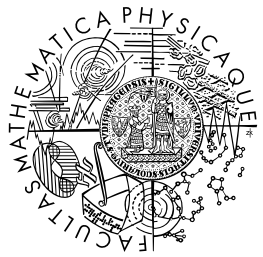

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Hynek Kydlíček

Implicit Information Extraction from News Stories

Institute of Formal and Applied Linguistics

Supervisor of the bachelor thesis: Mgr. Jindřich Libovický, Ph.D.

Study programme: Computer Science

Study branch: Artificial Intelligence Bc.

Prague 2023

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

Firstly, I would like to thank my supervisor Mgr. Jindřich Libovický, Ph.D. for his guidance and support throughout the whole process. Secondly, I am grateful to the faculty for providing me with computational resources. Last but not least, I would like to extend my thanks to my family, girlfriend, and friends for their support and encouragement.

Title: Implicit Information Extraction from News Stories

Author: Hynek Kydlíček

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. Jindřich Libovický, Ph.D., Institute of Formal and Applied Linguistics

Abstract: This work deals with information extraction from Czech News Stories. We focus on four tasks: Publishing server, Article category, Author's textual gender and Publication day of week. Due to the absence of a suitable dataset for the tasks, we present CZEch NEws Classification dataset (CZE-NEC), one of the most extensive Czech classification datasets, composed of news articles from various sources, spanning over twenty years. Tasks are solved using Logistic Regression and pre-trained Transformer encoders. Emphasis is put on fine-tuning methods of the Transformer models, which are evaluated in detail. The models are compared to human evaluators, revealing significant superiority over humans on all tasks. Furthermore, the models are pitted against the commercial large language model GPT-3, outperforming it on half of the tasks, despite GPT-3 being significantly larger. Our work sets strong baseline results on CZE-NEC allowing for further research in the field.

Keywords: News, Information Extraction, Czech News Classification Dataset, NLP, BERT

Contents

Introduction

Every day, there are millions of articles published on the internet. Various authors write those with different backgrounds and opinions. Their goal is, however, the same; to convey the information to the reader. While the information is usually explicit, there are some cases when it is implicit. Such implicit information is usually intentionally used to shape the reader's opinion. However, it's also possible that the author inserts such implicit information unintentionally. It could be the result of the author's background or state of the world at the time of writing. The author's texts might be more pessimistic at the start of the week, while more optimistic before the weekend. It might also be possible that senior authors write articles slightly differently than their colleagues. We thus ask ourselves the following question:

> *Which and how much implicit information can be extracted from the news articles?*

As there could be an infinite amount of such fingerprints, we narrow the scope of our research to the following tasks:

1. Article publishing server
2. Article category
3. Author's textual gender
4. Publication day of week

Furthermore, we limit the scope of the research to the Czech language.

For simplicity, we will further refer to *Article publishing server* as **Server**, to *Article category* as **Category**, to *Author's textual gender* as **Gender**[1], and to *Publication day of week* as **Day Of Week**

[1]Note that we by no means are referring to the author's actual gender

Related Work

With the rise of Natural Language Processing (NLP) and Machine Learning (ML), there has been a lot of research on implicit information in textual content, most notably on sentiment analysis and text classification. Most notably, Joulin et al. [1] showed that a simple Bag of Words approach with a few hidden layers could achieve State of The Art (SoTA) results while being very fast. Zhang and LeCun [2], inspired by the usage of Convolutional Neural Networks (CNNs) in image classification, proposed a similar approach for text classification with excellent results.

In recent years, SoTA results have been achieved by the transformer models [3]. Such an architecture was used by BERT [4], where the authors achieved SoTA results on all the tasks of GLUE benchmark [5], which among others, includes sentiment analysis task.

To cope with BERT English mono-lingual training, Straka et al. [6], followed by Lehečka and Švec [7], proposed Czech versions of BERT, called RobeCzech and Fernet-News, respectively.

Our approach

We evaluate the defined tasks on CZEch NEws Classification dataset (CZE-NEC), the large dataset of Czech news articles of over 1.6 million articles, described in chapter 2. To assess human ability on the tasks, we take a subset of the dataset and ask humans to perform the tasks (section 3.1).

We continue by training a Multinomial Logistic Regression model with Bag of Words features to establish a keyword extraction performance (section 3.2). Next, we fine-tune Deep Learning (DL) models, specifically the RobeCzech [6] and Fernet-News [7] models, using various approaches to enhance their performance (section 3.4).

Lastly, we explore the multi-lingual capabilities of the commercial Large Language Model GPT-3 [8] by fine-tuning it on CZE-NEC (subsection 3.3.2).

Acknowledgements

The Github Copilot[2] was used when writing all source codes including training, data analysis and crawlers. Grammarly[3] was used to check the grammar and spelling of the thesis.

[2] `https://github.com/features/copilot`

[3] `https://app.grammarly.com/`

Chapter 1

Text classification

Information extraction from the text can be stated as a text classification task. Given a text, we want to assign a class it belongs to. The class can be anything, from the text's category to the text's sentiment or the author's age. Depending on the number of classes, we can distinguish between **binary** (2) and **multiclass** (2+) classification. We might want to classify the text by multiple labels; we thus also distinguish between **single-label** and **multi-label** classification.

Therefore, we can rephrase our research question as a **single-label multiclass text classification of Czech News articles**. To avoid ambiguity, we will refer to a single news article text as a **document** going forward.

1.1 Evaluation Metrics

To assess performance, we need to define some metrics. We start by defining metrics for binary classification and then extend them to multiclass classification.

In Binary classification, we have two classes; positive and negative. Thus we can have four possible outcomes:

1. True Positive (TP) – prediction positive, label positive
2. True Negative (TN) – prediction negative, label negative
3. False Negative (FN) – prediction negative, label positive
4. False Positive (FP) – prediction positive, label negative

The most basic metric is accuracy, which is defined as

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

However, accuracy often fails to capture the whole picture. Let's consider a dataset of 1000 people, where 990 are healthy and 10 are sick. Such a dataset is called **imbalanced**, as the number of classes is distributed unequally. We will consider sick people to be positive class and healthy people to be negative class. On this dataset, we could achieve 99% accuracy by always predicting that the person is healthy. Such a model would be useless in practice, as it could not detect sick people.

1.1.1 Precision and Recall

Precision and recall deal with the issue of imbalanced datasets. They are defined as

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

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

Considering the same model, we will score 0% recall and undefined precision. If we were to increase the recall by always predicting sick, we would achieve 100% recall and 1% precision. A good model should thus have both high precision and recall.

1.1.2 F_β Score

The F_β score answers our requirements. It combines precision and recall into one metric and is defined as

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}}$$

We can control the importance of precision and recall by changing the β parameter.

1.1.3 Micro and Macro Averaging

We can use micro and macro averaging to extend the binary metrics to a multiclass case. In the case of micro averaging, we consider $\text{TP}, \text{TN}, \text{FP}$ and FN as the sum of respective values for each class. We then calculate the metrics as before. In the case of macro averaging, we calculate the metrics for each class and then average them. This results in treating each class equally irregardless of the number of samples in each class, compared to micro averaging, which treats each sample equally.

1.1.4 Interannotator Agreement and Cohen's Kappa

One can use the **Cohen's kappa** metric to assess agreement between annotators. Assume we have two annotators annotating a dataset of n samples with k classes. Denote class annotated by annotator i to sample j as c_{ij}. Denote a total number of samples annotated by annotator i as class c as n_{ic}. We can then calculate Cohen's kappa as

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where p_o is the observed agreement and p_e is the expected agreement. The expected agreement is calculated as

$$p_e = \sum_{i=1}^{k} \frac{n_{1i} n_{2i}}{n^2}$$

The observed agreement is calculated as

$$p_o = \frac{1}{n} \sum_{i=1}^{n} \frac{c_{1i} = c_{2i}}{n}$$

The interpretation of the metric is not well established, as it is unclear how high the metric should be to be considered good. Landis and Koch [9, page 165] suggest the following interpretation:

- $\kappa < 0$ – poor agreement
- $0 \leq \kappa < 0.2$ – slight agreement
- $0.2 \leq \kappa < 0.4$ – fair agreement
- $0.4 \leq \kappa < 0.6$ – moderate agreement
- $0.6 \leq \kappa < 0.8$ – substantial agreement
- $0.8 \leq \kappa < 1$ – almost perfect agreement

1.2 Notation intermezzo

From now on, we will use the following notation:

- Non-bold lowercase letters (e.g. x) will denote a scalar.
- Bold lowercase letters (e.g. $\mathbf{x}$) will denote a vector.

- Bold uppercase letters (e.g. $\mathbf{X}$) will denote a matrix.
- Letters with hat (e.g. $\hat{x}$) will denote a predicted value.
- $|V|$ will denote size of set V.

All vectors are column vectors unless otherwise stated.

1.3 Document Representation

To train a model, we need to represent the document in a way the model can understand. Let's consider a dataset D of N documents. We can use the following representations:

1.3.1 Bag of Words

One way to represent the document is to encode each word based on the number of occurrences in the document. Such is an idea of the Bag of Words (BOW). To obtain a BOW representation, we do the following:

1. Tokenize the dataset. This is a process of splitting the text into smaller units called tokens (usually words).
2. Create a vocabulary V of all the tokens in the dataset.
3. Represent document d as vector of weights $(C_d^1, C_d^2, \ldots, C_d^{|V|})$, where C_d^i is the number of occurrences of token i in document d.

This representation allows us to store the dataset as a sparse $N \cdot |V|$ matrix.

1.3.2 TF-IDF

TF-IDF is an extension of the BOW representation, where we also consider the document's token frequency. To represent the weight of token t in document d, we use the following formula:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \cdot \text{IDF}(t)$$

where TF and IDF are defined as

$$\text{TF}(t, d) = \frac{C_d^t}{\sum_{j \in V} C_d^j}$$

$$\text{IDF}(t) = \log \frac{|D|}{\sum_{d \in D}(C_d^t)}$$

1.3.3 Subword Units

The problem with preceding representations is that they cannot capture unseen words and require an extensive vocabulary. To solve this problem, we can use subword units. Unlike the previous representations, subword units are not fixed-sized.

Byte Pair Encoding

Popularized by Sennrich, Haddow, and Birch [10], the Byte Pair Encoding (BPE) algorithm is following:

1. Split sentences into words by whitespace and punctuation.
2. Split each word into separate characters and add the special end of the word token.
3. Initializes the vocabulary with all the found characters and the end of the word token.
4. Iteratively merge the most frequent pair of characters into a single character.
5. Stop when the vocabulary size reaches the desired size.

We then use the vocabulary to encode the document as a sequence of token ids.

WordPiece

WordPiece is a modification of BPE that uses a different merging strategy and pre-tokenization. It was first introduced by Schuster and Nakajima [11] and improved by Wu et al. [12]. For pre-tokenization, it adds a special token at the beginning instead of the end of the word. As for the merging strategy, we choose the pair that maximizes the likelihood of the training data when merged.

SentencePiece

Introduced by Kudo and Richardson [13], SentencePiece (SPM) is rather a modular tokenizer than an algorithm. It unifies the preprocessing and tokenization steps and allows different sub-word algorithms to be used. Among other things, it also addresses the problem of encoding multiple space characters, which is impossible in the above sub-word algorithms. It does so by introducing a special token for the space character.

Byte-level BPE

First described by Radford et al. [14], Byte-level BPE (BBPE) is a modification of BPE. It doesn't split into words and characters but directly into bytes. The authors noticed that the encoding was doing suboptimal merges as adding exclamation marks to the end of the words. To solve this problem, merges can only happen between the same character categories.

1.4 Machine Learning Models

This section reviews the ML models we will use in experiments. Our intention is not to provide a comprehensive review of text classification approaches. If the reader is interested in such a work, we recommend work by Kowsari et al. [15]. We will not review either CNNs or Recurrent Neural Networks (RNNs). However, we will refer to them and expect the reader to know them.

1.4.1 Multinomial Logistic Regression

Multinomial Logistic Regression (MLR) is a simple linear model. The MLR with k classes is represented by matrix of weights $\mathbf{W} \in R^{k,d} = (\mathbf{w_1}, \mathbf{w_2}, \dots \mathbf{w_n})^T$ and vector of biases $\mathbf{b} = (\beta_1, \beta_2, \dots \beta_k)$.

Given input $\mathbf{x} \in R^d$, with correct class C_t, the probability of model predicting class C_i is given by

$$\text{softmax}(\mathbf{z}) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$
$$P(C_i|\mathbf{x}; \mathbf{W}, \mathbf{b}) = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b})$$

The optimized loss function is cross-entropy loss:

$$\mathcal{L}(\mathbf{x}; \mathbf{W}, \mathbf{b}) = \log P(C_t\mathbf{x}; \mathbf{W}, \mathbf{b})$$

The model is simple, so it can't capture complex relationships between features. However, it serves as a good baseline model, and unlike later models, it can be easily interpreted.

1.4.2 Transformers

Transformers have been first proposed for machine translation by Vaswani et al. [3]. They have since become a SoTA model for many NLP tasks, including Text Classification. The original paper suggested using encoder-decoder architecture

due to the nature of the machine translation task. However, for text classification, there is no need for the decoder. Thus, the architecture can be simplified to only the encoder. In the following sections by **transformer**, we will refer to the encoder part of the architecture.

Architecture

The architecture of the transformer can be seen at Figure 1.1. The key components are

- Embedding layer
- Multi-Head Attention Block
- Task Head layer

We will now describe each of them in detail.

Embedding Layer

The first layer of the transformer is an embedding layer, it takes the input text encoded as a vector $\mathbf{x} \in R^n$, with values in the range $[0, V)$, where V is the vocabulary size. The embedding layer embeds each vector element into vector space by employing a learnable matrix $\mathbf{E} \in R^{V,d}$, where d is the arbitrary embedding dimension. To encode the word's position in the sentence, we add a learnable positional embedding $\mathbf{P} \in R^{n,d}$. Thus, the output of the embedding layer is given by

$$\mathbf{O} = \mathrm{OneHot}(\mathbf{x}) \cdot \mathbf{E} + \mathbf{P}$$

where $\mathrm{OneHot}(\mathbf{x})$ is a matrix of one-hot vectors, where each row corresponds to the one-hot vector of the corresponding element in $\mathbf{x}$. This makes the model only accept fixed-length inputs. [1]

Self-attention

Self-attention is a key mechanism of transformer architecture. The Queries ($\mathbf{Q}$), Keys ($\mathbf{K}$), and Values ($\mathbf{V}$) matrices are computed from inputs as follows

$$\mathbf{Q} = \mathbf{X}\mathbf{W_q}$$
$$\mathbf{K} = \mathbf{X}\mathbf{W_k}$$
$$\mathbf{V} = \mathbf{X}\mathbf{W_v}$$
$$\text{where } \mathbf{W_q}, \mathbf{W_k}, \mathbf{W_v} \in R^{d,d} \text{are trainable matrices}$$

[1] The original paper allowed for variable length inputs by employing a different strategy for positional encoding.

The output of the self-attention layer is then computed as follows [2]

$$\mathbf{O} = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}}\right)\mathbf{V}$$

The intuition behind the self-attention is that by doing a scalar product between query and key vectors, the model learns to focus on the most relevant parts of the input. Such parts will then have the highest weight in the output in values multiplication.

Multihead attention

The multi-head attention is an extension of self-attention. Instead of doing just one attention, we do multiple attentions in parallel, every attention with different Query, Key and Value matrices. The outputs are then concatenated and passed through a linear layer.

Task Head Layer

Task Head Layer is task-dependent, as its goal is to convert the transformer's output to the desired output.

Benefits and Drawbacks

What transformers mostly improved were two things:

- Capturing of long-range dependencies.
- Parallelization of the model. Unlike RNNs, we don't process a single input at a time, but all of them in parallel. This allows for much faster training.

However, advantages haven't come without a cost. One of the biggest problems with transformers is their memory complexity. The self-attention layer requires $O(n^2)$ memory and $O(d \cdot n^2)$ time complexity. This limits the use of transformers to lower-length inputs. However, much work has been done in recent years to mitigate this issue [16].

[2] The dimension hidden state of self-attention can be different from the embedding dimension. We use the same for simplicity.

1.4.3 Transfer Learning

Transfer learning is a technique that allows using knowledge gained from one task to solve another. The popular method of transfer learning is fine-tuning. To fine-tune a model, we take a pre-trained model (**backbone**) and train it on a new task with a new task-specific head.

1.4.4 Language Model

The popular backbone models are Language Models (LMs). The goal of the LM is to predict the next token in a sentence. Due to the simplicity of the task, it can be trained in an unsupervised manner allowing for large-scale training. The approach is relatively old and was used even before the transformers with RNNs [17, 18]. The transformers allowed this method to scale to much larger datasets and models.

1.4.5 BERT

Bert [4] is a successor to the previous attempts of training transformer LMs, namely ELMo [19] and GPT [20]. The main difference between BERT and previous methods is that it uses a bi-directional encoder. Previous work used a single-direction encoder; LM model only attending to the left or right context. The bi-directional encoder allows the model to attend to the context in both directions. That's achieved by the Masked Language Modeling (MLM) optimization objective.

Masked Language Modeling

The goal of the MLM is to predict the masked tokens in the text. The model is trained to predict the masked words by taking input tokens and replacing some of them with a special token `[MASK]`. As the model won't encounter the mask tokens during inference, some tokens in the input are replaced with random tokens from the vocabulary instead of the mask token.

Next Sentence Prediction

To improve the model's ability to capture relationships between two sentences, the BERT further uses the Next Sentence Prediction (NSP) objective. Given two sentences, the model is trained to predict if the second sentence follows the first in the original text.

Input Representation

The BERT uses a WordPiece tokenization (section 1.3.3) to convert the input text to tokens. Apart from learned tokens, BERT introduces three special tokens:

1. `[CLS]` – Token appended to start of token sequence. The predicted output is based on the hidden state of this token.

2. `[SEP]` – The token appended to the end of the token sequence.

3. `[MASK]` – The token that is used for masking in section 1.4.5.

Bert-base and Bert-large

	BERT-base	BERT-large
Hidden size	768	1024
Number of layers	12	24
Number of attention heads	12	16

Table 1.1 Comparison of BERT-base and BERT-large.

Bert model exists in two versions: BERT-base and BERT-large; they differ by the number of layers and hidden layer size as seen in Table 1.1.

1.4.6 RoBERTa

	BERT	RoBERTa
Batch size	256	512
Sequence length	128 tokens (90%) and 512 tokens (10%)	512 tokens (100%)
Tokenization	WordPiece	BPE
Data	16 GB	160 GB
Objective	MLM + NSP	MLM

Table 1.2 Comparison of BERT and RoBERTa.

RoBERTa [21] is a further refinement of the BERT model. It doesn't change the model's architecture, but instead changes how the model is trained. The key differences are displayed in Table 1.2.

1.4.7 GPT-3

GPT-3 is a family of models introduced by Brown et al. [8]. Due to them being developed by **OpenAI** [3], they are not publicly available and not much is known about them. As per Brown et al. [8], the GPT-3 models are transformer-based LMs with parameters ranging from 125M to 175B, trained on a large multi-lingual dataset with more than 570GB of text. The tokenizer is based on the BBPE (section 1.3.3) algorithm. To solve efficiency issues (section 1.4.2), the model also employs sparse attention modification [22]. The OpenAI offers paid fine-tuning and inference of the derivatives of GPT-3 models, but no information about the models themselves.

1.4.8 Multi-linguality of Language Models

Both BERT and RoBERTa are trained exclusively on English data. To allow extension to other languages, multi-lingual models were introduced. Namely, XLM [23], XML-R [24] and m-BERT [25]. GPT-3 models could also be considered a multi-lingual model as 7% of the training data is in non-English languages and show great results on translation tasks to English.

While the models have shown multi-lingual capabilities, they possess two major drawbacks:

- The embedding layer must be substantial to accompany many languages.
- Length of tokenized sequences tends to be very long due to multi-lingual tokenization training.

It has been also shown by [6] and [26] that the mono-lingual models can still outperform the multi-lingual models on certain target language tasks, while being significantly smaller.

1.4.9 Czech Mono-lingual Language Models

Czech pre-trained LMs are depicted in Table 1.3. We will now briefly describe Fernet-News and RobeCzech models, as we will use them in our experiments.

Fernet-News

Developed at the *University of West Bohemia in Pilsen*, Fernet-News is a Czech RoBERTa model trained on over 3.3B words of Czech text. The training data mainly consists of Czech news articles and transcripts of TV and radio shows.

[3] `https://openai.com/`

Model	Base Model	Tokenizer	Vocab	Training Data	# Params
M-BERT [4]	BERT-base	WordPiece	120k	Wiki, 104 langs	179M
XLM-RoBERTa-base [24]	RoBERTa-base	SPM	250k	CC, 100 langs (2TB)	278M
XLM-RoBERTa-large [24]	RoBERTa-large	SPM	250k	CC, 100 langs (2TB)	560M
Czert [27]	BERT-base	WordPiece	40k	Syn+Wiki+News (37GB)	110M
RobeCzech [6]	RoBERTa-base	BBPE	52k	Syn+Wiki+Czes+W2C	126M
FERNET-C5 [7]	BERT-base	SPM	100k	C5 (93GB)	164M
FERNET-News [7]	RoBERTa-base	BBPE	50k	News Corpus (21GB)	124M
Small-E-Czech [28]	Electra-base	WordPiece	30k	Internal corpus (253GB)	13 M

Table 1.3 Table comparing multi-lingual and Czech mono-lingual models. Electra-base [29] is a transformer architecture trained with discriminative pre-training rather than generative. *CC* stands for Common Crawl. The table is a slightly modified version of Lehečka and Švec [7, Table 2].

The tokenizer is BBPE based, containing 50k tokens. The model was trained for 700k steps with a batch size of 2048 and a learning rate of 1e-4. Unlike the original RoBERTa model, the model was first pre-trained on 128-long inputs for 600k steps, followed by 100k steps of 512-long inputs.

RobeCzech

Developed at the *Charles University in Prague*, RobeCzech is a Czech RoBERTa model trained on over 4.1B tokens of Czech text. The training data are gathered from 4 different sources:

- **SYN v4** [30] – a corpus of contemporary (written) Czech texts. It contains a large proportion of journalistic texts from the Czech presses.
- **W2C** [31] – corpora containing texts from Wikipedia and web. As it contains 120 languages, only the Czech part was selected.
- **Czes** [32] – a corpus of Czech magazine and newspaper articles.
- **Wiki** [6] – Wikipedia extracted texts.

The vocabulary was created with BBPE and contains 51960 tokens. The model was trained for 91k steps with a batch size of 8192 and a learning rate of 7e-4.

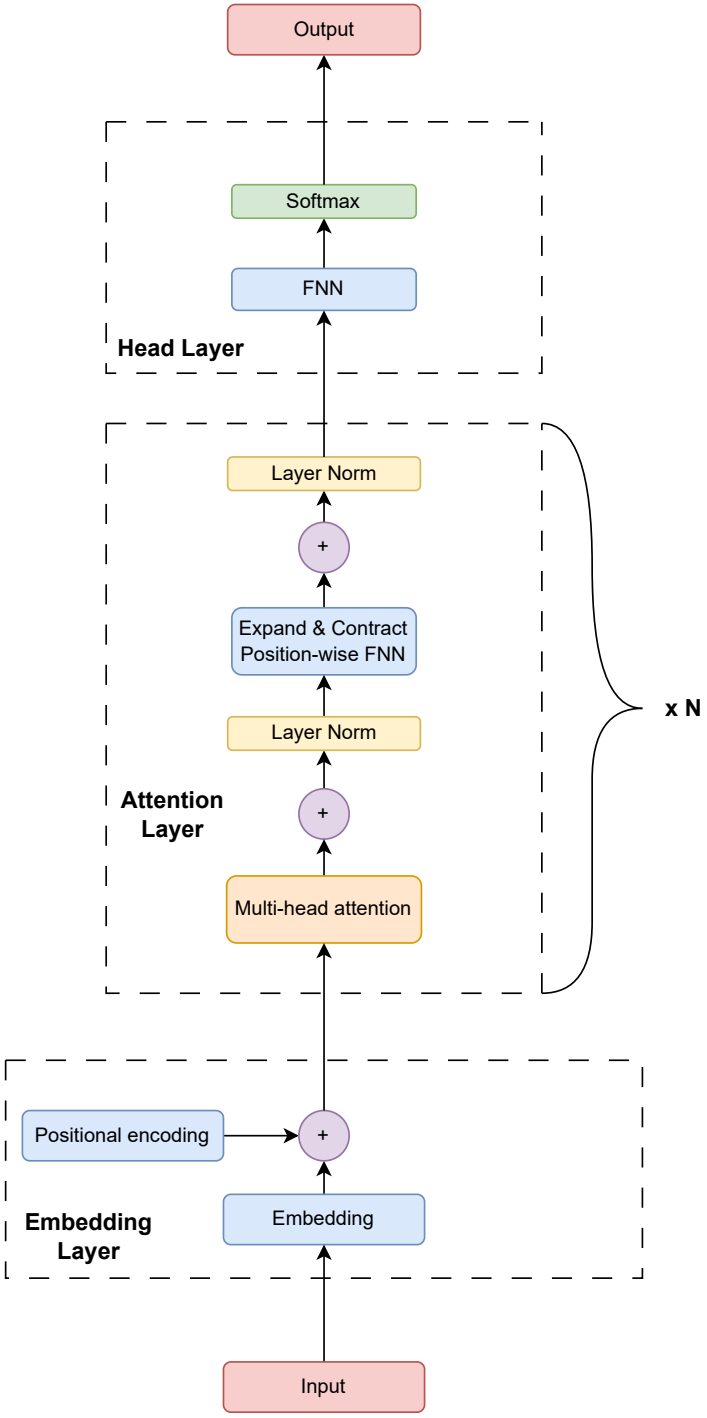

Figure 1.1 Transformer architecture without Dropout blocks for clarity.

Chapter 2

Czech News Classification dataset

CZEch NEws Classification dataset (CZE-NEC) is compiled from news stories published online in major Czech media outlets between January 2000 and August 2022. The news article content is protected by copyright law; therefore, we cannot distribute the dataset directly. Instead, we release a script[1] for dataset collection.

2.1 Dataset Creation Process

2.1.1 Data source

We collected the data from the following news servers: *SeznamZprávy.cz*, *Novinky.cz*, *Deník.cz*, *iDnes.cz*, *Aktuálně.cz*, *iHNed.cz* and *iRozhlas.cz*.

We used Common Crawl[2] as a data source, as crawling live websites would be infeasible. For extraction, we developed a custom tool *C'monCrawl*[3], which allows end-to-end extraction of Common Crawl data. We then deployed it in distributed setting on Artificial Intelligence Cluster (AIC)[4], processed 49.2M URLs and extracted 3.2M articles.

2.1.2 Filtering

Filtering was done in several steps. We employed Hugging Face (HF) datasets[5] library, as it allows for parallel processing of the data. This allowed us to filter the data in a few hours on a single machine.

[1] `https://github.com/hynky1999/Czech-News-Classification-dataset`
[2] `https://commoncrawl.org/`
[3] `https://github.com/hynky1999/CmonCrawl`
[4] `https://aic.ufal.mff.cuni.cz/`
[5] `https://huggingface.co/docs/datasets/index`

iHNed.cz articles

Similar to Straka et al. [33], we found that the iHNed.cz articles contained a high number of paywalled articles and overall contained few samples. We thus decided to remove all articles by iHNed.cz.

Czech filtering

Since we were interested in Czech articles, we decided to filter out articles not in the Czech language. For this purpose, we used FastText Language detection model [34, 1]. For every article, we used the model to predict the language of every line. This allowed us to interpret fractions of lines predicted as Czech as the confidence of the article being in the Czech language. We inspected the articles with lower confidence and the most occurring problems were:

1. Articles in Ukraine language on SeznamZprávy.cz, due to the recent war in Ukraine.
2. Articles with a list of sports results, where most texts were: the result, team and match highlights.
3. Articles with comparison tables, e.g., mobile comparison.
4. Galeries; there were few articles with galleries of pictures or videos with little text.
5. English articles on iDnes.cz and iRozhlas.cz.

Since we had many articles, we decided to filter out articles with a confidence lower than 1.0.

Removing wrongly parsed articles

To remove wrongly parsed articles, we only kept the ones with the following properties: content length of at least 400 characters, headline length of at least 20 characters, and a brief length of at least 40 characters. To exclude non-textual content, we only kept articles with the following properties:

1. The average word length is at least 4
2. The number of words per total article length in characters is in the interval (0.11, 0.22)
3. The ratio of non-alphanumeric characters is at most 4.5% per length (0, 0.045)

Filtering by headline content

As in Straka et al. [33], many articles contained prefixes at headlines like 'VIDEO: ', 'FOTO: ', 'GALERIE: ' etc.... Since we were interested in the articles and not galleries, we dropped the articles with prefixes that indicate non-news content. However, unlike Straka et al. [33], we didn't remove these prefixes in non-filtered headlines/briefs.

Headline/Brief/Content dedupliation

The last filtering round removed articles with identical briefs, headlines or content. We were afraid that this would also affect the article across different servers. It turned out that the deduplication only deleted around 3k articles because of cross-server duplicates. When choosing, which duplicate to use, we took the one with the most metadata filled or the longer article length. Therefore, every Brief/Content/Headline is unique in the dataset.

2.1.3 Data Augmentation and Postprocessing

Category

Due to the wide variety of collected data, we had to normalize categories. After extraction, we got a total of 3383 categories. Since there was considerable overlap between each category, we selected 25 categories among the most popular ones. We focused on choosing the categories with the most samples, while ensuring a slight overlap between selected categories. That's why we dropped categories like *News, Tips, Your News, Other, etc.*..., even though they had many samples. We then mapped from the remaining categories to these 25 categories if such a mapping was possible. Examples of such mappings are:

1. Football, Tennis, Biathlon... → Sport
2. Praha, Domažlicko, Ústecko... → Home
3. She, Women, Fashion... → Lifestyle

Authors

After extraction, there were 27k unique authors. The obvious problem was that not all authors were people. Surprisingly, the most prevalent were the news institutions: *ČTK, IDnes, MF DNES, etc.*... There were also many nicknames we couldn't decode, companies, and common names like *Redakce, externí, etc.*...

We employed heuristics and manual filtering to mitigate these problems and ended up with 11K authors. As for postprocessing, we removed occupation and academic titles.

Gender

To infer the gender of the author's name, we used Namsor [6]. If the article contained more than one author, we chose the homogeneous gender if possible. Otherwise, we labeled the Gender as Mixed

Postprocessing

Lastly, we applied common postprocessing steps including Unicode and HTML normalization and formatting adjustments to content, brief and headline.

2.1.4 Splits

We divided the dataset into the train, validation, and test sets based on publication date, using a 34:3:3 ratio. The sets are ordered by publication date, i.e., the train set contains the earliest articles, while the test set contains the latest articles. The dataset division is depicted in Figure 2.1.

Due to experiments, we also additionally created the following splits:

1. **Train Small** – 50K most recent samples from the training set, containing all metadata
2. **Test Small** – 10K randomly selected samples from the test set, containing all metadata
3. **Test Human** – 100 randomly selected samples from the test set, containing all metadata

2.2 Dataset Summary

The summarization of the dataset is shown in Table 2.1. The dataset contains the following features:

- *Server* – Server that published the article
- *Content* – Actual text content of the article

[6] `https://namsor.app/`

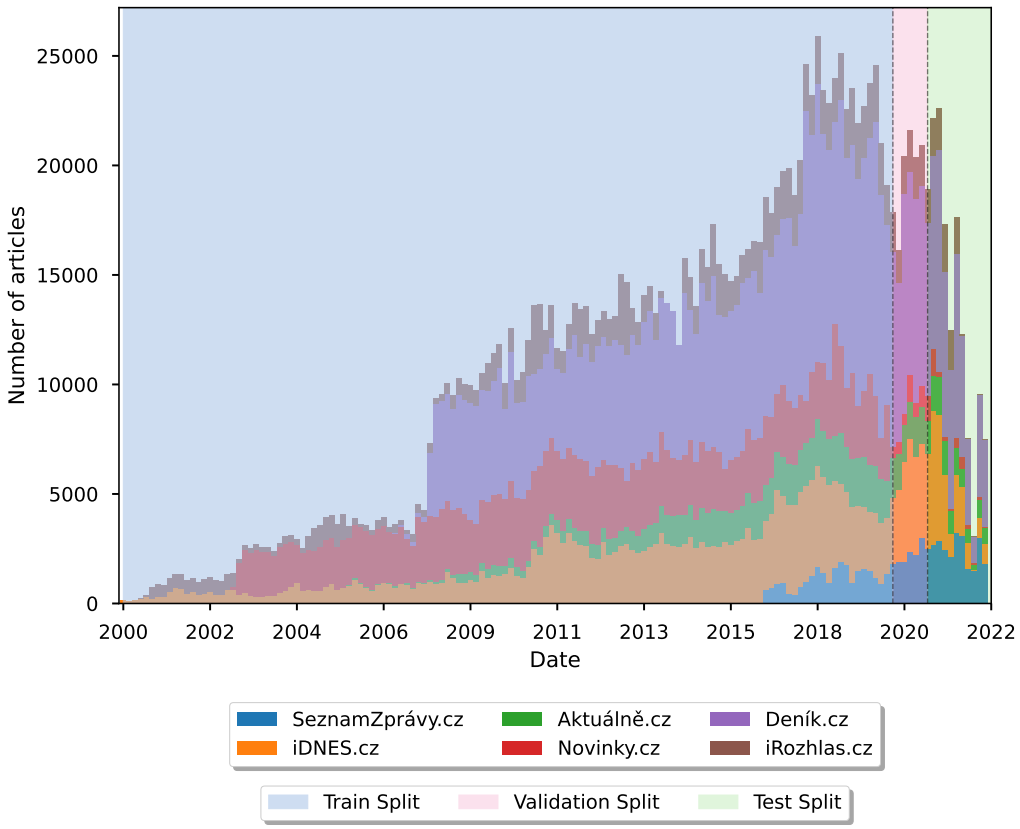

Figure 2.1 Distribution of news servers over time with dataset split boundaries.

- *Brief* – Brief/Perex of the article
- *Headline* – Headline/Title of the article
- *Category* – Both post-processed and original category
- *Published Date* – Date of publication and inferred day of week
- *Authors* – List of article authors
- *Inferred gender* – Inferred gender of author(s) name(s)
- *Keywords* – Extracted keywords from the article
- *Comments Count* – Number of comments in the discussion section

Server	Size	Authors	Categories	Start date	Words per article
Deník.cz	664,133	2,497	18	2007	332
Novinky.cz	321,417	2,518	17	2002	274
iDnes.cz	295,840	4,386	21	2000	423
iRozhlas.cz	167,588	1,900	8	2000	287
Aktuálně.cz	112,960	633	19	2005	468
SeznamZprávy.cz	65,472	382	11	2016	443
Total	1,627,410	10,930	25	2000	362

Table 2.1 Dataset summary. Article words were calculated based on Moses tokenization.

Set	Server	Category	Gender	Day of week
Train	1,383,298	879,019	919,840	1,383,298
Validation	122,056	78,084	82,936	122,056
Test	122,056	82,352	83,269	122,056
Total	1,627,410	1,039,455	1,086,045	1,627,410

Table 2.2 Tasks distribution over sets.

2.3 Task Definitions

We provide more details about the tasks in this section. Each task is provided an article body as input, excluding the brief and headline. As not all metadata are available for all articles, we show the distribution of samples over sets in Table 2.2.

2.3.1 Server

The Server classification task involves predicting the publishing server of articles from a set of 6 labels, as shown in Figure 2.2. It is important to note that there is a significant distribution shift between the training and validation set, which is caused by differences in the launch dates of the servers and parsing issues (especially with Novinky.cz)

2.3.2 Category

The Category classification task requires predicting the category of an article from a set of 25 labels, as depicted in Figure 2.3. When selecting the categories, we carefully identify the most frequent ones while striving to maintain diversity

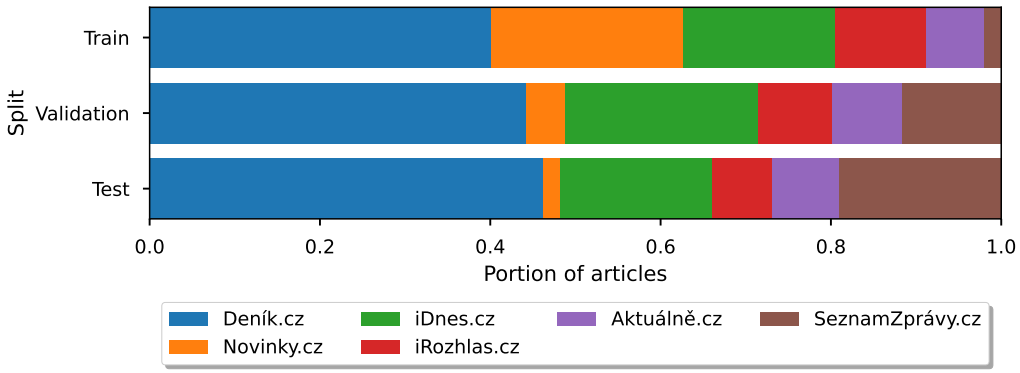

Figure 2.2 Graph depicting proportion of servers among the datasets.

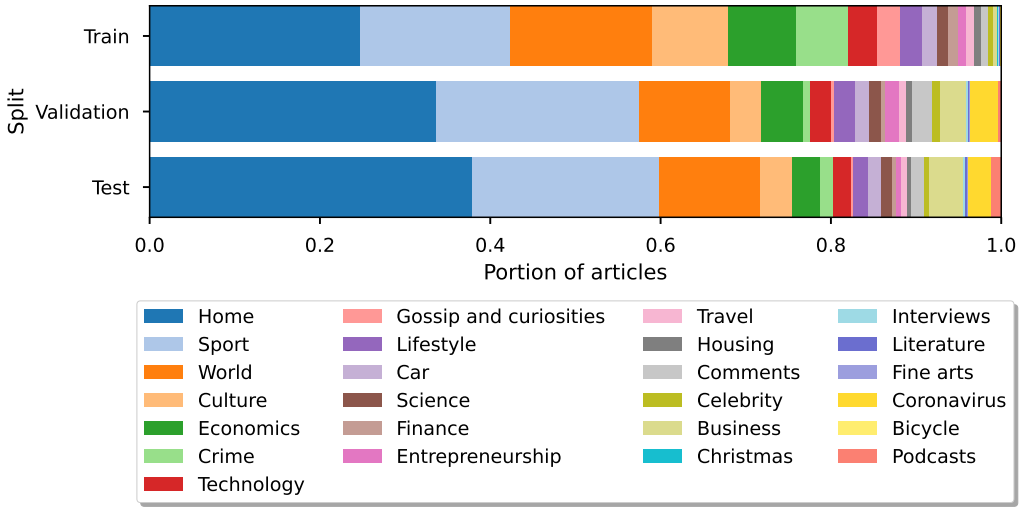

Figure 2.3 Graph depicting distribution categories among the datasets.

and minimize any potential overlap between them.

2.3.3 Gender

This classification task has 3 labels, as shown in Figure 2.4. Its goal is to predict the inferred gender of the article author(s). While we are aware that the inferred gender is not always correct, we think that it serves as a good proxy predicted, due to the strong association of social and grammatical gender in the Czech language. This task is not meant to label individuals and the text they produce, and we discourage future users of CZE-NEC from doing so.

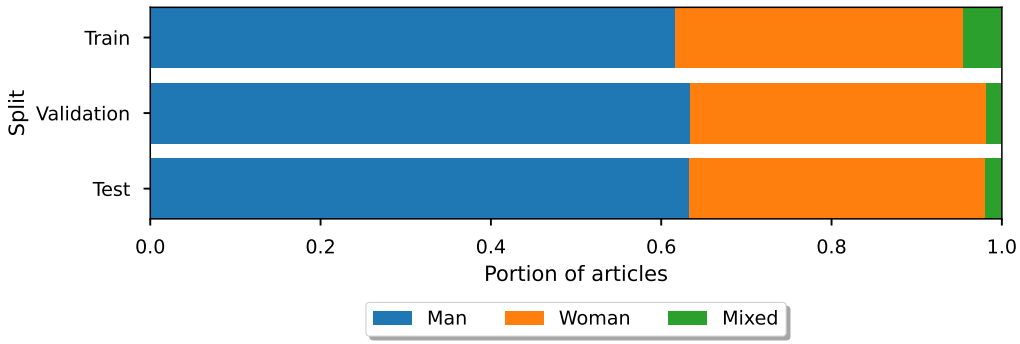

Figure 2.4 Graph depicting the distribution of genders among the datasets.

2.3.4 Day of the Week

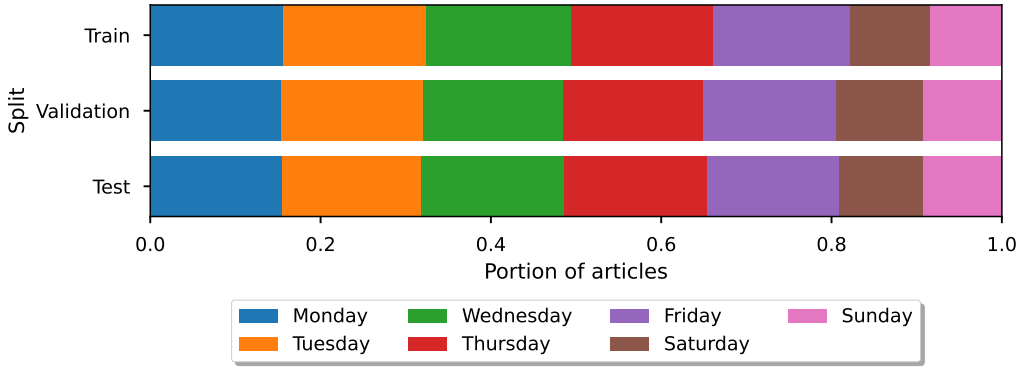

Figure 2.5 Graph depicting the distribution of weekdays among the datasets.

The Day of Week task is a classification challenge consisting of seven distinct labels, as illustrated in Figure 2.5. The objective is to accurately predict the day of the week a given article was published. Given the absence of any apparent approaches to tackle this task, we deem it to be the most challenging among the tasks considered.

Chapter 3

Experiments

In this section, we describe the experiments we conducted. We start with the human evaluation, then move onto ML models. To assess the performance of a keywords-based model we trained a MLR. A possible high performance of this model would mean that the tasks are solvable only by spotting typical keywords for the classes without any deeper understanding. Subsequently, we fine-tune Czech pre-trained Transformers on our tasks to evaluate the importance of textual dependencies for the tasks. Lastly, we fine-tune GPT-3 to evaluate its multilingual capabilities and compare the model to the substantially smaller Czech Transformers.

3.1 Human Performance

To evaluate human performance, we let four evaluators classify the articles on the Test Human set (subsection 2.1.4). Each evaluator got access to Google Sheet[1], where they had to assign each document a correct label for each task. Evaluators were offered all options for each task, not just the subset from the Test Human set. We will denote averaged scores of all evaluators as **Human**.

3.2 Baseline Model

We used MLR with TF-IDF features. Following Straka et al. [33], we included the following features:

- Number of words
- Number of words with only non-alphabetic characters

[1] https://docs.google.com/

- Number of uppercase words
- Number of digits words
- Number of capitalized words

We used 1-2 grams with a maximum document frequency of 1% and MosesTokenizer[2] with Czech stop words[3] on lowercase text as a tokenization method. We then run a grid search over Inverse Regularization term with values $1, 10, 100, 1000$ and selected the model with the highest F1 Macro score on the validation set. As for the solver, we used SAGA [35] implementation from Scikit-learn library[4], with max iterations set to 800 and an early stopping tolerance of 0.001.

With this setting, we created 2 models; **LR-50** with 50k TF-IDF features and **LR-200** with 200k TF-IDF features.

3.3 Pre-trained Transformers

For backbone models, we employed **RobeCzech**, **Fernet-News**, and **GPT-3**. We chose RobeCzech and Fernet-News as both are Czech mono-lingual transformers with the same architecture and the relatively same number of parameters (Table 1.3). The main difference is the training data and training setting (section 1.4.9, section 1.4.9). We hypothesized that the similarity of task and Fernet-News domains would lead to better task performance.

3.3.1 RobeCzech and Fernet-News

For fine-tuning, we mostly followed hyperparameter settings recommend by Sun et al. [36]. As for architecture, we took the backbone model and replaced the head as shown in Figure 3.1. We fine-tuned both models for 2 epochs with linear decay, 0.1 warmup, 48 effective batch size, and AdamW as the optimizer. All layers were unfrozen except for the embedding layer. Learning rates were selected based on the best validation score of a grid search over a 0.4 fraction of the training data. Possible learning rate values were:

1. RobeCzech: 3e-5, 4.5e-5, 7.5e-5
2. Fernet-News: 1e-5, 2e-5, 3e-5

[2] `https://pypi.org/project/mosestokenizer/`
[3] `https://pypi.org/project/stop-words/`
[4] `https://scikit-learn.org`

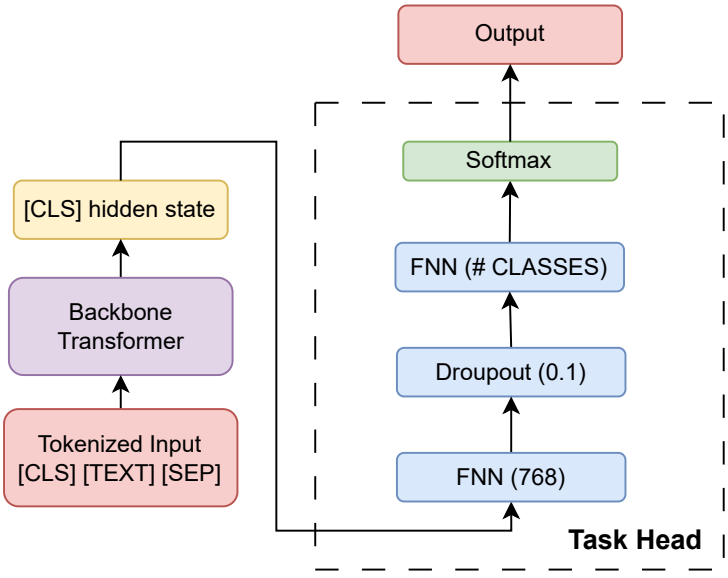

Figure 3.1 Fine-tuning architecture for the classification task.

The proposed learning rate values for RobeCzech and Fernet-News differ due to the divergence of Fernet-News with higher learning rates. To deal with long texts, we chose to truncate them to the first 510 tokens

For implementation, we used *Pytorch Lightning 2.0*[5], *HF Transformers 4.24*[6] and *Pytorch 2.0.0*[7]. We trained models on AIC with a single GeForce RTX 2080 Ti. We denote the models as **R-Base** and **F-Base**.

3.3.2 GPT-3

We chose the Ada version of GPT-3, as it is the cheapest. Ada costs 0.00004$/1k tokens for fine-tuning and 0.00016$/1k tokens for inference. The model was trained in a multi-task setting, utilizing the article text (query) and corresponding task labels in Czech (text completion) as input. For example, a sample text completion might be:

"Deník.cz Sport Žena Pondělí"

The Ada model takes a maximum of 2048 tokens; thus, we truncated documents to the first 1400 characters. To save the costs, we used **Train-small** for fine-tuning and **Test-small** (see subsection 2.1.4) for inference. We finetuned for 2 epochs.

[5] `https://www.pytorchlightning.ai/`
[6] `https://huggingface.co/docs/transformers/index`
[7] `https://pytorch.org/`

For comparison, we fine-tuned and evaluated RobeCzech and Fernet-News on the same splits. We denote the models as **GPT-3**, **R-Small** and **F-Small**.

3.4 Fine-tuning Enhancements

We were interested in ways to improve fine-tuning performance without changing the backbone model. We tested three approaches inspired by Howard and Ruder [37] and Sun et al. [36]. All the experiments were done on **RobeCzech** model, with settings as in subsection 3.3.1 unless stated otherwise.

3.4.1 Truncation

The base models are trained with truncation to the first 510 tokens. Due to the nature of the task, we hypothesized that the last part of the article might contain more relevant information:

> Pro Idnes.cz Jana Křížová

Therefore, we truncated the text by taking the last 510 tokens. We denote the model as **Truncate**.

3.4.2 Further Language Modeling

Both Howard and Ruder [37] and Sun et al. [36] found that Further Language Modeling (FLM) on task dataset can improve performance. We thus further pre-trained RobeCzech on the content of the article in FULL-SENTENCES setting [21] with a batch size of 192 and a learning rate of 5e-5 for 10 epochs. We then used the pre-trained model as the backbone and trained with the same setting as the Base model. We denote the resulting model as **LM-Tune**.

3.4.3 Gradual Unfreezing with Discriminative Learning Rates

Howard and Ruder [37] showed that we could improve model performance by gradually unfreezing layers. As the original method was used on RNNs, we were interested if it would work with Transformers. We also added a discriminative learning rate which sets a smaller learning rate for the lower layers. We set the discriminative factor to 0.95 as in [36]. The Gradual Unfreezing (GU) is not well described in [37], thus we tried two approaches:

1. **Grad-12** Unfreeze 1 layer per epoch, starting from the last layer, and run for 12 epochs.

2. **Grad-24** Unfreeze 1 layer per epoch, starting from the last layer, and run for 24 epochs (12 epochs in the full unfrozen state).

The epoch lengths were adjusted, so that the total number of optimizer steps stays the same as in subsection 3.3.1 (full 2 epochs). For both approaches, we unfroze the classifier layer at the start of training with the learning rate decaying from 1e-3 to 5e-5. The remaining layers were adjusted based on the discriminative learning rate. Each unfrozen layer had a scheduler with the same setting as Base models (section 3.3). However, it started scheduling when the layer was unfrozen.

3.5 Final Model

Finally, we created a model combining the best pre-trained transformer and fine-tuning approaches. Therefore, we used **RobeCzech** as a backbone with FLM. Additionally, we reused the classifier scheduling as in subsection 3.4.3. Due to the great results on the restricted dataset, we also lowered the warmup steps to just 0.01 and added Most-recent sampling.

3.5.1 Most-recent Sampling

To add weight to more recent articles, we sampled article i from the dataset with the following probability:

$$P(i) = \frac{\exp(2d_i)}{\sum_{j=1}^{n} \exp(2d_j)}$$

where d_i equals:

$$d_i = \frac{t_i}{\max_{0 \leq j \leq n} t_j}$$

and where t_i is the time difference between the published date of article i and the dataset's eldest article in days.

Chapter 4

Results and discussion

In this chapter, we present the results of our experiments. When comparing tasks performance, we implicitly use F1 Macro, as all tasks but Day Of Week are imbalanced. We start with an overall comparison of the models. We then perform a closer analysis of the individual tasks and put them into the context of the related research.

4.1 Results

4.1.1 Human Agreement and Performance

	Server		Category		Gender		Day of week	
	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro
Human	27.03	30.25	40.26	60.76	50.09	61.75	13.53	13.75
Final	**71.22**	**80.00**	**52.04**	**79.59**	**52.79**	**79.00**	**28.37**	**29.00**

Table 4.1 Results on the Human Test set. We use bold to denote the best result for each task.

We found the average Cohen's kappa to be: 0.08 for Server, 0.65 for Category, 0.20 for Gender and 0.01 for Day Of Week. Based on the classification from subsection 1.1.4, only the category task showed at least substantial agreement.

The performance results presented in Table 4.1 shows that the final models significantly outperform the human baseline. The largest difference, a 44 percent improvement, was observed on the Server task.

Overall, low agreement and performance suggest the high difficulty of the tasks.

	Server		Category		Gender		Day of week	
	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro
LR-50	36.92	52.78	33.30	72.32	43.62	69.15	18.13	19.62
LR-200	37.27	53.38	32.77	72.69	44.06	69.83	18.34	19.97
R-Base	69.74	78.19	54.35	79.67	51.18	74.67	29.43	29.49
F-Base	69.39	77.68	53.97	79.55	-	-	29.24	29.34
R-Small	59.48	67.85	36.55	77.46	44.97	69.95	17.42	19.61
F-Small	-	-	37.84	77.72	-	-	16.08	17.99
Truncate	68.71	77.31	53.90	79.37	50.36	74.21	29.52	29.57
LM-tune	70.06	78.40	55.18	80.14	51.13	**75.09**	**29.98**	**30.00**
Grad-12	67.81	76.36	51.93	78.69	49.98	73.82	29.11	29.21
Grad-24	69.07	77.69	53.22	78.97	49.75	74.06	29.36	29.47
Final	**71.04**	**79.25**	**56.06**	**80.47**	**51.94**	75.04	29.68	29.69

Table 4.2 Results on the Test set. We use - to denote a failure of a model to converge for all tested learning rates.

4.1.2 Baseline Model

The baseline model results are shown in Table 4.2. A tiny improvement was gained by adding the additional 150k features. Appendix B shows the features with the highest weights for each task and class, adding insight into the model's reasoning process.

4.1.3 Pre-trained Transformers

Table 4.2 shows a significant improvement of Pre-trained transformer models over Logistic Regression across the tasks. This demonstrates the importance of capturing textual dependencies for better performance. Contrary to the initial expectation that Fernet-News would achieve higher scores due to its same-domain training data, RobeCzech outperformed Fernet-News across the tasks. One possible explanation could be RobeCzech's slightly higher capacity, which may be more important for long training.

	Server		Category		Gender		Day of week	
	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro	F1-macro	F1-micro
R-Base	**78.43**	**82.88**	**56.17**	**80.65**	**52.38**	**75.61**	**27.96**	**28.33**
F-Base	78.04	82.26	55.51	80.51	-	-	27.25	27.78
R-Small	75.12	80.37	37.88	77.69	47.45	74.30	17.41	19.82
F-Small	-	-	39.31	77.88	-	-	17.68	19.20
GPT-3	67.30	73.12	44.76	75.21	42.92	70.28	19.49	20.83

Table 4.3 Results on the Test Small. We use - to denote a failure of a model to converge for all tested learning rates.

The Small train setting yielded opposite results as seen in Table 4.3. For every task where F-Small didn't diverge, it outperformed R-Small. The dominance of Fernet-News in the Small setting could be explained by better weights initialization, as the domain of the data is the same in the case of Fernet-News unlike in RobeCzech. Compared to the fully trained models, small models showed great results considering that they were trained on less than 6% of the data. However, it is apparent that further training is beneficial (Table 4.2).

Regarding GPT-3, it outperformed both Small models on Category and Day of Week, showing its multi-lingual capabilities. The model exhibited the tokenization issues of multilingual models (subsection 1.4.8). It tokenized just 1.77 characters per token compared to 4.68 and 4.46 in the case of RobeCzech and Fernet-News, making the training more expensive than expected. Overall we spent around 40$ for both training and inference.

Surprisingly, we observed better Server task results of the models on the Test Small. The underlying reason for this observation remains unclear, especially considering that the distributions are relatively similar across the test sets. The only noticeable change is iDnes.cz having slightly higher representation at the expense of Deník.cz.

4.1.4 Fine-tuning Enhancements

Article beginning truncation didn't help the performance, as seen in Figure 4.1. The same was observed with GU, which even significantly worsened the performance in some cases. From validation curves, we can observe that in all tasks but Category, GU performance falls behind considerably at the beginning of the training and catches up only at the end. That is caused by the different alignment of the tasks with the Language Modelling objective. Since the weights of lower layers are frozen at the start, good initialization is needed to keep up with the fully-unfrozen model. The only positive aspect of GU was smaller memory consumption at early training stages, allowing for larger batch sizes.

Out of all fine-tuning approaches, only FLM increased the performance.

4.1.5 Final Model

The performance of the Final model can be seen in Table 4.2. It shows the importance of further pretraining and recency sampling.

4.2 Tasks Evaluation

We selected the Final model for in-depth evaluation of all tasks, even though it wasn't the best-performing model for Day Of Week.

4.2.1 Server

	Precision	Recall	F1
SeznamZprávy.cz	82.44	65.06	72.72
iDnes.cz	68.56	75.60	71.91
Aktuálně.cz	83.08	81.30	82.18
Novinky.cz	29.64	67.75	41.23
Deník.cz	91.75	86.39	88.99
iRozhlas.cz	60.43	80.95	69.20
Macro Avg	69.32	76.17	71.04

Table 4.4 Classification report of Server on Test set.

To our knowledge, there is no previous work on the Server task. Therefore our only comparison is to human performance, which was outperformed by the model significantly.

We observed the most problematic origin to be Novinky.cz. The model incurs a significant precision drop on this origin as seen in Table 4.4. This could be explained by the distribution change in a train and test set. Compared to the training set, the test set contains a smaller number of samples from Novinky.cz. Overpredicting Novinky.cz is then natural behavior. The opposite was observed with SeznamZprávy.cz. The model misclassifies underrepresented SeznamZprávy.cz mainly as iDnes.cz (Figure C.1), which results in a low recall.

4.2.2 Category

The Category was researched in the works of [38] and [39]. In the first work, the authors report 90.85% F1 Micro; however, the authors only used four categories from a single source. The second work is more comparable as the authors used the same number of categories (25) but from a single source. The authors report 68.38% F1 Micro. We thus find our results reasonably good.

When observing the results(Table 4.5, Figure C.2) we noticed our preprocessing likely made the task harder. For example, the model often misclassified Fine Arts and Literature as Culture, due to Culture being a parent category of both. Similar could be observed with Business, Entrepreneurship and Finance being mistaken for Economy.

	Precision	Recall	F1
World	82.62	86.77	84.65
Home	86.89	82.91	84.86
Sport	97.78	97.97	97.87
Culture	58.07	87.26	69.73
Celebrity	81.91	67.78	74.18
Gossip and Curiosities	28.08	45.34	34.68
Economics	51.35	65.99	57.76
Crime	60.58	62.30	61.43
Entrepreneurship	28.68	51.80	36.92
Car	85.45	86.31	85.88
Science	54.35	51.48	52.87
Comments	68.88	85.70	76.38
Travel	36.83	39.55	38.14
Finance	60.21	59.44	59.82
Technology	79.76	82.70	81.20
Housing	67.56	75.69	71.39
Coronavirus	42.72	33.24	37.39
Business	62.78	51.72	56.71
Interviews	7.95	8.64	8.28
Podcasts	0.00	0.00	0.00
Lifestyle	53.12	46.00	49.30
Literature	90.24	82.68	86.30
Christmas	11.11	25.00	15.38
Fine Arts	92.19	71.08	80.27
Bicycle	0.00	0.00	0.00
Macro Avg	55.57	57.89	56.06

Table 4.5 Classification report of Category on the Test set.

Additionally, we found that the model often misclassified Coronavirus as Home. When looking at the data, we found that some servers do not have the category Coronavirus and rather classify the article as Home, since the topic of the article is often also about politics. The fact that articles can frequently be assigned to multiple categories raises the question of whether single-label evaluation is appropriate.

4.2.3 Gender

Gender task has also been studied in a few works such as by [40] . The paper is a results overview of the shared gender classification task in Dutch. The data are gathered from multiple Dutch news sources but don't consider multi-author articles. Authors report 68.9% F1 Micro for the best model. This makes our results look great. Note, however, that the language difference could play a huge role.

	Precision	Recall	F1
Man	82.70	78.73	80.67
Woman	63.79	72.22	67.75
Mixed	23.89	4.38	7.41
Macro Avg	56.79	51.78	51.94

Table 4.6 Classification report of Gender on Test set.

Confusion matrix (Figure C.3) and classification report (Table 4.6) show that the Mixed class is highly problematic for the model, which likely stems from low data representation. When it comes to Man and Woman comparison, we observed Woman to have both precision and recall smaller. While we expected the recall to be smaller, due to imbalance, we didn't expect the precision to be lower as well.

4.2.4 Day of week

	Precision	Recall	F1
Monday	30.08	34.02	31.93
Tuesday	27.69	28.23	27.95
Wednesday	30.28	27.57	28.86
Thursday	25.45	34.57	29.32
Friday	33.22	28.60	30.73
Saturday	34.86	22.63	27.44
Sunday	33.95	29.37	31.49
Macro Avg	30.79	29.28	29.68

Table 4.7 Classification report of Day Of Week on Test set.

No such study of the task is known to us. We anticipated the task to be the most challenging one, which was shown to be true by the lowest agreement and performance by humans among the tasks. Considering the human results, we find our results excellent.

From the confusion matrix (Figure C.4), we can see the model misclassifying the days as surroundings of the target day. The only exception to this rule is weekend days. This can be observed with both Monday and Friday, as they are rather misclassified as random weekdays rather than surrounding Sunday, respectively Saturday.

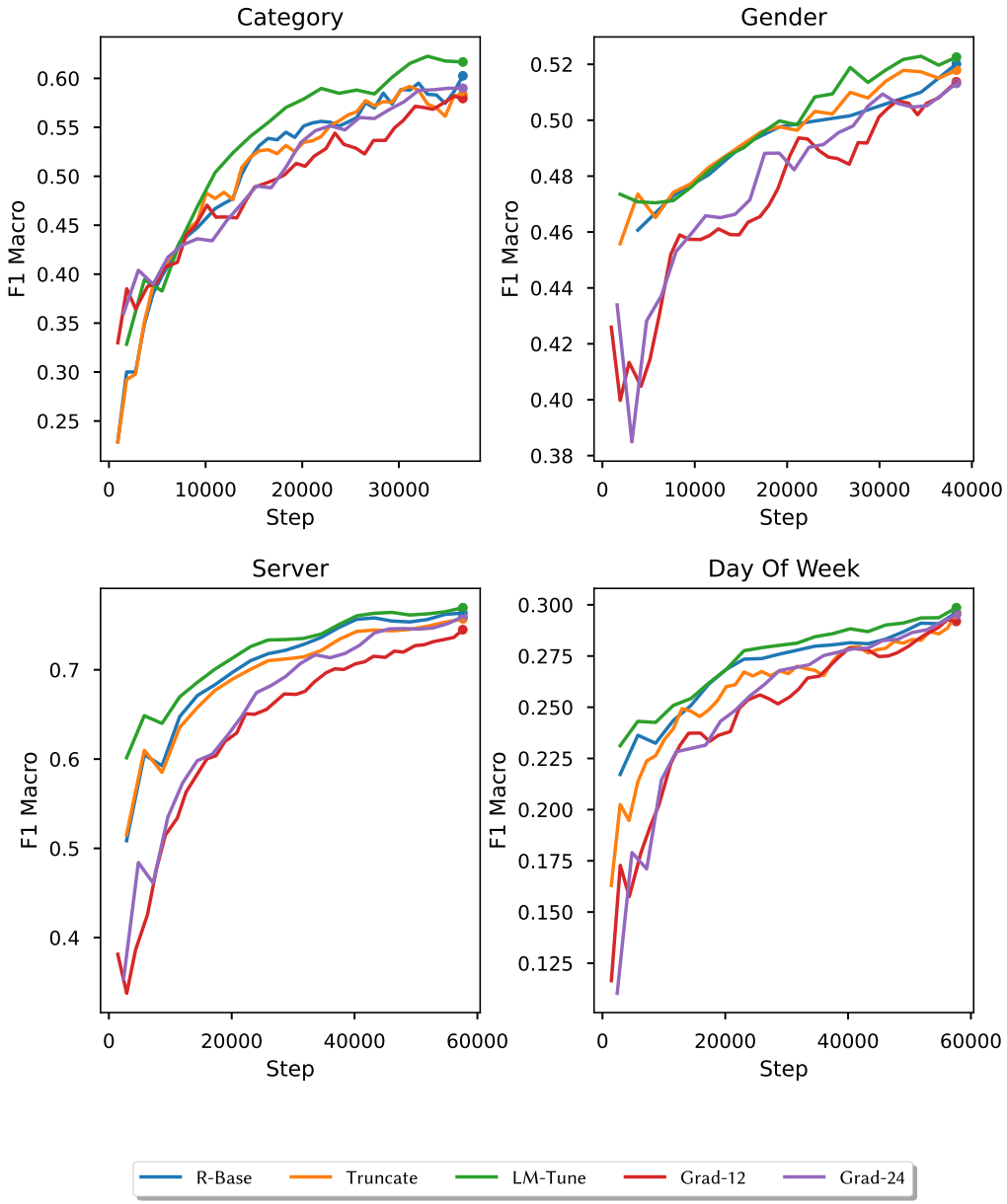

Figure 4.1 Validation curves for all tasks and fine-tuning approaches. The curves are displayed with linear smoothing. Each Validation was run on 10% of the Validation set. The last validation was run on the whole Validation set.

Conclusion

4.3 Our Contribution

In this work, we proposed **CZE-NEC**, an extensive dataset with a large amount of metadata, created with our **C'monCrawl** utility. We then trained and evaluated several ML models, showing that additional textual dependencies greatly improve the performance on the tasks compared to simple keyword extraction.

Additionally, we tested a few fine-tuning enhancements of pre-trained transformers, among others revealing that GU worsens the performance of the models, contrary to findings of Howard and Ruder [37] and that combination of FLM and recency sampling can further improve the performance of the pre-trained transformers on our tasks.

We also compared the performance of the pre-trained Czech transformers, to the commercial multi-lingual model GPT-3, showing that the Czech models can still beat the significantly larger Large Language Model (LLM) on some tasks.

Lastly, we observed our best models perform significantly better than humans on all tasks.

We open-source the code for C'monCrawl[1]. Due to copyright issues, we can't redistribute the dataset, but we offer a script for data collection [2]. Furthermore, we share all of our final models at HF hub.

- **Server**: `https://huggingface.co/hynky/Server`
- **Category**: `https://huggingface.co/hynky/Category`
- **Gender**: `https://huggingface.co/hynky/Gender`
- **Day of week**: `https://huggingface.co/hynky/Day_of_week`

We also created a user-friendly application (see Appendix A), combining all models into a single interface.

[1] `https://github.com/hynky1999/CmonCrawl`
[2] `https://github.com/hynky1999/Czech-News-Classification-dataset`

4.4 Future Work

When it comes to the dataset itself, we have discussed the problematic crawling of Novinky.cz (subsection 2.3.1). We have also pointed out the Category task fallbacks (subsection 4.2.2), which could be improved.

When it comes to tasks themselves, we haven't dealt with transformers' memory issues as aligned in section 1.4.2. We thus encourage researches to apply memory-efficient transformers to the tasks. Further, it would be interesting to include more features when classifying.

Lastly, due to the extensiveness of the dataset, other Classification or Regression tasks could be researched i.e. predicting the number of comments in the discussion section.

Bibliography

[1] Armand Joulin et al. *Bag of Tricks for Efficient Text Classification*. Aug. 9, 2016. arXiv: 1607.01759 [cs]. URL: http://arxiv.org/abs/1607.01759. preprint.

[2] Xiang Zhang and Yann LeCun. *Text Understanding from Scratch*. Apr. 3, 2016. arXiv: 1502.01710 [cs]. URL: http://arxiv.org/abs/1502.01710. preprint.

[3] Ashish Vaswani et al. *Attention Is All You Need*. Dec. 5, 2017. arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762. preprint.

[4] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 24, 2019. arXiv: 1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805. preprint.

[5] Alex Wang et al. "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP. Brussels, Belgium: Association for Computational Linguistics, 2018, pp. 353–355.

[6] Milan Straka et al. "RobeCzech: Czech RoBERTa, a Monolingual Contextualized Language Representation Model". In: vol. 12848. 2021, pp. 197–209. arXiv: 2105.11314 [cs].

[7] Jan Lehečka and Jan Švec. "Comparison of Czech Transformers on Text Classification Tasks". In: vol. 13062. 2021, pp. 27–37. arXiv: 2107.10042 [cs].

[8] Tom B. Brown et al. *Language Models Are Few-Shot Learners*. July 22, 2020. arXiv: 2005.14165 [cs]. URL: http://arxiv.org/abs/2005.14165. preprint.

[9] J. Richard Landis and Gary G. Koch. "The Measurement of Observer Agreement for Categorical Data". In: *Biometrics* 33.1 (1977), pp. 159–174. JSTOR: 2529310.

[10] Rico Sennrich, Barry Haddow, and Alexandra Birch. *Neural Machine Translation of Rare Words with Subword Units*. June 10, 2016. arXiv: 1508.07909 [cs]. URL: http://arxiv.org/abs/1508.07909. preprint.

[11] Mike Schuster and Kaisuke Nakajima. "Japanese and Korean Voice Search". In: *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Mar. 2012, pp. 5149–5152.

[12] Yonghui Wu et al. *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. Version 2. Oct. 8, 2016. arXiv: 1609.08144 [cs]. URL: http://arxiv.org/abs/1609.08144. preprint.

[13] Taku Kudo and John Richardson. "SentencePiece: A Simple and Language Independent Subword Tokenizer and Detokenizer for Neural Text Processing". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 66–71.

[14] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: 2019.

[15] Kamran Kowsari et al. "Text Classification Algorithms: A Survey". In: *Information* 10.4 (4 Apr. 2019), p. 150.

[16] Bohan Zhuang et al. *A Survey on Efficient Training of Transformers*. Feb. 2, 2023. arXiv: 2302.01107 [cs]. URL: http://arxiv.org/abs/2302.01107. preprint.

[17] Andrew M Dai and Quoc V Le. "Semi-Supervised Sequence Learning". In: *Advances in Neural Information Processing Systems*. Vol. 28. Curran Associates, Inc., 2015.

[18] Matthew E. Peters et al. *Semi-Supervised Sequence Tagging with Bidirectional Language Models*. Apr. 28, 2017. arXiv: 1705.00108 [cs]. URL: http://arxiv.org/abs/1705.00108. preprint.

[19] Matthew E. Peters et al. *Deep Contextualized Word Representations*. Mar. 22, 2018. arXiv: 1802.05365 [cs]. URL: http://arxiv.org/abs/1802.05365. preprint.

[20] Alec Radford and Karthik Narasimhan. "Improving Language Understanding by Generative Pre-Training". In: 2018.

[21] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. July 26, 2019. arXiv: 1907.11692 [cs]. URL: http://arxiv.org/abs/1907.11692. preprint.

[22] Rewon Child et al. *Generating Long Sequences with Sparse Transformers*. Apr. 23, 2019. arXiv: 1904.10509 [cs, stat]. URL: http://arxiv.org/abs/1904.10509. preprint.

[23] Guillaume Lample and Alexis Conneau. *Cross-Lingual Language Model Pretraining*. Jan. 22, 2019. arXiv: 1901.07291 [cs]. URL: http://arxiv.org/abs/1901.07291. preprint.

[24] Alexis Conneau et al. *Unsupervised Cross-lingual Representation Learning at Scale*. Apr. 7, 2020. arXiv: 1911.02116 [cs]. URL: http://arxiv.org/abs/1911.02116. preprint.

[25] Telmo Pires, Eva Schlinger, and Dan Garrette. *How Multilingual Is Multilingual BERT?* June 4, 2019. arXiv: 1906.01502 [cs]. URL: http://arxiv.org/abs/1906.01502. preprint.

[26] Raphael Scheible et al. *GottBERT: A Pure German Language Model*. Dec. 3, 2020. arXiv: 2012.02110 [cs]. URL: http://arxiv.org/abs/2012.02110. preprint.

[27] Jakub Sido et al. *Czert – Czech BERT-like Model for Language Representation*. Aug. 20, 2021. arXiv: 2103.13031 [cs]. URL: http://arxiv.org/abs/2103.13031. preprint.

[28] Matěj Kocián et al. *Siamese BERT-based Model for Web Search Relevance Ranking Evaluated on a New Czech Dataset*. Dec. 3, 2021. arXiv: 2112.01810 [cs]. URL: http://arxiv.org/abs/2112.01810. preprint.

[29] Kevin Clark et al. *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators*. Mar. 23, 2020. arXiv: 2003.10555 [cs]. URL: http://arxiv.org/abs/2003.10555. preprint.

[30] Michal Křen et al. *SYN v4: Large Corpus of Written Czech*. 2016.

[31] Martin Majliš. *W2C – Web to Corpus – Corpora*. 2011.

[32] Unknown author. *Czes*. 2011.

[33] Milan Straka et al. "SumeCzech: Large Czech News-Based Summarization Dataset". In: *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*. LREC 2018. Miyazaki, Japan: European Language Resources Association (ELRA), May 2018.

[34] Armand Joulin et al. *FastText.Zip: Compressing Text Classification Models*. Dec. 12, 2016. arXiv: 1612.03651 [cs]. URL: http://arxiv.org/abs/1612.03651. preprint.

[35] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. *SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives*. Dec. 16, 2014. arXiv: 1407.0202 [cs, math, stat]. URL: http://arxiv.org/abs/1407.0202. preprint.

[36] Chi Sun et al. *How to Fine-Tune BERT for Text Classification?* Feb. 5, 2020. arXiv: 1905.05583 [cs]. URL: http://arxiv.org/abs/1905.05583. preprint.

[37] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. ACL 2018. Melbourne, Australia: Association for Computational Linguistics, July 2018, pp. 328–339.

[38] Pramod Sunagar et al. "News Topic Classification Using Machine Learning Techniques". In: *International Conference on Communication, Computing and Electronics Systems: Proceedings of ICCCES 2020*. Springer. 2021, pp. 461–474.

[39] O. Fuks. "Classification of News Dataset". In: 2018.

[40] Hessel Haagsma et al. "Overview of the CLIN29 Shared Task on Cross-Genre Gender Prediction in Dutch." In: *CEUR workshop proceedings*. 2453. 2019, pp. 1–5.

Acronyms

AIC Artificial Intelligence Cluster

BBPE Byte-level BPE

BOW Bag of Words

BPE Byte Pair Encoding

CNN Convolutional Neural Network

CZE-NEC CZEch NEws Classification dataset

DL Deep Learning

FLM Further Language Modeling

FN False Negative

FP False Positive

GU Gradual Unfreezing

HF Hugging Face

LLM Large Language Model

LM Language Model

ML Machine Learning

MLM Masked Language Modeling

MLR Multinomial Logistic Regression

NLP Natural Language Processing

NSP Next Sentence Prediction

RNN Recurrent Neural Network

SoTA State of The Art

SPM SentencePiece

TN True Negative

TP True Positive

Appendix A

Gradio Application

Gradio [1] was used to create a web application incorporating the Final models. The application is hosted at HF Spaces [2]. A screenshot of the application is shown in Figure A.1.

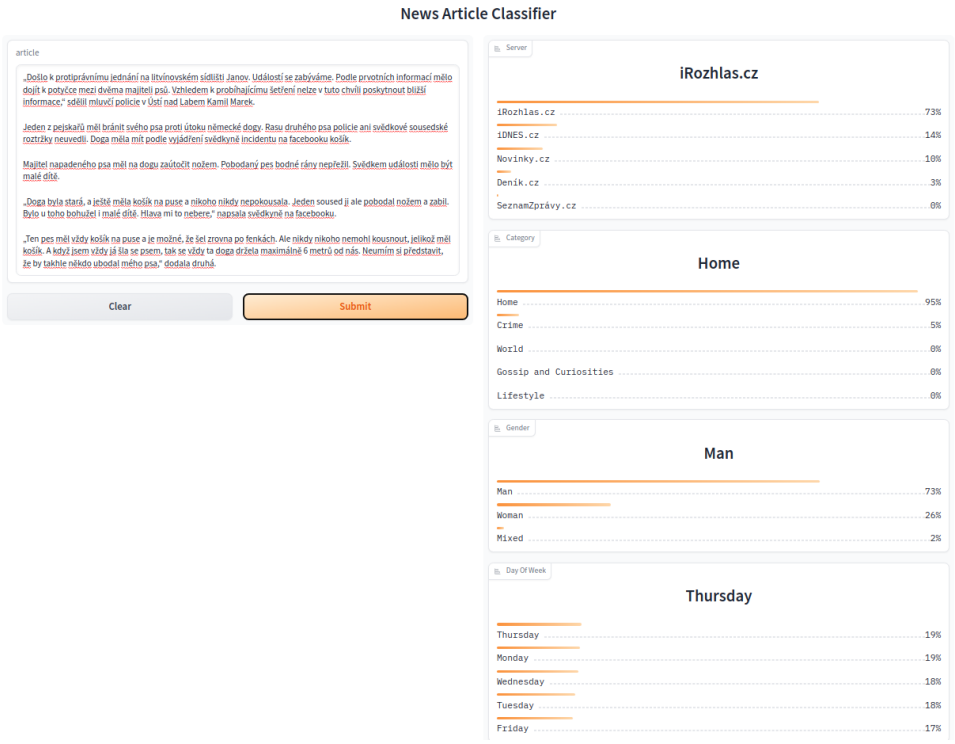

Figure A.1 Screenshot of the Gradio application for Czech News Classification.

[1] `https://gradio.app/`
[2] `https://huggingface.co/spaces/hynky/News-Classification`

Appendix B

Baseline Features

This appendix showcases features with the highest weight of LR-200 (section 3.2) for Server (Table B.1), Category (Table B.2), Gender (Table B.3 and Day of Week (Table B.4) tasks.

	1	2	3	4
SeznamZprávy.cz	videu	seznam zprávy	podívejte	” říká
iDNES.cz	idnes.cz	mf dnes	více zde	více čtěte
Aktuálně.cz	aktuálně.cz	praha –	video :	brno -
Novinky.cz	řekl právu	miliónů korun	novinkám	, právo
Deník.cz	čtěte také	deníku .	řekl deníku	-
iRozhlas.cz	radiožurnálu	radiožurnál	| zdroj	českého rozhlasu

Table B.1 Top 4 features with the highest weight of LR-200 for Server task.

	1	2	3	4
Home	mf dnes	radiožurnálu	idnes.cz	klaus
Sport	fotbalu	nhl	turnaji	kluby
World	osn	. úřady	: reuters	vyšetřovatelé
Culture	album	čt	koncertu	snímek
Economics	banka	analytik	řsd	ekonomiky
Crime	mluvčí policie	řekl právu	miliónů korun	řekla právu
Technology	herní	pc	windows	. hra
Tabloid News and Curiosities	miss	novinkám	daily	zpěvačka
Lifestyle	sexuální	university	sex	sexu
Car	automobilky	automobilů	motor	kw
Science	vědci	vzdělávání	fakulty	školách
Comments	.)	babiše	andrej babiš	zemana
Travel	turistů	turisté	—	profimedia.cz
Finance	bank	pojištění	banka	pojišťovny
Entrepreneurship	úřadu práce	okd	pivovaru	horal
Housing	interiéru	architekt	. :	vily
Coronavirus	koronaviru	koronavirem	koronavirus	nakažených
Business	akcie	čez	zaměstnavatel	meziročně
Interviews	videu	výzvě	? podívejte	podívejte
Podcasts	~	■	poslechněte	poslechněte si
Tabloid News and Curiosities	miss	novinkám	daily	zpěvačka
Literature	román	nakladatelství	kniha	románu
Christmas	vánoce	dárky	vánočních	cukroví
Fine Arts	galerie	národní galerie	obraz	děl
Bicycle	~	cyklisty	cyklisté	cyklistů

Table B.2 Top 4 features with the highest weight of LR-200 for Category task.

	1	2	3	4
Man	důchodový systém	důkazy	důchodců a	" kroutil
Woman	epa	došlých	elektricky	etapu
Mixed	band .	dějství	děkanka	bili

Table B.3 Top 4 features with the highest weight of LR-200 for Gender task.

	1	2	3	4
Monday	pondělní	pondělí ráno	minulého týdne	pondělí mluvčí
Tuesday	úterní	úterý ráno	úterního	pondělní
Wednesday	středeční	úterní	středu ráno	řekl středu
Thursday	čtvrteční	čtvrtek ráno	středeční	. čtvrtek
Friday	pátek ráno	příští týden	páteční	čtvrteční
Saturday	sobotu ráno	páteční	příští týden	sobotu dopoledne
Sunday	nedělní	václava moravce	neděli ráno	sobotním

Table B.4 Top 4 features with the highest weight of LR-200 for Day of Week task.

Appendix C

Confussion Matrices

Collection of confusion matrices of Final models (section 3.5) evaluated on the Test set.

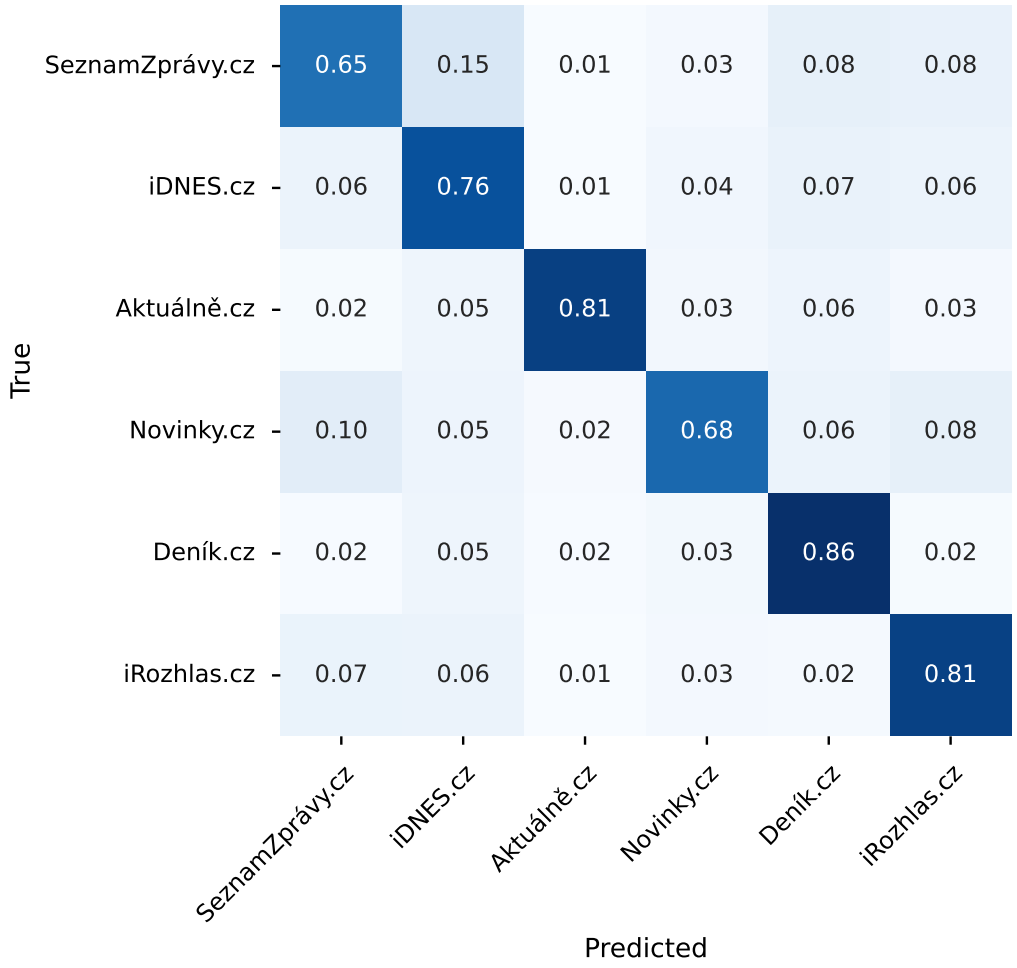

Figure C.1 Confusion matrix of Final model for Server task evaluated on the Test set. Matrix is normalized over True values, so that row sums are equal to 1.

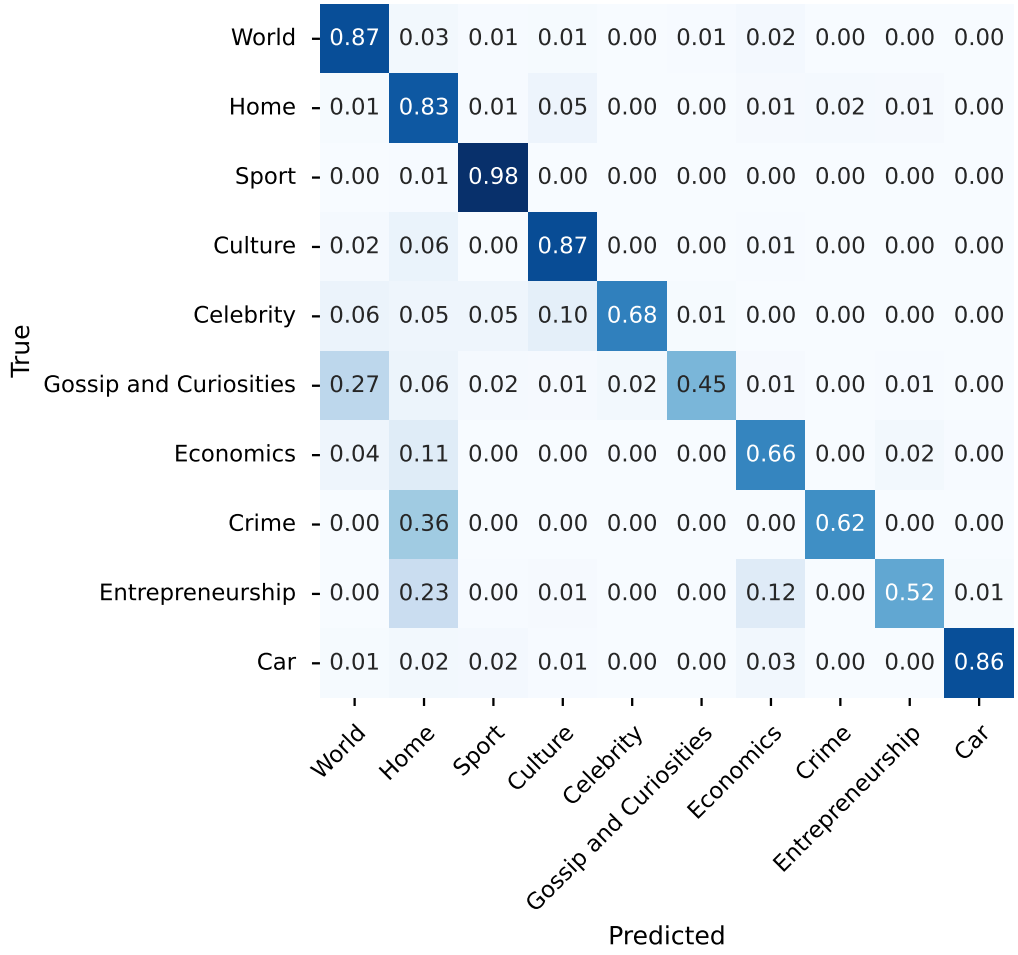

Figure C.2 Confusion matrix of Final model for Category task evaluated on the Test set. Matrix is normalized over True values, so that row sums are equal to 1. We only show the first 10 categories, due to visual clarity.

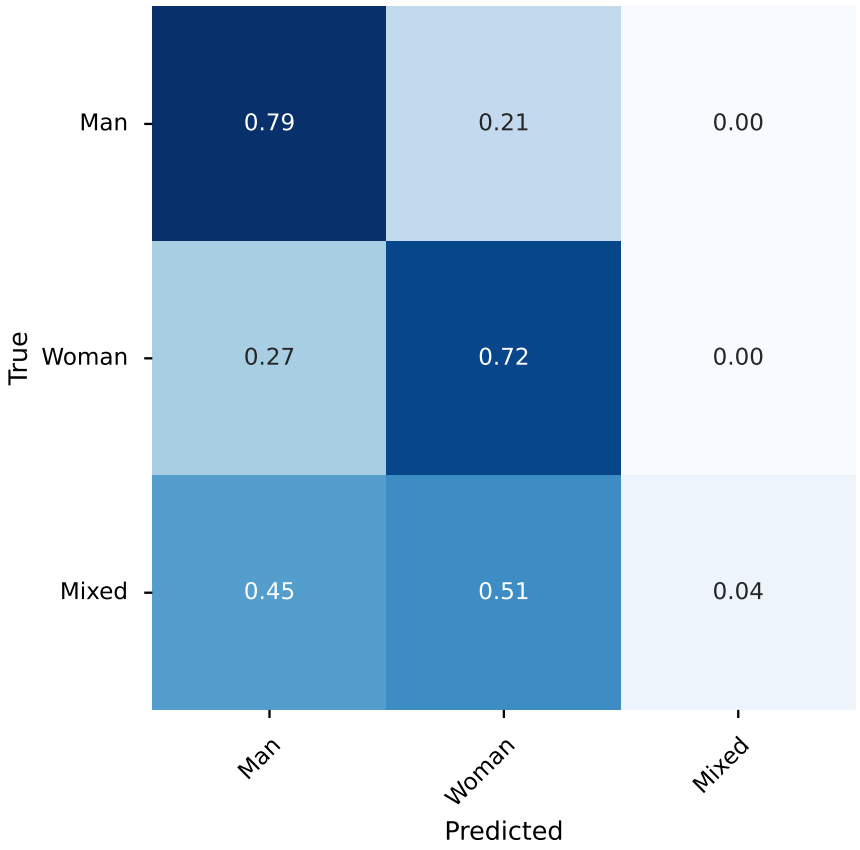

Figure C.3 Confusion matrix of Final model for Gender task evaluated on the Test set. Matrix is normalized over True values so that row sums are equal to 1.

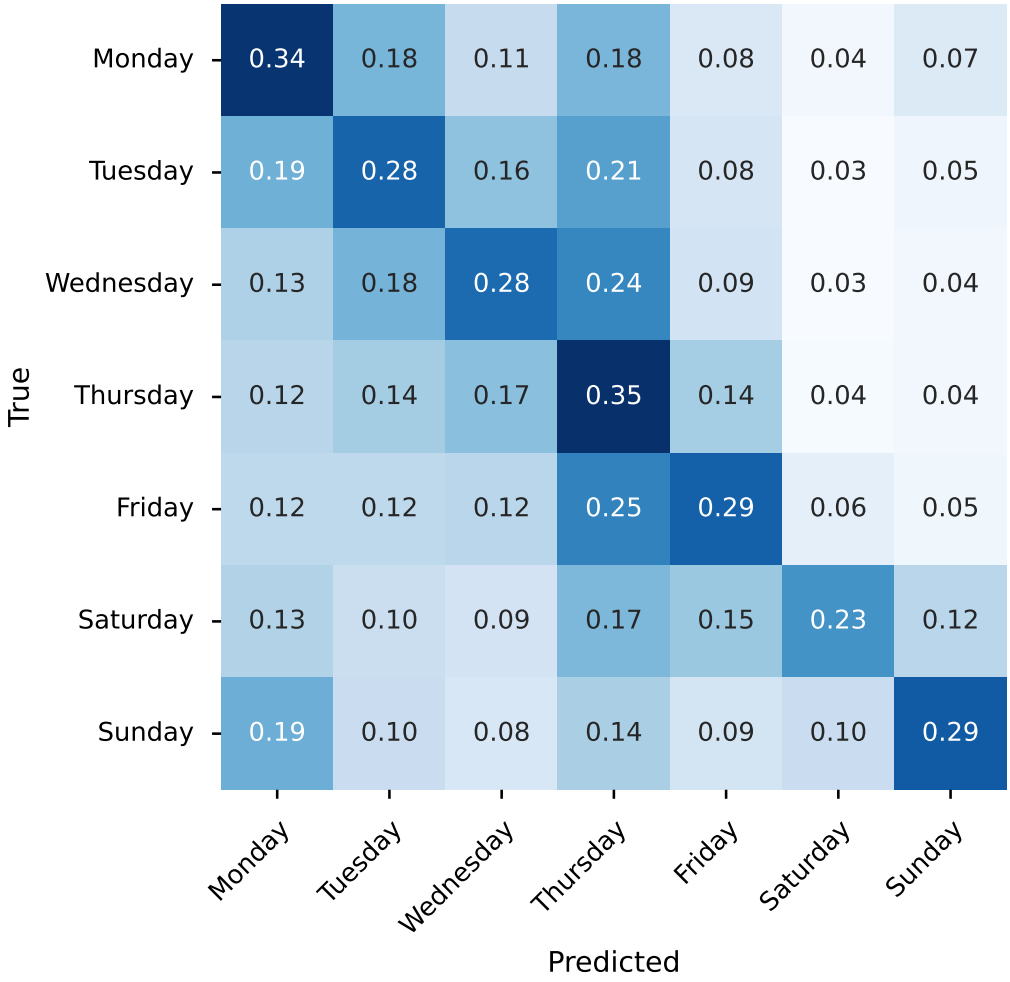

Figure C.4 Confusion matrix of Final model for Day Of week task evaluated on the Test set. Matrix is normalized over True values, so that row sums are equal to 1.