-Pacific is 'well...	«Уровень атак» в Азиатско-Тихоокеанском регион...	2023-11-24 17:00:46

Figure 3: Example of raw data in dataset.

After pre-processing and lematization dataset looks like on Figure 4.

It contains 'Not published' news - 1711 (55.48%) in this data set and 'Published' news - 1373 (44.52%) in this dataset.

	published_flag	combined_text_ru	combined_text_ru_clean
723	1	Школьный гимн и музыка на электрогитаре, котор...	школьный гимн и музыка на электрогитара которы...
2997	0	Недавний минимум в 185 юаней: самостоятельная ...	недавний минимум в 185 юань самостоятельный по...

Figure 4: Example of data after pre-processing.

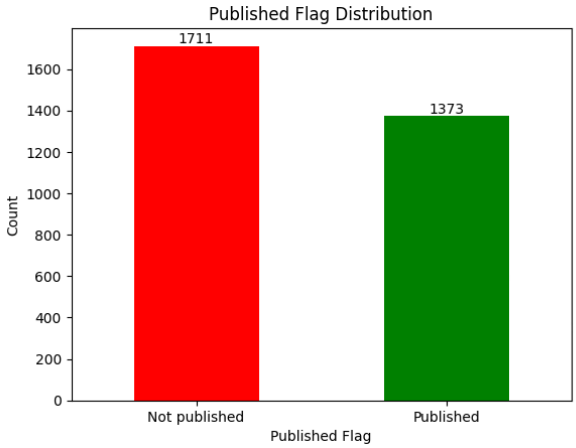

Figure 5: Published flag distribution

This dataset was divided on train dataset (2158 records - 70%) and test dataset (926 - 30%).

For Transformer (rubert-tiny2) model train dataset was divided on 2 subsets: train subset - 1726 records, validation subset - 432 records.

4 Model Description

I decided to use three train models: Logistic Regression, BERT-based Transformer and GPT-based model.

4.1 Logistic Regression

As a simple approach for text classification I have chosen logistic regression over TF-IDF embedding. This approach is going to be considered as baseline. First I have pre-processed the data. The pre-processing consisted of text lemmatisation with **pymorphy3** library.

Then **TfidfVectorizer** from the **sklearn** library was used to get a number for each word. The goal of using TF-IDF is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus. The formula describing TF-IDF embedding:

$$\text{idf}(t) = \log \frac{n}{\text{df}(t)} + 1$$

where n is total quantity of documents in document corpus and $df(t)$ is quantity of documents containing word t.

The resulting vectors of original comments samples became the inputs to the **LogisticRegression** model imported from **sklearn** library.

For notational ease, we assume that the target y_i takes values in the set $\{0, 1\}$ for data point i. Once fitted, the predict method of **LogisticRegression** predicts the probability of the positive class $P(y_i = 1|X_i)$ as

$$\hat{p}(X_i) = \text{expit}(X_i w + w_0) = \frac{1}{1 + \exp(-X_i w - w_0)}.$$

As an optimization problem, binary class logistic regression with regularization term $r(w)$ minimizes the following cost function:

$$\min_w C \sum_{i=1}^{n} (-y_i \log(\hat{p}(X_i)) - (1 - y_i) \log(1 - \hat{p}(X_i))) + r(w).$$

4.2 Transformer (rubert-tiny2)

The transformer-based language models have been showing promising progress on a number of different natural language processing (NLP) benchmarks. The combination of transfer learning methods with large-scale transformer language models is becoming a standard in modern NLP. Therefore, the next stage of our research was the use of pre-trained transformers of the BERT type. The rubert-tiny2 (sentence encoder model) model was chosen for classification. Since at the moment, among the Russian-language models of the sentence encoder, this model wins in terms of the balance of speed and quality For solution implementation the pre-trained tokenizer and model were loaded from Hugging Face rubert-tiny2. For classification, it was necessary to add a fully connected layer, the number of inputs of which is the internal dimension of the embedding of the network, and the output is the number of classes for classification.

4.3 GPT (Saiga/Mistral)

It is expected that GPT models will become universal solution for most of NLP tasks. In order to test it I implemented Retrieval-Augmented Generation (RAG). RAG allows to provide questions (prompts) together with context information. RAD solution architecture is shown on Figure 3. This information is taken from Русский LLM-помощник (saiga) с кэшем, используя RAG (Retrieval-Augmented Generation)

Following steps are shown on the Figure 3.

1. Prompt (text information) comes from dataset pandas DataFrame. In my solution this text information is news information. And it is used as context for GPT model.

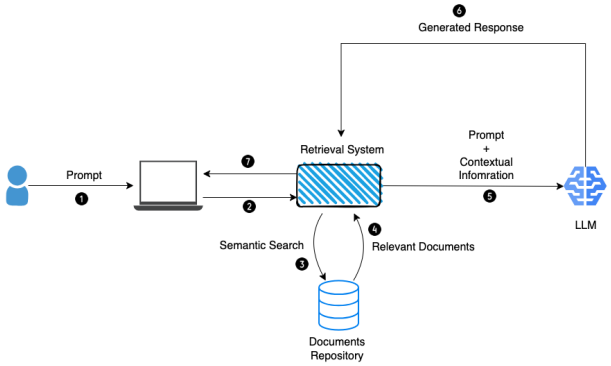

Figure 6: RAG solution architecture

2. The Prompt goes to Retrieval System. It adds Qustion "Содержит ли этот текст информацию о китайском искуственном интеллекте?" ("This text contains information about Chinese artificial intelligence, right?" in Russian)
3. Is not implemented in this version of solution.
4. Is not implemented in this version of solution.
5. Then the Prompt goes to LLM (GPT model).
6. LLM generates Response: "Да" (Yes) or "Нет" (No).
7. The Retrieval System converts this Response to 0/1 codes. These codes are used as prediction.

langchain library is used as RAG addendum for LLM.
Mistral-7B-OpenOrca is used as LLM base model.
saiga-mistral-7b-lora is used as adopted weights.

5 Experiments

5.1 Metrics

5.1.1 F1-score

In this project the F1-score was used as metric to evaluate the results of each model.

F1-score is calculated as the harmonic mean of the Precision and Recall:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Here the Precision shows the number of true positive (TP) outcomes from the entire set of positive model answers (TP + FP):

$$Precision = \frac{TP}{TP + FP}$$

The Recall shows the number of true positive (TP) outcomes from the entire set of positive actual samples (TP + FN):

$$Recall = \frac{TP}{TP + FN}$$

5.1.2 Time of prediction

Another metrics is time of prediction. It is measured by Jupyter Notebook magic %%time. Prediction was made for Test dataset: 926 records.

5.2 Experiment Setup

5.2.1 Logistic Regression

For **TfidfVectorizer** I have set **ngram_range** parameter to (1,2) which means using unigrams and bigrams. All words were lowcased on pre-processing stage. **max_features** parameter was set to 10'000. Stop words were not excluded. The received TF-IDF embeddings are between [0,1].

LogisticRegression was used with default settings.

Training and prediction code was run in Google Colab environment with CPU.

5.2.2 Transformer

Params:

1. Loss: CrossEntropyLoss().
2. Optimizer: AdamW() (learning rate=2e-5, correct_bias=False).
3. Sheduler: get_linear_schedule_with_warmup().
4. Number epoch: 10.
5. Max length: 512.

Training and prediction code was run in Google Colab environment with GPU.

5.2.3 GPT

Params:

1. top_p: 0.5.
2. temperature: 0.3.
3. max_new_tokens: 100.
4. do_sample: True.

Training and prediction code was run in Google Colab environment with GPU.

6 Results

6.1 F1-score

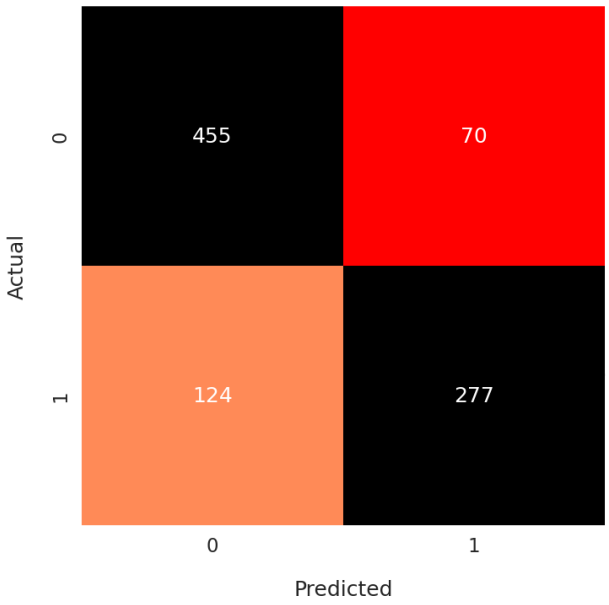

Figure 7: Confusion matrix for Logistic Regression

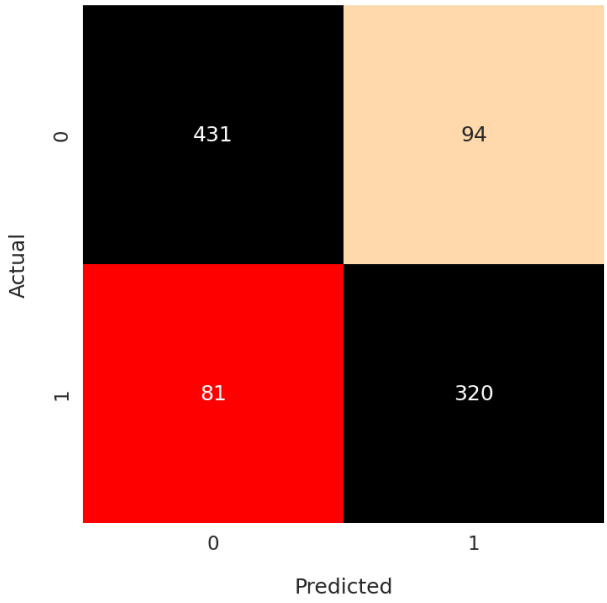

Figure 8: Confusion matrix for rubert-tiny2.

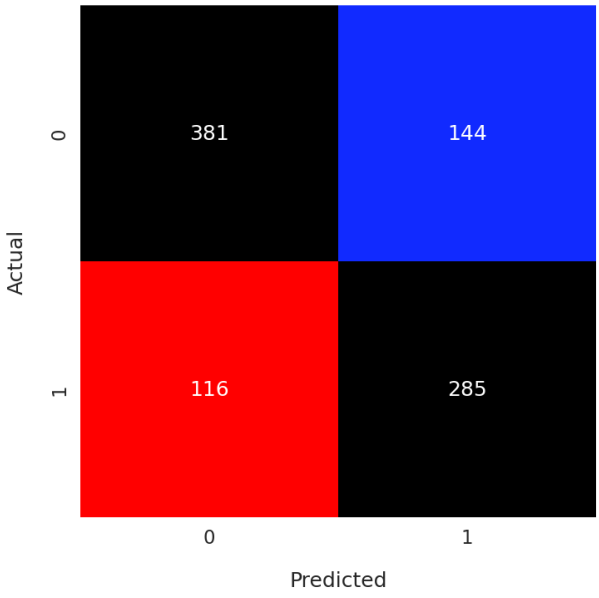

Figure 9: Confusion matrix for GPT (Saiga/Mistral) model.

Best performance in terms of F1-score was shown by rubert-tiny2 model.

Model	F1-score
Log-Reg	0.78
Transformer (rubert-tiny2)	**0.82**
GPT (Saiga/Mistral)	0.72

Table 1: F1-scores for tested models.

6.2 Time of prediction

Best performance in terms of Time of prediction was shown by Logistic Regression model.

Model	Time of prediction
Log-Reg	**191 ms**
Transformer (rubert-tiny2)	5.04 s
GPT (Saiga/Mistral)	2h 16min 34s

Table 2: Time of prediction for tested models.

7 Conclusion

I tested three models: Logistic Regression, rubert-tiny2 and Saiga/Mistral.

Logistic Regression shown good performance on my dataset and it is already used for my project since beginning of December.

rubert-tiny2 shown even batter perfomnce and I'l going to switch my project to this model.

Saiga/Mistral was very slow in my experiment. I'm going to implement cash in the architecture of this solution. And check if Time of prediction will become better.

References

[Habernal et al., 2016] Habernal, I., Zayed, O., and Gurevych, I. (2016). C4corpus: Multilingual web-size corpus with free license. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 914–922.

[Kowsari et al., 2019] Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L. E., and Brown, D. E. (2019). Text classification algorithms: A survey. *Information*, 10(4).

[Levenshtein, 1966] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10:709.

[Merity et al., 2017] Merity, S., Xiong, C., Bradbury, J., and Socher, R. (2017). Pointer sentinel mixture models. In *Proceedings of International Conference of Learning Representation*.

[Pathak, 2022] Pathak, C. (2022). News article classification task using sota models and their comparison.

[YULIANTO, 2022] YULIANTO, Y. (2022). News classification with tf-idf machine learning.

[Левенштейн, 1966] Левенштейн, В. И. (1966). Двоичные коды с исправлением выпадений, вставок и замещений символов. *Доклады Академий Наук СССР*, 163(4):845–848.