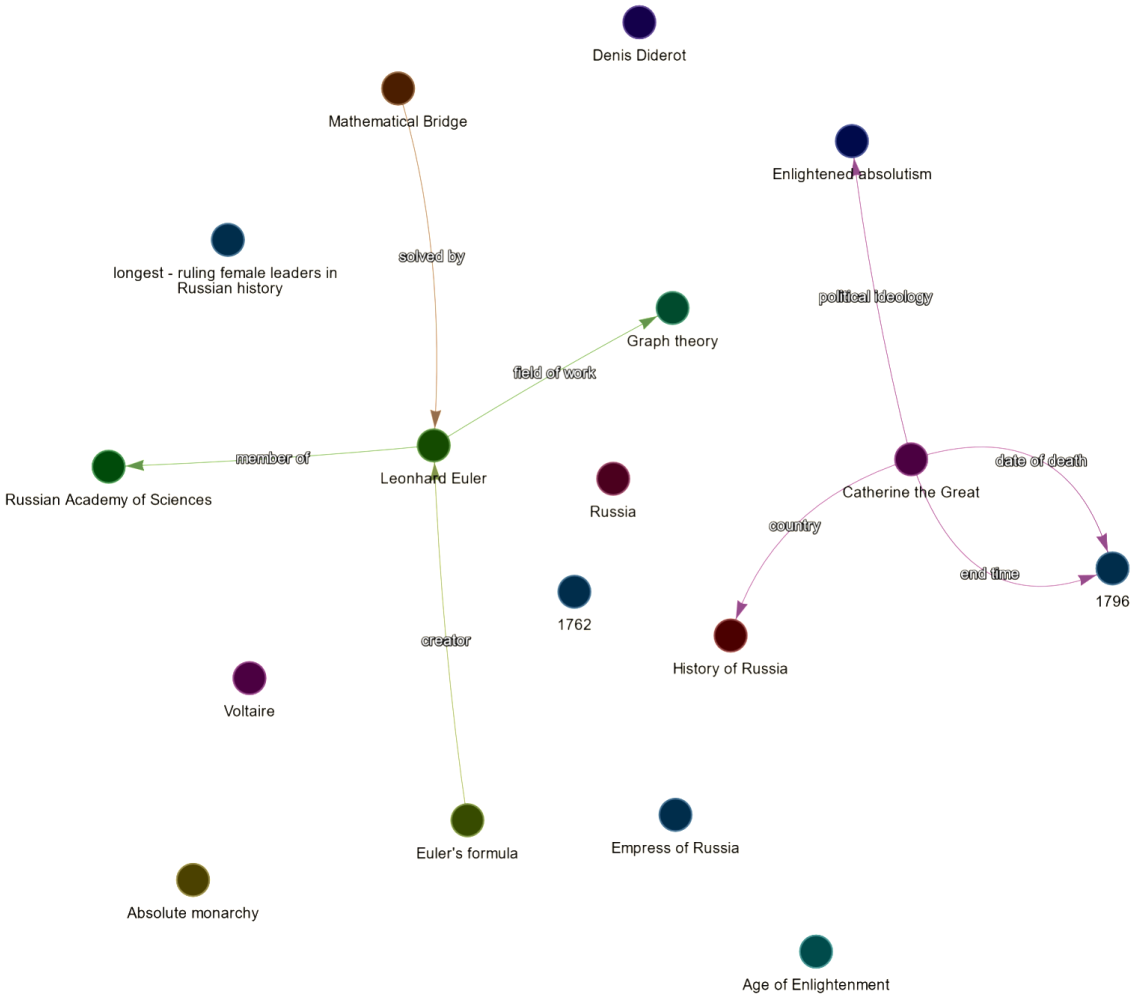
Practical Machine Learning and Deep Learning

Fall Semester 2024-2025



Modern NER, EL & RE

Final Technical Report

Anton Kudryavstev
a.kudryavtsev

Sofya Tkachenko
s.tkachenko

Anatoly Soldatov
a.soldatov

24 november 2024

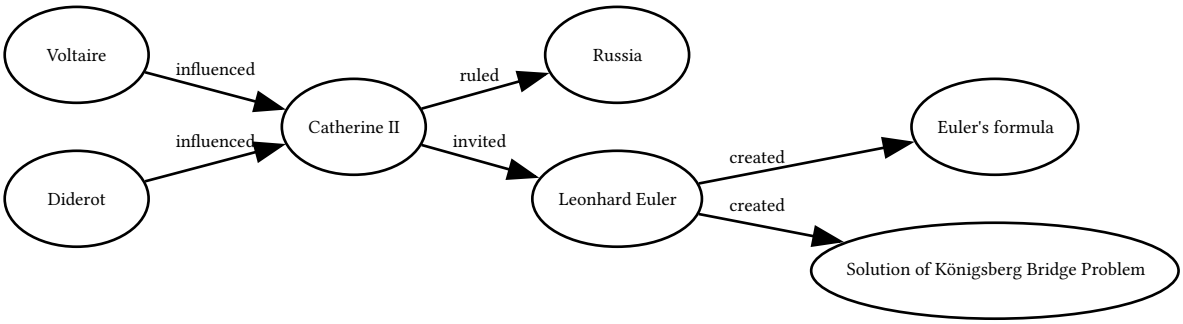
Table of contents

1 - Project topic

We intended to implement a service that would create a knowledge graph based on the text submitted for input and present this information as a graph or a mindmap to the end user. Key technologies for that are named Entity Recognition (NER), Entity Linking (EL), and Relation Extraction (RE).

Example of usage of such service would be:

> *Catherine II, also known as Catherine the Great, was the Empress of Russia from 1762 until her death in 1796. She is considered one of the most renowned and longest-ruling female leaders in Russian history. Catherine was a proponent of Enlightened Absolutism, which advocated for the use of absolute power to bring about reforms inspired by the Enlightenment. She corresponded with notable figures like Voltaire and Diderot. Leonhard Euler was invited to the Imperial Russian Academy of Sciences by Catherine II. Leonhard Euler singnificantly contributed to mathematics and graph theory during his time in Russia. His achivements include Euler's formula for polyhedra and solution of Königsberg Bridge Problem.*



2 - Source code

The source code for our final product, along with notebooks and experimental results, is going to be available on GitHub at this link: PodYapolskiy/entity-extraction. We have also created some modified versions of open-source projects like dartt0n/SPN4RE-NEREL and dartt0n/relik. We are still running experiments and we will publish the final list of work in the main project repo on GitHub soon.

3 - Project timeline

In this section, the we will describe the timeline of this project. Firstly, the general timeline of our project will be presented as a list of various research topics we encountered and the problem formulation. The most significant steps and results will be presented in the next subsections.

Our primary objective was to conduct research on relevant technologies and materials within the field, which included the study of natural language processing methods. These materials were outside the scope of university courses, and we therefore wished to gain a deeper understanding of them.

We are deeply immersed in the study of topics such as

1. Stemming, Lemmatization, Tokenization, Ngrams
2. Bag-of-Words (BoW), SpaCy [1], NLTK [2], Statistical Text Processing
3. Word2vec (SkipGram and Continuos BoW) [3]
4. Recurrent Neural Network (RNN) [4], Long short-term memory (LSTM) [5], Gated Recurrent Unit (GRU)
5. Named Entity Recogntion
6. Attention, Self-Attention [6]

7. Bidirectional encoder representations from transformers (BERT) [7], Highly Dimensional Text Representations
8. BERT Question-Answering, BERT optimizations and finetunes (RoBERTa, DistilBERT, RuBERT, RuBERTa) [8]
9. Entity Linking (EL) and Named Entity Disambiguation (NED)
10. Approximate k-Nearest Neighbors (AKNN), R-Tree, KD-Tree, Annoy, IVFFlat
11. Large Language Models (LLMs) and their applications for sentence similarity and sequence-to-sequence transforms.
12. Relation Extraction (RE), Dense Passage Retrieval (DPR) [9], and reader-retriever architecture for RE.

By studying these topics, we learned almost all the necessary information to create state-of-the-art (SotA) models for the joint task for EL and RE, however more time is needed to be invested to create a new model and train it on a large corpus of texts.

Task	**Assignee**	**Time**
Try nltk + spacy approaches to NER	All	01.10 - 03.10
Exploring NER using BERT	Anatoly	03.10 - 06.10
Exploring NER using LSTM	Sofya	04.10 - 06.10
Research on EL and RE	Anton	06.10 - 27.10
Research on GNN RE capabilites	Anton	21.10 - 22.10
ReLiK fork	Anton	28.10 - 23.11
Understanding and fixing ReLiK	All	29.10 - 10.11
Learning SPN4RE approach	Anton	14.11 - 20.11
Learning PURE approach	Sofya	15.11 - 21.11
Learning PL-Marker	Anatoly	17.11 - 23.11
Adapting NEREL dataset to models	All	22.11 - 23.11

3.1 - Domain Research

We started by exploring the text processing domain - natural language processing. This is a family of techniques that deals with mathematical modeling of a language for various tasks, such as semantic segmentation, semantic analysis, named entities recognition, text generation, and etc.

One of the challenges we are interested in, and which is highly relevant in the real world, is creating a reliable representation of textual information for further processing. Specifically, one issue we have identified is the hallucination problem associated with generative large language models, and one potential solution is the use of knowledge graphs, such as GraphRAG. This paper has led us to idea of developing a more robust approach, in which each connection between a subject and object is directly supported by a citation from the original source document.

3.2 - Named Entity Recognition

The first step in solving this problem is to find all the entities in the text and assign entity type labels to parts of the text (person, organization, date, geolocation, etc.). This is task of Named Entity Recognition.

Named Entity Recognition is the most popular and “solved” technology of the bunch. We have found that NER could be done using one of the approaches:

- **The Rule-Based Approach**. In this approach, sets of rules are created that determine which sequences of words in the text can be named entities. These rules can be based on regular expressions, patterns, or linguistic features. Examples of libraries: spaCy (with support for custom rules), NLTK. Example of such rules could be “select only nouns” or “find sequence (adj, noun, noun)”.
- **A Machine Learning-Based Approach**. This approach uses machine learning algorithms such as CRF (Conditional Random Fields), LSTM (Long Short-Term Memory) and BERT (Bidirectional Encoder Representations from Transformers) to train the model to recognize named entities. The model is trained on marked-up data, where entities are marked in the text. Examples of libraries: spaCy (with trainable models), Stanford NER, Flair, and all the machine learning frameworks (e.g. PyTorch, Tensorflow)
- **Hybrid Approach**. This approach combines rules and machine learning to improve NER accuracy. You can first apply rules to highlight entities, and then run the text through a machine learning model to refine the results.

Machine learning-based solutions have shown significantly better results, so we have decided to focus on them. We have implemented bi-directional LSTM models on TensorFlow and PyTorch, as well as a BERT fine-tune to solve NER problem:

3.2.1 - TensorFlow model

The TensorFlow model consisted of an Embedding Layer, Dropout Layer, two Bidirectional LSTM layers, and a Time Distributed Layer. The dataset used for training and testing is “NER data” from Kaggle [10]. The accuracy was 91.12%. Nonetheless, when we updated TensorFlow to a new version, accuracy dropped below 60%. We assume that there was a bug in TensorFlow that influenced training or evaluation. For that reason, next week we switched to PyTorch.

3.2.2 - PyTorch model

After some experimentation, we have also decided to simplify the overall architecture, going from this in Tensorflow version:

```
vector_size = 128

i = Input(shape=(max_length,))
x = Embedding(input_dim=V+1, output_dim=vector_size, mask_zero=True)(i)
x = Dropout(0.2)(x)
x = Bidirectional(LSTM(256, return_sequences=True, recurrent_dropout=0.2))(x)
x = Bidirectional(LSTM(128, return_sequences=True, recurrent_dropout=0.2))(x)
x = TimeDistributed(Dense(K, activation='softmax'))(x)
```

Listing 1 - Tensorflow Implementation of Bi-LSTM

To the following architecture in the PyTorch version:

```
embed_dim = 128

embedding = nn.Embedding(input_dim, embed_dim, padding_idx=pad_idx)
dropout = nn.Dropout(dropout)
lstm    =    nn.LSTM(embed_dim,    hidden_dim,    num_layers=1,    bidirectional=True,
dropout=dropout)
linear = nn.Linear(hidden_dim * 2, output_dim)
```

Listing 2 - PyTorch Implementation of Bi-LSTM

And with the changes in framework, architecture, and training techniques, **we achieved validation accuracy of 97.8%**.

Here are the results of the PyTorch implementation:

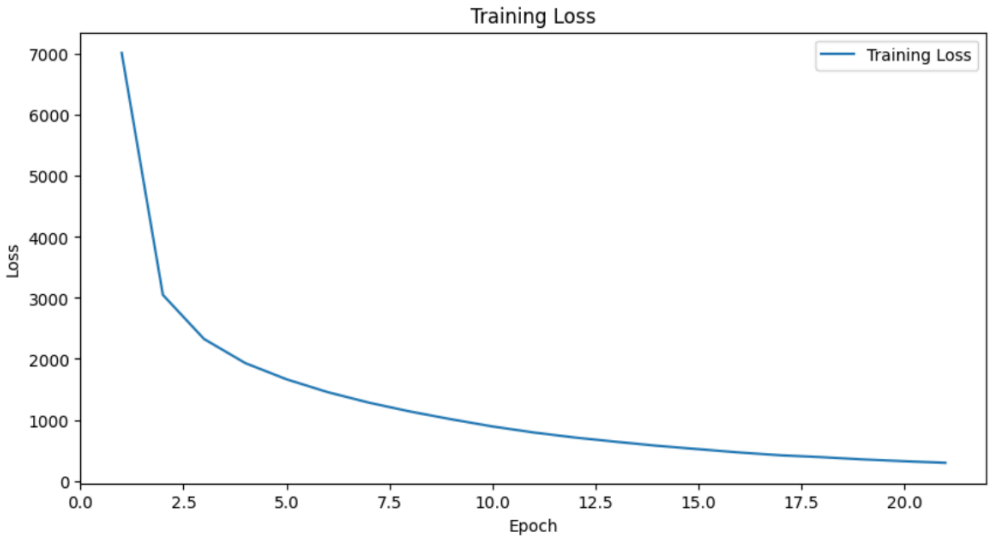


Figure 1 - LSTM train loss

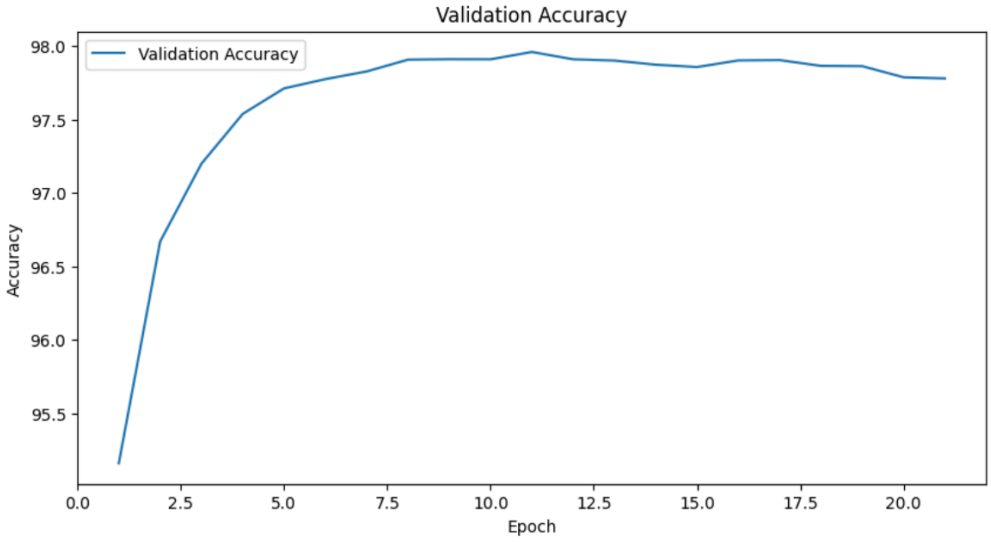


Figure 2 - LSTM validation accuracy

3.2.3 - BERT

While experimenting, **ClearML** was used as tool for tracking metrics, logs and different setups. During the experiments, two approaches were used. First of all, we tried vanilla DistilBERT model [11] on the “wnut_17” dataset [12] consisting of NER tasks. The results appeared to be promising even for no fine-tuning **90%** accuracy. After that we fine-tuned model via LoRA on the train part of mentioned data and we gain **6%** improvement for accuracy reaching **96%**.

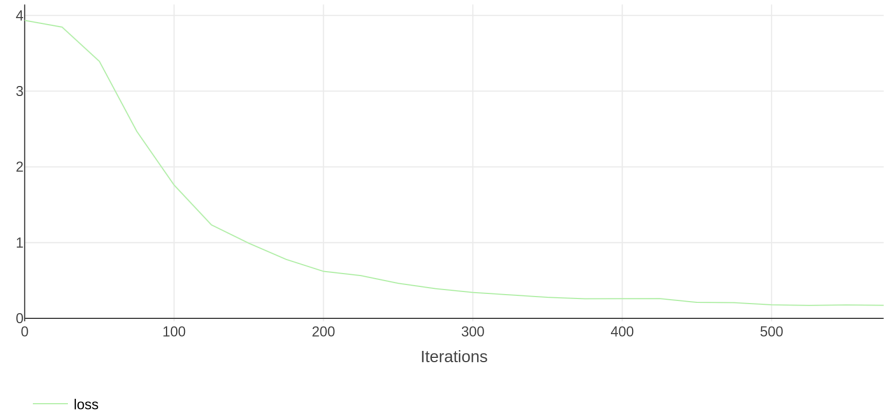


Figure 3 - BERT evaluation loss

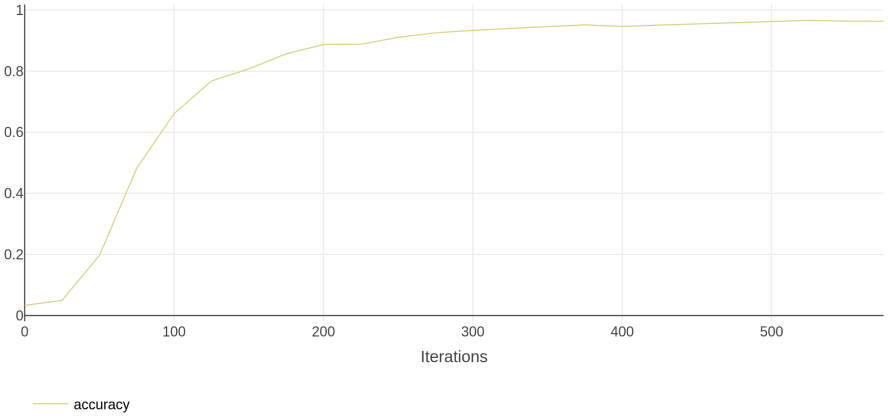


Figure 4 - BERT evaluation accuracy

3.2.4 - NER conclusion

As you can see from the results presented above, a simple LSTM can achieve satisfactory results on a big NER corpus. Especially if the NER corpus used in training is of simular nature to the testing information. Therefore we decided to deepen our research into significantly more complicated techniques: Entity Linking and Relation Extraction.

3.3 - Entity Linking and Relation Extraction

We are deeply immersed in the topic of Entity Linking and Relationship Extraction. The problem of Entity Linking is formulated as a task to link entity e to existing document d describing this entity e. Example of such linking could be a link of entity Catherina II to the Wikipedia article “Catherine the Great”. Relation extraction task, as previously stated, is a task to generate triplets $< s, r, e >$ or to find a correct relation r between subject s and object o. For example: $< \mathrm{Catherina\ II}, \mathrm{PROFESSION}, \mathrm{Empress} >$. We have explored a variety of approaches, and in the following section, we will discuss some of the most significant ones.

3.3.1 - Bert-based

The paper “Joint Entity and Relation Extraction with Set Prediction Networks” [13] proposes the set prediction networks (SPN) model, which uses transformers with non-autoregressive decoding to generate multiple trees at a time. The model consists of the following parts:

- Sentence Encoder: Uses the BERT model to represent each token in a sentence with a context-sensitive representation.
- Non-autoregressive Decoder: Generates multiple triplets simultaneously using self-learning mechanisms.
- Set-Based Loss Function: Used to train a model based on binary labels that take into account the correspondence between predicted and true triplets.

Firstly, the model demonstrates a significant increase in performance and accuracy compared to standard BERT-based models, which try to predict relationships in the order in which they appear in the text. Secondly, the proposed model can effectively handle sentences that contain different types of overlap and intersection of entity relationships. However, it requires substantial computing power for training and testing. The quality of the training dataset and pretrained models significantly affects the performance of the proposed model.

3.3.2 - DPR

The paper, “Dense Passage Retrieval for Open-Domain Question Answering” [9] describes the DPR method for open-domain search in the field of question processing (Question Answering). The DPR method uses a dual encoder to index and search for texts that contain answers to questions. The authors demonstrate how a trained DPR outperforms traditional methods such as TF-IDF or BM25 in search tasks. However, they also note some limitations, including the difficulty of learning the model and the high latency in the search. The DPR model can be useful for EL and RE tasks as it is allows more accurately determine the contexts in which the desired entities or relationships may be located. In particular, DPR can help improve the accuracy of identification of entities and relationships between them, as it takes into account the semantic proximity of the text, and not just lexical coincidences. These advantages are heavily used in the following works.

3.3.3 - EntQA

The paper “EntQA: Entity Linking as Question Answering” [14] proposed method is to solve the EL problem as Question-Answering (QA) problem. It treats documents as questions, and the answers are links to entities from the database.

To solve the problem, the EntQA model is proposed, which consists of two modules:

1. a retriever module for searching for entities.
2. a reader module for reading text.

The first part finds entities that can be mentioned in the document, and the second part determines the mentions of these entities in the text. Then the model performs QA to link entities.

The model uses pre-trained BERT and ELECTRA models to represent documents and entities. It has been shown that EntQA surpasses modern methods in many metrics. However, there are several limitations. First, EntQA requires a large amount of memory to store and index dense attachments, which can be difficult to implement in practice. Secondly, EntQL uses a static threshold to filter responses, which is a hyperparameter that requires configuration. Thirdly, the model requires special pre-training methods, such as pre-training on weak labels.

3.3.4 - ReLiK

The paper, “Relik: Retrieve and Link, Fast and Accurate Entity Linking and Relation Extraction on an Academic Budget”, [15], is the most similar to our project and addresses the same goal. The core idea

of the proposed solution is to improve the methodology proposed in [14]. The authors implemented a model that solves both Entity Linking and Relational Extraction tasks in a single pass. This model uses a bi-encoder architecture with Reader and Retrieval encoders, respectively. The model retrieves candidate entities and relations, reads and contexts the text and candidates, then links and extracts the entities and triples, providing a final set of extracted relationships and entities.

3.3.5 - REBEL

The paper, “REBEL: Relation Extraction by End-to-End Language Generation” [16] proposes another solution. The authors use a sequence-to-sequence model that generates triplets (e_i, e_j, r), identifying relations in the text. By using this technique, the authors managed to achieve state-of-the-art results in relation extraction tasks while extracting over 200 different relation types. However, this model is based on BART-Large, which has a significant impact on the computational requirements for training and evaluation.

The paper “Relation Extraction By End-to-end Language generation” [16] introduces a method called REBEL for extracting relations from raw text. The main idea behind REBEL is to frame the task of relation extraction as a sequence-to-sequence transform task, where a text sequence is translated into a sequence of tokens representing the relational facts. The method uses an autoregressive model based on BART to generate the target sequence of relational facts given the source text.

Such model achieves state-of-the-art performance on various RE benchmarks, with improvements ranging from 1.2 to 6.7 F1 score. The pre-trained REBEL performs particularly well on document-level RE tasks (finding relations with-in one or more document). The results show the flexibility of the proposed method, as it can be applied to a wide variety of datasets and tasks.

However, REBEL requires pre-training on a large silver dataset, which might limit its applicability and performance significantly degrades for smaller datasets or those with many entity types. Moreover, the complexity of the model means it might be difficult to deploy in resource-constrained environments.

Another, advantage of REBEL model, is that it could be adjusted to multilingual model as proposed in “REDFM: a Filtered and Multilingual Relation Extraction Dataset” [17] paper. However, we have not yet managed to test this model.

3.4 - Fine-tuning on Russian language

All of the EL and RE models we mentioned above **do not work with the Russian language**. Moreover, according to our research, there are no modern solution for joint EL and RE task. We decided to fix that.

The first step to fixing it is finding an applicable dataset. We decided to use NEREL [18], because it is:

1. Fresh. The latest version of this dataset was published in September of 2023, which is important for relevant entities;
2. Accurate. It was published by a reputable AI research institute after several iterations;
3. Generalizable. It includes almost all features needed by the models listed above, with rare exceptions.

3.4.1 - ReLiK [15]

The current performance of ReLik on Russian language remains objectively unusable. We tried to fine-tune it but we faced many problems: not all the components are presented in repository, repository itself is buggy and is configured badly, that’s why we forked it. Also, we have adapted NEREL for fine-tuning but faced with insufficient components presented on the ReLik crepository.

3.4.2 - SPN4RE [13]

The most promising adaptation that we covered. But even after debugging and rewriting a significant part of the repository (including all of data loading), we could not adapt it for Russian embeddings due to high dimensionality of multi-lingual embeddings.

3.4.3 - PURE [19]

The oldest repository on this list. Authors used Python 3.6 and libraries that are now oficialy deleted from python. Despite having most of the stars on GitHub and being successefully adapted to NEREL, we failed to teach it due to the code being severely outdated. And we did not fail without trying, we spent over six hours on version matching.

3.4.4 - PL-Marker [20]

One of the oldest approaches presented here. We faced problems with converting NEREL data format to suit the fine-tuning scripts. Additionally, facing bad code practices in the repository.

In conclusion, what is left for us to do now is to chose (and rewrite) the lesser evil.

4 - Web Interface and Deployment

One part of our solution is Named Entity Recognition. We have developed an interface and a full deployment pipeline for this part:

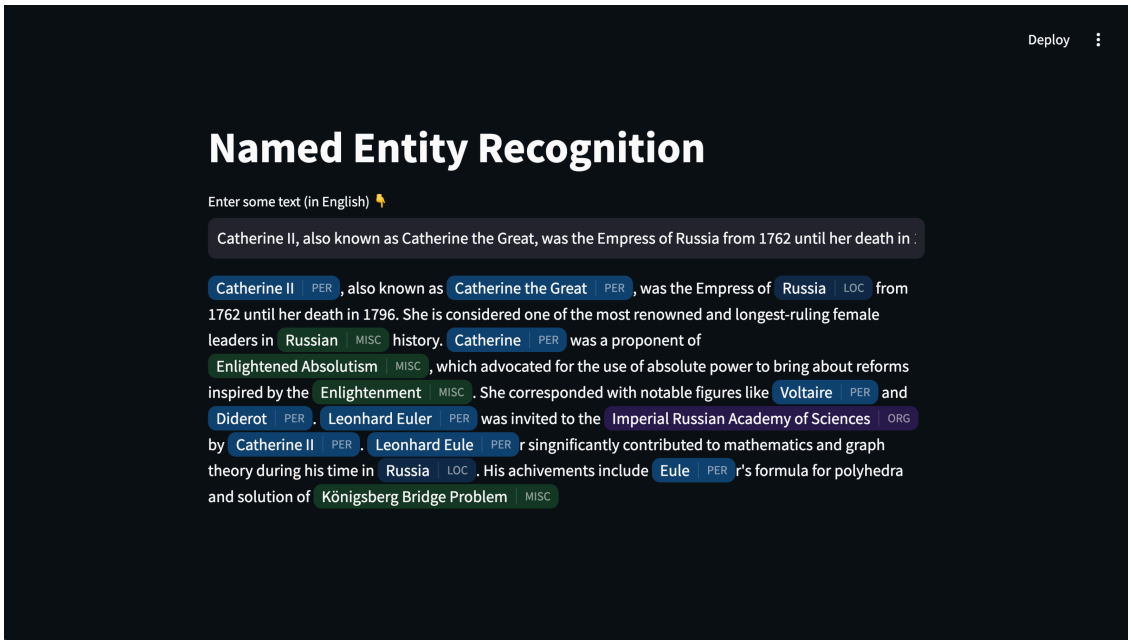


Figure 5 - NER interface

4.1 - Backend

The backend is built using FastAPI and runs the fine-tuned `distil-bert` model for NER. It includes:

- **Model Caching**: To improve performance by caching the model.
- **Optimized Docker Configuration**: To ensure efficient resource usage.
- **Health Checks**: To monitor the status of the API.

4.2 - Frontend

The frontend is built using Streamlit and includes:

- **Input Fields**: For users to input text.
- **Prediction Area**: To display the annotated entities with colorful labels.

4.3 - Technologies Used

- **Docker**: For containerization.
- **FastAPI**: For building the backend API.

- **Streamlit**: For building the frontend web application.
- **HuggingFace and PyTorch**: For the `distil-bert` model.
- **st-annotated-text**: For color annotation in the frontend.
- **Poetry**: For dependency management.
- **ClearML**: For tracking experiments

Nonetheless, for entity linking and relationship extraction we will need an interface, that is created specifically for the chosen model. We still need the time until presentation to finalize model selection and visualization. But the interface and deployment above lay a sturdy foundation for us to build upon.

Figure 6 - Example of joint EL and RE solution using ReLiK-cIE-Large

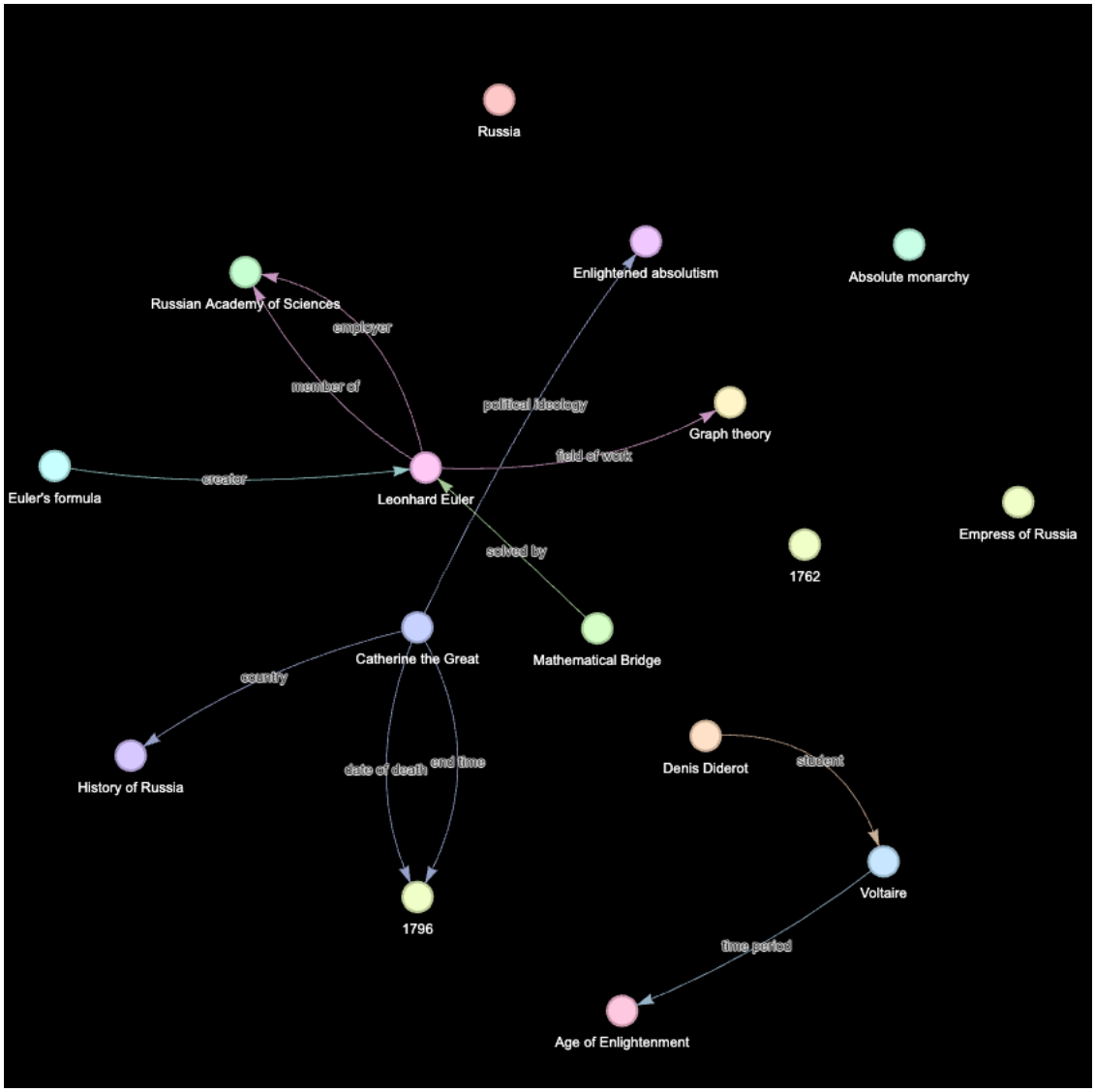


Figure 7 - Example of joint EL and RE solution using ReLiK-cIE-Large

5 - Results

Our research and experiments demonstrate that current approaches for Named Entity Recognition (NER) have reached impressive levels of performance, even in multilingual settings. However, Relation Extraction and Entity Linking together remain challenging, with notable gaps in accuracy and efficiency, even for English — the most studied language in this domain. These challenges are amplified in less-resourced languages, where the scarcity of annotated datasets, linguistic variability, and domain-specific complexities worsen model performance. For instance, while NER models benefit from advancements in contextual embeddings and transformer-based architectures, Relation Extraction often struggles with capturing interdependencies, and Entity Linking faces difficulties in resolving ambiguities, particularly in multilingual, low-resource scenarios and novel terms scenarios. These findings highlight the need for innovative approaches tailored to address these limitations, especially for Russian language.

5.1 - Artifacts

- Repositories: Main, ReLiK fork, SPN4NE fork
- Notebooks: BERT, LSTM
- Deploy: ReLiK
- Datasets: NEREL, wnut_17
- Models: ReLiK, Distilbert, LSTM-based

- Articles: ReLiK, SPN4NE, PURE, PL-Marker

6 - Contribution

- Anton - Teamlead, main researcher, bug fixer
- Sofya - Tensorflow & Pytorch dev, LSTM guru, NEREL dataset adapter
- Anatoly - Pytorch & BERT enjoyer, ReLiK tester

7 - Future Work

In future work, we aim to enhance our current solutions in modern Named Entity Recognition (NER), Entity Linking (EL), and Relation Extraction (RE). We look forward on possible applications and learning capabilities of mrebel dataset [21]. The key area of emphasis will be the development robust approaches to deal with russian language on closed Information Extraction (cIE) task, which is EL + RE. That challenge remains largely unexplored in the existing literature. We are loogking forward on opportunity to use brand new approach of ReLiK on the most diverse NER dataset on russian language - NEREL [18]. By doing so, we hope to contribute significant advancements to the field and enable practical applications in specialized domains requiring closed IE solutions.

References

[1] M. Honnibal and I. Montani, “spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing,” 2017.

[2] E. Loper and S. Bird, “NLTK: The Natural Language Toolkit.” [Online]. Available: https://arxiv.org/abs/cs/0205028

[3] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality.” [Online]. Available: https://arxiv.org/abs/1310.4546

[4] R. M. Schmidt, “Recurrent Neural Networks (RNNs): A gentle Introduction and Overview.” [Online]. Available: https://arxiv.org/abs/1912.05911

[5] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] A. Vaswani *et al.*, “Attention Is All You Need.” [Online]. Available: https://arxiv.org/abs/1706.03762

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” [Online]. Available: https://arxiv.org/abs/1810.04805

[8] Y. Liu *et al.*, “RoBERTa: A Robustly Optimized BERT Pretraining Approach.” [Online]. Available: https://arxiv.org/abs/1907.11692

[9] V. Karpukhin *et al.*, “Dense Passage Retrieval for Open-Domain Question Answering.” [Online]. Available: https://arxiv.org/abs/2004.04906

[10] R. Patel, “NER Data.” [Online]. Available: https://www.kaggle.com/datasets/rajnathpatel/ner-data

[11] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter,” *ArXiv*, 2019.

[12] L. Derczynski, E. Nichols, M. van Erp, and N. Limsopatham, “Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition,” in *Proceedings of the 3rd Workshop on Noisy User-generated Text*, Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 140–147. doi: 10.18653/v1/W17-4418.

[13] D. Sui, Y. Chen, K. Liu, J. Zhao, X. Zeng, and S. Liu, “Joint Entity and Relation Extraction with Set Prediction Networks,” *CoRR*, 2020, [Online]. Available: https://arxiv.org/abs/2011.01675

[14] W. Zhang, W. Hua, and K. Stratos, “EntQA: Entity Linking as Question Answering.” [Online]. Available: https://arxiv.org/abs/2110.02369

[15] R. Orlando, P.-L. H. Cabot, E. Barba, and R. Navigli, “ReLiK: Retrieve and LinK, Fast and Accurate Entity Linking and Relation Extraction on an Academic Budget.” [Online]. Available: https://arxiv.org/abs/2408.00103

[16] P.-L. Huguet Cabot and R. Navigli, “REBEL: Relation Extraction By End-to-end Language generation,” in *Findings of the Association for Computational Linguistics: EMNLP 2021*, M.-F. Moens, X. Huang, L. Specia, and S. W.-t. Yih, Eds., Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 2370–2381. doi: 10.18653/v1/2021.findings-emnlp.204.

[17] P.-L. H. Cabot, S. Tedeschi, A.-C. N. Ngomo, and R. Navigli, “RED^{\rm FM}: a Filtered and Multilingual Relation Extraction Dataset”. [Online]. Available: https://arxiv.org/abs/2306.09802

[18] N. Loukachevitch *et al.*, “NEREL: a Russian information extraction dataset with rich annotation for nested entities, relations, and wikidata entity links,” *Language Resources and Evaluation*, pp. 1–37, 2023.

[19] Z. Zhong and D. Chen, “A Frustratingly Easy Approach for Joint Entity and Relation Extraction,” *CoRR*, 2020, [Online]. Available: https://arxiv.org/abs/2010.12812

[20] D. Ye, Y. Lin, and M. Sun, “Pack Together: Entity and Relation Extraction with Levitated Marker,” *CoRR*, 2021, [Online]. Available: https://arxiv.org/abs/2109.06067

[21] P.-L. Huguet Cabot, S. Tedeschi, A.-C. Ngonga Ngomo, and R. Navigli, “RED^{\rm FM}: a Filtered and Multilingual Relation Extraction Dataset”, in *Proc. of the 61st Annual Meeting of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada: Association for Computational Linguistics, Jul. 2023. [Online]. Available: https://arxiv.org/abs/2306.09802