# ODS NLP Course Report

Oxana Kulguskina

May 13, 2024

**Abstract**

This report covers the development of the final project for the ODS NLP Course. The project consists of two separate LSTM models trained on a Russian news dataset and wrapped in fastApi web applications. It is a pet project with no purpose but educational and entertaining and probably no use to anyone but its owner who learned a lot in the process. The project repository is accessible at `https://github.com/dalyeth/NewsParsing-Classification-SeqGeneration`

## 1 Introduction

This project is an educational pet-project aimed solely at reinforcing a number of skills of its owner: web parsing, use of RNNs for various NLP tasks, building simple web-applications with fastApi. The choice of tools, tasks and methods was dictated for the most part by personal preferences and educational track of the project owner. While all personal goals were successfully achieved, the project itself has either no scientific nor research nor novelty value. What (probably) somehow distinguishes it from a bunch of other classification/generation students' pet projects is the use of pretrained BERT sentence embeddings and tokenizer with LSTM and a combination of top-k sampling and beam search for text generation (which results in surprisingly good fake news titles).

### 1.1 Team

**Oxana Kulguskina** in the owner of the project and prepared this report.

## 2 Related Work

#### 2.0.1 Deep Learning for Text Classification:

Liu et al. (2016), Zhou et al. (2016), Minaee et al. (2021) et al

#### 2.0.2 LSTM for Text Generation:

Mangal et al. (2019) et al

# 3 Dataset

The project dataset is a self-made sandbox set of short news articles parsed by the project owner from lenta.ru, iz.ru, ria.ru with BeautifulSoup tools.

The parser code for each data source is provided in the project repository.

Each article was parsed with the following metadata: docid, url, title, text. Archives with parsed articles by source are accessible at `https://drive.google.com/drive/folders/1X3zd3zjTv6-Bho3z3FpCAoByM0N2PNro?usp=sharing`

The following 9 news categories were parsed: Health, Incidents, Ex USSR, Science, Sport, Travel/Tourism, Society, Economy, Realty/Construction.

Articles from different sources were combined and checked for duplicates. A new synthetic feature 'titletext' was generated by concatenating 'title' and 'text' fields and used for the classification task.

## 3.1 Classification

For the classification task the data was sampled to provide a uniform distribution of all 9 categories in the dataset. The proportion of classes Society and Incidents in the data was slightly increased to help the model tackle vague classes.

The resulting dataset consisted of 146616 news articles. Of them Society: 36654, Incidents: 24436, 12218 per each other class.

## 3.2 Generation

For the generation task all parsed data was used, 479794 news titles total.

Table 1: Distribution of news categories in the dataset

| | |
|---|---|
| Health | 14522 |
| Ex USSR | 70248 |
| Incidents | 53269 |
| Science | 52710 |
| Sport | 65520 |
| Tourism/Travel | 28643 |
| Society | 109656 |
| Economy | 73008 |
| Realty | 12218 |

For both tasks resulting datasets were split into train/test samples with test data size 25%.

The train data was further split into train and validation data with validation data size 10%.

The distribution of classes in test data and validation data reproduces the distribution of classes in train data.

# 4 Metrics

**f1-score, precision, recall, accuracy** for the **classification** task

**perplexity for the generation task**

# 5 Embeddings

## 5.1 Classification

1. Texts were split into sentences with Spacy and truncated/padded to 50 sentences.
2. Each sentence was encoded with rubert-tiny2 sentence embedder.

## 5.2 Generation

Titles were encoded with rubert-tiny2 tokenizer and fed to nn.Embedding layer(83828, 2048)

# 6 Training Models

Both models were intended as one-shot models aimed at reinforcing some coding skills and utilizing particular frameworks and RNN, so no baselines for comparison and no fullscale experiments.

## 6.1 Classification:

A bidirectional LSTM with 4 layers (dropout 0.3 in between layers), mean pooling and tanh activation function, trained for 20 epochs total with CrossEntropyLoss() as loss function. Training was stopped when metrics on validation data began degrading.

Table 2: Class mapping:

| | | | | | |
|---|---|---|---|---|---|
| label | 0 | 1 | 2 | 3 | 4 |
| category | Society | Economy | Incidents | ex USSR | Sport |
| label | 5 | 6 | 7 | 8 | |
| category | Healthcare | Realty | Tourism | Science | |

### 6.1.1 Model results on test data:

The model often misclassifies 0 (Society), 2 (Incidents) and 3 (ex USSR), however it shows a passable in general classification result with f1-score exceeding 0.85 for 6/9 classes and f1-score exceeding 0.9 for specific classes (Sport, Health, Realty, Tourism/Travel). Better quality may probably be achieved by oversampling classes 0, 2 and 3 but as for now we are satisfied with the result.

Figure 1: Classification Report

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.82 | 0.71 | 0.76 | 9164 |
| 1 | 0.84 | 0.88 | 0.86 | 3054 |
| 2 | 0.78 | 0.85 | 0.82 | 6109 |
| 3 | 0.74 | 0.82 | 0.78 | 3055 |
| 4 | 0.97 | 0.95 | 0.96 | 3055 |
| 5 | 0.89 | 0.93 | 0.91 | 3055 |
| 6 | 0.93 | 0.95 | 0.94 | 3054 |
| 7 | 0.86 | 0.94 | 0.90 | 3054 |
| 8 | 0.89 | 0.85 | 0.87 | 3054 |
| accuracy | - | - | 0.84 | 36654 |
| macro avg | 0.86 | 0.87 | 0.87 | 36654 |
| weighted avg | 0.85 | 0.84 | 0.84 | 36654 |

## 6.2 Generation:

### 6.2.1 Model architecture and training

The model was trained for 12 epochs total with CrossEntropyLoss as loss function and Adam as optimizer. Perplexity on validation data and a sample sentence were printed after each epoch. Training was stopped when perplexity rebounced and sample sentences began deteriorating visibly

Best perplexity on validation data was 113, this model showed perplexity 115 on test data.

### 6.2.2 Sampling and sequence generation function

Top-k random sampling and beam search were both tested. While top-k was more creative and fun, many generated sentences were grammatically incorrect or incoherent, prone to sticking or cycling, some had no sense whatsoever.

Sequences generated with a beam search algorithm were shorter and for the most part grammatically correct but also less variable and less creative. Besides, beam search worked much slower.

Figure 2: Classification Confusion Matrix

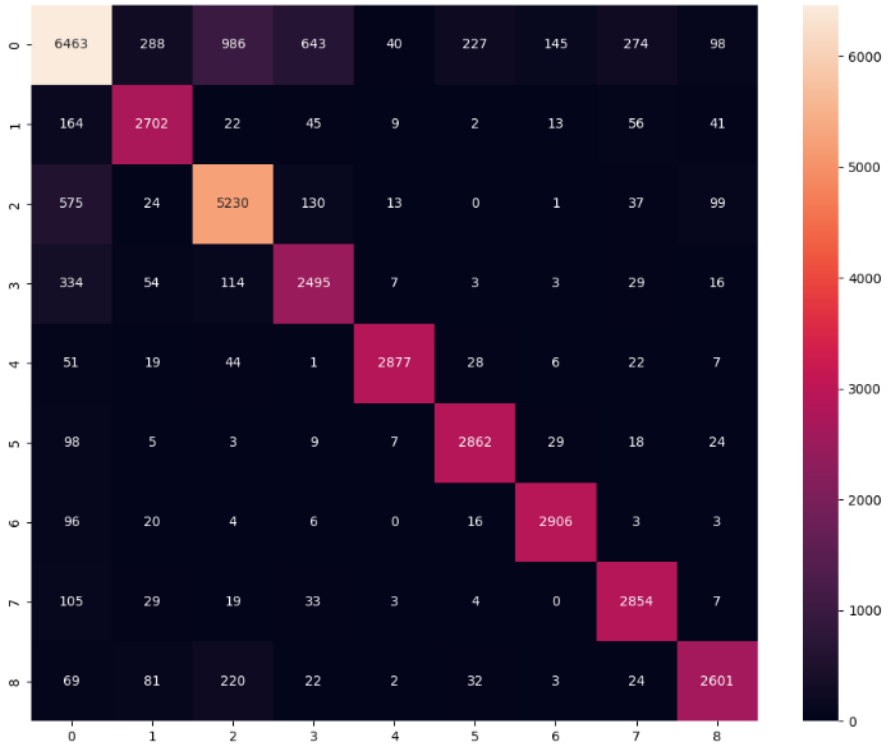


Figure 3: Loss and sample sentence output

```
Epoch: 0, loss: 4.651966475332657, train perplexity: 57.660967679175506, val perplexity: 115.52200821853948

Sample generation: В России отреагировали против « нормандского формата »

Epoch: 1, loss: 4.440952158472476, train perplexity: 51.74927602596425, val perplexity: 112.73279510286896

Sample generation: В России назвали условия возобновления рейсов на курорты

Epoch: 2, loss: 4.389076871832019, train perplexity: 49.006671409793505, val perplexity: 113.29159223385436

Sample generation: В России разработали новый препарат лечения коронавируса и гриппа на COVID
```

So the final generation function combines both algorithms: it uses top-k (top 5) sampling to generate the first tokens of a sequence for better variability and then applies a beam search algorithm to finish it.

Figure 4: Examples of sequences generated with top-k random sampling algorithm:

```
for i in range(10):
    print(generate_sequence(model,'Стали известны'))
```

```
Стали известны личности погибших при пожаре во Внукском отеле в Абхазии детей с начала года на Урале на 2023 году до 2023 г
Стали известны условия возвращения в Европу
Стали известны детали нападения в России в ЕСПЧ на водителя
Стали известны подробности нападения россиян на инкассатор « Свидетели в доме на »
Стали известны личности погибших на территории России в Донбассе военных на фоне обстрела украинскими войсками ВСУ
Стали известны условия существования « АвтоВАЗ - Ареной нефти » после 2020 году
Стали известны детали задержания бывшего начальника Росры областей Украины в Минске и Петербурге за убийство отца в Минске
Стали известны подробности задержания подозреваемого во взяточничестве девочки из « банды » на Украине из России на восток Ро
ссии
Стали известны подробности задержания устроиавших бой в российской колонии
Стали известны личности убитых в российском офисе
```

```
for i in range(10):
    print(generate_sequence(model,'Новые'))
```

```
Новые люди оказались под угрозой
Новые военные уничтожили « Градник на территории » из - под воды
Новые военные уничтожили украинский штурмовики на одном из подразделений ВСУ
Новые российские регионы получат новые учебные корабли на Украину через Керченского пролив
Новые люди в Москве устроили забастовку на месте преступления и устроили погромов на голове в Петербурге
Новые люди предложили сделать все российские города с помощью нейросетей для детей
Новые регионы предложили продлить запрет продажи рыбы
Новые российские города стали чаще жаловаться с коллегами
Новые российские компании оказались под угрозой дефолта
Новые школы оказались под ударом в Казахстане
```

Figure 5: Examples of beam-search generated sequences:

```
В России за сутки госпитализировали 21 заболевших COVID - 19
В России оценили возможность снижения цен на нефть
В России назвали сроки возобновления авиасообщения с Россией
В России предложили ввести налог на тунеядство
В России отреагировали на заявление Кравчука о « зачистке »
В России заявили о готовности к отказу от поездки в Турцию
В России заявили о готовности к отказу от поездки в Европу
В России отреагировали на атаку ВСУ на Работино
В России отреагировали на слова Кравчука о « зачистке »
В России заявили о готовности к отказу от поездки в Турцию
```

# 7 Application

Both models are wrapped in fastApi. The code for FastApi apps can be found in the main project repository in the corresponding app folders. Trained models are accessible at `https://drive.google.com/drive/folders/1XjTdqggBd_r_fWjQSyBOGI4032oQfrhO?usp=sharing` To run the app the project folder must contain a subfolder named 'model' with the corresponding pretrained model file.

Input for the classification app should be a csv file and a name of a text column. The project folder contains a demo csv file (a column name for the demo is 'text' ).

Input for the generation app is a starter sequence string and a number of fake titles to generate.

Figure 6: **Examples of generated titles:**

```
generate_title(model, 'Новые', 10)
```

```
Новые школы оказались под угрозой
Новые регионы Украины предложили переименовать страну в России
Новые регионы предложили продлить кредитные каникулы в России
Новые российские туристы оказались любителями картами
Новые регионы предложили повысить налоги
Новые люди устроили массовую драку на борту упавшего в море самолета
Новые школы научились зарабатывать на коронавирус
Новые военные учения оказались под угрозой
Новые военные учения получат « летающий радар »
Новые регионы в России предложили повысить налоги
```

```
generate_title(model, 'В России', 10)
```

```
В России отреагировали на сообщения об отставке главы МИД Украины
В России оценили возможность дефицита топлива
В России назвали сроки возобновления полетов в Турцию
В России отреагировали на заявление Кравчука о « зачистке » в Донбассе
В России захотели наказывать за продажу алкоголя
В России захотели наказывать за продажу алкоголя из - за коронавируса
В России предложили ввести отдельный налог на бездетность
В России захотели наказывать за нарушение режима прекращения огня
В России захотели упростить продажу алкоголя
В России предложили увеличить расходы на оборону
```

```
generate_title(model, 'В Москве', 10)
```

```
В Москве за сутки госпитализировали 23 пациента с COVID - 19
В Москве задержали участников оппозиционной акции в поддержку Pussy Riot
В Москве за сутки госпитализировали 190 пациентов с коронавирусом
В Москве прошел форум по борьбе с безработицей
В Москве мужчина расстрелял жену и покончил с собой
В Москве прошел форум " Россия "
В Москве за сутки госпитализировали 53 пациента с COVID - 19
В Москве мужчина выстрелил в грудь и попал под следствие
В Москве мужчина захватил заложников из магазина
В Москве мужчина открыл стрельбу по полицейским
```

# 8 Conclusions

The followings steps were successfully completed in the process of development:

1. Data was parsed and collected.
2. Two different datasets for classification and generation tasks were prepared.
3. A LSTM-based model for news classification was implemented, trained and serialized.
4. A LSTM-based model for text generation was implemented, trained and serialized.
5. Different sampling approaches for sequence generation were tested, a custom sampling function was implemented.

6. Applications based on fastApi framework were implemented and deployed.

7. A project report via a LaTex editor was compiled.

# References


Liu, P., Qiu, X., and Huang, X. (2016). Recurrent neural network for text classification with multi-task learning.

Mangal, S., Joshi, P., and Modak, R. (2019). Lstm vs. gru vs. bidirectional rnn for script generation.

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., and Gao, J. (2021). Deep learning based text classification: A comprehensive review.

Zhou, P., Qi, Z., Zheng, S., Xu, J., Bao, H., and Xu, B. (2016). Text classification improved by integrating bidirectional lstm with two-dimensional max pooling.