



**Czech
Technical University
in Prague**

F3

Faculty of Electrical Engineering
Department of Computer Science

Improving Facticity of Summarization Methods

Master Thesis of

Bc. Václav Halama

Supervisor: **Ing. Jan Drchal, Ph.D.**

Field of study: **Open Informatics**

Subfield: **Artificial Intelligence**

May 2023

I. Personal and study details

Student's name: **Halama Václav** Personal ID number: **508478**
Faculty / Institute: **Faculty of Electrical Engineering**
Department / Institute: **Department of Computer Science**
Study program: **Open Informatics**
Specialisation: **Artificial Intelligence**

II. Master's thesis details

Master's thesis title in English:

Improving Facticity of Summarization Methods

Master's thesis title in Czech:

Zkvalit ování fakticity generovaných sumarizací

Guidelines:

The task is to design and experiment with text summarization methods aiming at improving the facticity of generated summaries.

- 1) Review state-of-the-art abstractive and extractive summarization methods as well as summarization quality/facticity measures.
- 2) Design summarization methods using a summarization facticity evaluation model as a module. Experiment with both abstractive and extractive approaches.
- 3) Research and choose (or develop) a method to improve fluency of the summaries generated by extractive approaches.
- 3) Measure improvement in facticity of generated summaries on datasets supplied by the supervisor.

Bibliography / sources:

[1] Marian, Krottil. Metody sumarizace textu v češtině. BS thesis. České vysoké učení technické v Praze. Vypočetní a informační centrum., 2022.
[2] Šimon, Zvára. Ověřování fakticity výstupů metod abstraktivní sumarizace textu. BS thesis. České vysoké učení technické v Praze. Vypočetní a informační centrum., 2022.
[3] Zhang, Tianyi, et al. "Bertscore: Evaluating text generation with bert." arXiv preprint arXiv:1904.09675 (2019).
[4] Liu, Yixin, et al. "BRIO: Bringing Order to Abstractive Summarization." arXiv preprint arXiv:2203.16804 (2022).
[5] Suleiman, Dima, and Arafat A. Awajan. "Deep learning based extractive text summarization: approaches, datasets and evaluation measures." 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS). IEEE, 2019.

Name and workplace of master's thesis supervisor:

Ing. Jan Drchal, Ph.D. Artificial Intelligence Center FEE

Name and workplace of second master's thesis supervisor or consultant:

Date of master's thesis assignment: **31.01.2023** Deadline for master's thesis submission: _____

Assignment valid until: **22.09.2024**

Ing. Jan Drchal, Ph.D.
Supervisor's signature

Head of department's signature

prof. Mgr. Petr Páta, Ph.D.
Dean's signature

III. Assignment receipt

The student acknowledges that the master's thesis is an individual work. The student must produce his thesis without the assistance of others, with the exception of provided consultations. Within the master's thesis, the author must state the names of consultants and include a list of references.

Date of assignment receipt

Student's signature

Acknowledgements

I would like to express my gratitude to my supervisor, Jan Drchal, for his invaluable guidance throughout this thesis. I am grateful to Hana Vincourová for her help with the annotations. To my family and my girlfriend, thank you for your love and encouragement.

Declaration

I hereby declare that I have completed this thesis independently and that I have listed all the literature and publications used. I have no objection to usage of this work in compliance with the act §60 Zákon č. 121/2000Sb. (copyright law), and with the rights connected with the copyright act including the changes in the act.

In Prague, 22. May 2023

Abstract

This thesis explores the field of abstractive and extractive text summarization, focusing on improving the factual consistency of generated summaries. State-of-the-art methods and evaluation measures are reviewed, and a novel facticity evaluation metric called BERTSource is proposed. Extractive models based on the XLM-R model are trained to optimize ROUGE and BERTScore scores, outperforming previous extractive methods. Additionally, a rewriting model is developed to enhance the fluency of extractive summaries. Two different abstractive models were created by combining the extractive and rewrite models, further improving ROUGE and BERTScore metrics performance. Our fully abstractive models are trained using a novel training paradigm called BRIO, which encourages the model to prefer summaries of higher quality, with multiple quality measures considered. Human evaluation reveals that while the combined models perform best in factual quality, no improvement is achieved over existing methods. The BRIO-based models performed poorly in human evaluation, even though they outperformed all previous methods in automatic evaluation. This work highlights the limitations of current evaluation metrics and suggests the need for a new metric to assess the factual consistency of generated summaries accurately. Although improvements in facticity were not achieved, this work contributes valuable insights for future works.

Keywords: abstractive summarization, extractive summarization, NLP, facticity evaluation, fluency enhancement

Supervisor: Ing. Jan Drchal, Ph.D.

Abstrakt

Tato práce se zabývá abstraktivní a extraktivní sumarizací textu s důrazem na zlepšení faktické kvality generovaných sumarizací. Byly prozkoumány moderní sumarizační metody a metriky na posouzení kvality sumarizací a byla navržena nová metrika na vyhodnocení fakticity nazvaná BERTSource. Extraktivní modely založené na modelu XLM-R byly trénovány tak, aby optimalizovaly metriku ROUGE a BERTScore, díky čemuž překonaly předchozí extraktivní metody. Kromě toho byl vyvinut přepisovací model pro zvýšení plynulosti extraktivních sumarizací. Kombinací extraktivních modelů a přepisovacího byly vytvořeny dva nové abstraktivní modely, které dále zlepšily ROUGE a BERTScore. Plně abstraktivní modely byly trénovány pomocí nové metody nazvané BRIO, která podněcuje model k tomu, aby preferoval sumarizace vyšší kvality, přičemž bylo použito několik metrik pro měření kvality. Lidské vyhodnocení ukázalo, že ačkoli kombinované modely dosahují nejlepších výsledků ve faktické kvalitě, není dosaženo zlepšení oproti současným metodám. Modely založené na metodě BRIO v lidském hodnocení dosáhly nejhorších výsledků, přestože v automatickém vyhodnocení překonaly všechny předchozí metody. Tato práce upozorňuje na omezení současných automatických metrik a poukazuje na potřebu vyvinout nové metriky pro přesné vyhodnocení faktické kvality generovaných sumarizací. Ačkoli nebylo dosaženo zlepšení ve faktičnosti, tato práce přináší cenné poznatky pro budoucí výzkum.

Klíčová slova: abstraktivní sumarizace, extraktivní sumarizace, NLP, vyhodnocení fakticity, zlepšování plynulosti

Contents

1 Introduction	1	7.3.4 Results	30
1.1 Motivation	1	7.4 Rewrite model	31
1.2 Hallucinations	1	7.4.1 Rewrite dataset	31
2 Pre-trained language models	3	7.4.2 Training	32
2.1 Transformer architecture	3	7.4.3 Results	32
2.2 BERT	5	7.5 EXT-RW model	32
2.3 BART	5	7.5.1 Inference setting	33
2.4 mBART	6	7.6 BRIO	33
2.5 RoBERTa	6	7.6.1 Base model	34
2.6 XLM-RoBERTa	7	7.6.2 Reduced dataset	34
3 Summarization methods	9	7.6.3 Generating candidates	35
3.1 BertSum	9	7.6.4 Training	35
3.2 BART for summarization	10	7.6.5 Inference setting	36
3.3 BRIO	11	7.7 Results	37
3.4 Inference of sequence-to-sequence models	12	7.7.1 Automatic evaluation	37
3.4.1 Greedy search	12	7.7.2 Human evaluation	39
3.4.2 Beam search	12	7.7.3 Example summaries	40
3.4.3 Sampling	13	7.7.4 Discussion	40
3.4.4 Diverse beam search	14	8 Conclusion	43
4 Evaluation metrics	15	Bibliography	45
4.1 ROUGE	15	A Acronyms	49
4.2 BERTScore	16	B Translations of examples	51
4.3 ROUGE-CS	16	C Attached files	55
4.4 Memes-CS	16		
5 Datasets	19		
5.1 SumeCzech	19		
5.2 CNC	20		
5.3 Final training dataset	20		
6 Methodology	21		
6.1 Extractive method	21		
6.1.1 Model	21		
6.1.2 Rewriting extractive summaries	22		
6.2 Abstractive methods	22		
6.2.1 BRIO-based methods	22		
7 Experiments	25		
7.1 Implementation details	25		
7.1.1 Tokenization	25		
7.2 Source-based evaluation	26		
7.2.1 Annotated dataset	26		
7.2.2 BERTScore revisited	26		
7.2.3 Metrics comparison	28		
7.3 Extractive model	28		
7.3.1 Extractive datasets	28		
7.3.2 Training	29		
7.3.3 Inference setting	30		

Figures

1.1 Example of model hallucination	2
2.1 Transformer architecture, taken from [Vaswani et al., 2017].	4
2.2 Simplified architecture of BART, taken from [Lewis et al., 2019].	6
3.1 BERT and BertSum architecture, taken from [Liu and Lapata, 2019].	10
3.2 Beam search illustration with beam size equal to 2.	13
6.1 Overview of our extractive model architecture.	22
7.1 Model-layer search of source-based BERTScore.	27
7.2 Distribution of sentences selected as extractive summary.	29
7.3 Example from the rewrite dataset.	32
7.4 Example from the EXT-R-RW pipeline.	33
7.5 Example of noisy sample.	34
7.6 Training process of the BRIO-based models.	36
7.7 Example summaries.	41
B.1 Translation of the example of model hallucination	51
B.2 Translation of the example from the rewrite dataset.	51
B.3 Translation of the example from the EXT-R-RW pipeline.	52
B.4 Translation of the example of noisy sample.	52
B.5 Translation of the example summaries.	53

Tables

5.1 Sources of the SumeCzech dataset, taken from [Straka et al., 2018].	19
5.2 Statistics of the SumeCzech dataset splits.	19
5.3 Statistics of the CNC dataset splits.	20
7.1 Evaluation of metrics for source-based evaluation	28
7.2 Overlap-based metric results of extractive models.	30
7.3 Model-based metric results of extractive models.	30
7.4 ROUGE score results of the rewrite model on the development split of the generated dataset.	32
7.5 Overlap-based metric results.	37
7.6 Model-based metric results.	38
7.7 Human evaluation results.	39

Chapter 1

Introduction

1.1 Motivation

Summarization is an essential task in today's time of information overload. Identifying the most critical information and presenting it concisely, quickly, and coherently is very important for making sense of all the textual data generated daily. While the field of natural language processing has made significant progress in this area, with the development of advanced methods for both abstractive and extractive summarization, a major problem that still needs to be addressed is ensuring that the generated summaries are factually consistent.

Abstractive summarization involves generating a new summary by rephrasing the original text, while extractive summarization involves selecting important sentences from the original text. Both methods have advantages and disadvantages, and the choice of which method to use depends on the specific task and mainly the desired level of accuracy. The use of large pre-trained language models has dramatically improved the performance of these methods, but the factual accuracy of the generated summaries remains a concern.

This work aims to address this problem by developing techniques for improving the factual consistency of summarization methods. This will involve evaluating the generated summaries and incorporating the facticity evaluation model as a module in the training process. Using the facticity evaluation model may provide additional context and help ensure that the generated summaries are factually consistent.

We will also look at the limitations of existing techniques and identify areas that need improvement. New methods for addressing these limitations will then be proposed and evaluated. This work attempts to bridge the gap between the current state of summarization techniques and the ideal state of generating factually correct summaries.

1.2 Hallucinations

In the context of summarization, model hallucination refers to the phenomenon when the summary generated by a model contains information that is not present in the source document. Hallucinating models can pose a problem because they can lead to summaries that are misleading or factually inconsistent with the source.

All generative models tend to hallucinate, and the exact source of hallucinations is unknown. There are several factors that can cause hallucinations. These may include a small training dataset, lack of context in the source document, noise, and biases in the training data.

Figure 1.1 shows an example of model hallucinations. The example document is taken

Source document

Americký prezident upozornil, že to, co se nyní děje v Bělorusku, může mít vliv na celý region střední a východní Evropy: "Kvůli tomu jsme méně bezpeční." Doplnil i to, co je potřeba udělat: "Je důležité, abychom Bělorusy podporovali při budování občanské společnosti." Obama reagoval na to, že soud v Minsku ve čtvrtek poslal na šest let a pět a půl roku do vězení dva neúspěšné kandidáty na prezidenta Mikalaje Statkeviče a Dzmitryje Usse. Oba byli odsouzeni za organizování demonstrace loni v prosinci po znovuzvolení prezidenta Lukašenka. Bílý dům rozsudky už v pátek odsoudil a oznámil nové sankce proti režimu v Minsku. Barack Obama také jednal s polským premiérem Donaldem Tuskem. Šéf Bílého domu se po schůzce s prezidentem Komorowským pozdravil s lidmi, kteří se zasloužili o pád komunismu v Polsku. Byli mezi nimi aktivisté nezávislé odborové organizace Solidarita, mimo jiné Tadeusz Mazowiecki a Zbigniew Janas. Na schůzce Obamy s někdejšími polskými disidenty chyběl Lech Walesa, který v pátek polským médiím řekl, že se mu to setkání nehodí. Walesa se prý o setkání dozvěděl na poslední chvíli. Walesu za to kritizovali. "Politik, který kvůli uražené hrdosti odmítá setkání s lídrem světové velmoci, s velkým spojencem Polska, dělá obrovskou chybu," uvedla Gazeta Wyborcza a dodala: "Tím, že ani chvíli nepromluvil s americkým prezidentem, Walesa urazil hosta polských nejvyšších představitelů."

Generated summary

Šéf Bílého domu Barack Obama v pátek po setkání s **běloruským** prezidentem **Bronislawem** Komorowským varoval, že situace v Bělorusku může mít dopad na celou střední a východní Evropu. Obama také jednal s polskými disidenty, kteří se zasloužili o pád komunismu v Polsku.

Figure 1.1: Example of model hallucination. Red words are hallucinated or wrongly inferred from the source document. English translation can be found in Figure B.1.

from the development split of the SumeCzech dataset [Straka et al., 2018], and the summary is generated using a model from [Krotil, 2022]. We can see that the model wrongly inferred the nationality of Mr. Komorowski (Belarusian instead of Polish) and hallucinated his first name, as it was not stated in the source document. It can sometimes happen that the hallucinated content is factual, like in this example – Bronisław Komorowski really was the Polish president when the article was released. However, we cannot efficiently check this, so we have to assume that the hallucinated content is not factual.

Chapter 2

Pre-trained language models

Model pre-training is an efficient way of training large language models. Pre-training is an unsupervised procedure during which the model learns to capture various knowledge from text corpus. These models are usually not very usable on their own, but they can be further fine-tuned on a downstream task.

Fine-tuning means training the model with its weights initialized from the pre-trained model on some specific task with a small learning rate. It typically requires only a fraction of the training steps compared to pre-training. For this reason, it is much more computationally efficient than training the model from scratch. Training task-specific models from scratch would be computationally comparable to pre-training, but we only have to pre-train once, and then we can reuse the model multiple times.

All the pre-trained language models follow one of the following paradigms:

- **Encoder** – These models aim to generate a single vector representation of the input sequence. This type of model is best suited for tasks like text classification.
- **Decoder** – At each time step, decoder-based models are fed the output from the previous step and all steps before. Decoders also output their hidden state at each timestep, which is why they are used in tasks concerning text generation. We usually provide the model with an initial input, also called a prompt, to give the model additional information on which it can base its output. Decoder models are also called autoregressive.
- **Encoder-Decoder** – Also called sequence-to-sequence, these models combine the previous two architectures. The representation obtained from the encoder is fed into the decoder as the first hidden state, and the decoder then follows to generate in an autoregressive manner. This architecture is ideal for conditional text generation.

This chapter will first introduce the Transformer architecture [Vaswani et al., 2017] used in all pre-trained language models, and then we will review a few of these, which will be used later in this work.

2.1 Transformer architecture

The Transformer architecture [Vaswani et al., 2017] is the most used model architecture in almost all natural language processing (NLP) tasks. The goal of this architecture was to reduce the computational cost of sequential models like recurrent neural networks. The Transformer delivered and made all of the current state-of-the-art results in NLP.

The main building block of the Transformer model is the self-attention function. This function is used to provide additional context for each position of the input sequence.

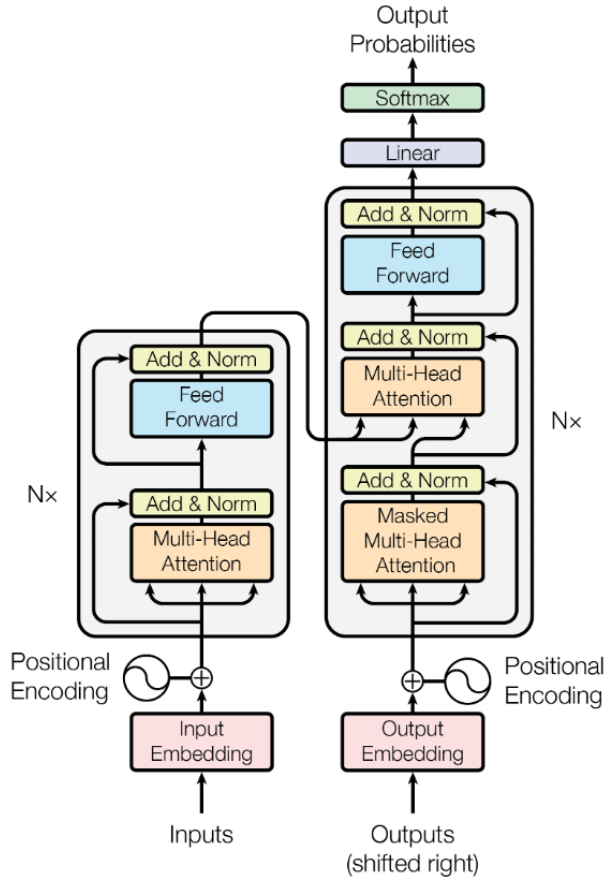


Figure 2.1: Transformer architecture, taken from [Vaswani et al., 2017].

When making predictions, it can be used to "attend" different parts of the sequence. This way, it mitigates the problem of long-range dependencies from which traditional sequential models suffer.

The self-attention's input is matrices of queries Q , keys K of dimensionality d_k , and values V of dimensionality d_v . Given the matrix of inputs X , we can obtain these matrices as follows:

$$\begin{aligned} Q &= X \times W^Q \\ K &= X \times W^K \\ V &= X \times W^V \end{aligned} \quad (2.1)$$

where W^Q , W^K , W^V are trainable matrices of weights. The self-attention is then given by:

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.2)$$

In practice, we compute the attention on the same input multiple times in parallel with different matrices Q , K , and V . These outputs are then concatenated and projected once more to create the final output. This operation is called Multi-Headed Attention.

The transformer model is based on the encoder-decoder model. The encoder contains multi-headed attention layers, each followed by a normalization layer, feed-forward layer,

and residual connections. These blocks are stacked multiple times. On top of that, the decoder contains masked self-attention to preserve autoregressiveness. Masked self-attention is implemented by setting all input values of the softmax in self-attention corresponding to unwanted connections to $-\infty$. The final model architecture is shown in Figure 2.1.

Lastly, each input token is summed with positional embeddings before feeding it to the model to preserve its positional information. These embeddings can be either learnable or fixed.

2.2 BERT

Bidirectional Encoder Representations from Transformers (BERT) [Devlin et al., 2018] is a Transformer-based encoder model which achieved new state-of-the-art results in multiple NLP tasks, such as natural language understanding, natural language inference, and question answering at the time of its release.

The architecture of BERT’s Transformer blocks is identical to the encoder part of the Transformer introduced in [Vaswani et al., 2017]. The model can handle both single sentences and sentence pairs. To create the input representation, the authors used the WordPiece tokenizer [Wu et al., 2016]. A special [CLS] token is prepended to each input sequence. This token’s last hidden state vector is then used for classification tasks. Sentence pairs are handled as a single sentence. To differentiate between the two sentences, they are separated using a special [SEP] token. The final representation of each token is then obtained by summing it with its positional embedding and segment embedding, indicating if the token belongs to the first or second sentence.

Before BERT, models were pre-trained to learn the language representation in a left-to-right or right-to-left manner using only the left or right context, not utilizing the bidirectional nature of Transformer models. The authors introduced a new bidirectional pre-training paradigm divided into two unsupervised subtasks:

- **Masked Language Modeling (MLM)** – For the model to learn bidirectional language representation, some tokens of the input text are randomly masked and then predicted.
- **Next Sentence Prediction (NSP)** – In order to learn the relationship between sentences, the model is given two sentences and predicts whether or not the second sentence follows the first one.

2.3 BART

The BART model [Lewis et al., 2019] is representative of the encoder-decoder model architecture. On its release, it achieved new state-of-the-art results in numerous text generation tasks, including abstractive text summarization.

It uses the same architecture for the bidirectional encoder and autoregressive decoder as the original Transformer model [Vaswani et al., 2017]. The only notable modification is in its choice of the activation function. The original uses ReLU, while the BART model uses GeLU [Hendrycks and Gimpel, 2016].

The main innovations of BART lie in its pre-training schema. The authors experimented with several pre-training objectives, including some previously introduced ones, e.g., Masked Language Modeling, and some novel ones. They did a comprehensive comparison and ablation study on how different objectives affect pre-training and the following

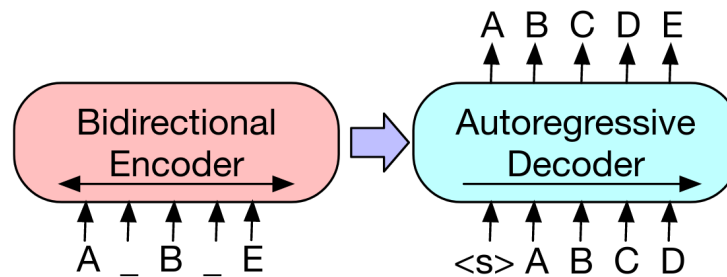


Figure 2.2: Simplified architecture of BART, taken from [Lewis et al., 2019].

fine-tuning. It was found that the effectiveness of a pre-training objective is highly dependent on the task the model is later fine-tuned on. Based on these results, the final model was pre-trained using a combination of two tasks:

- **Text Infilling** – Similar to the MLM objective, but instead of masking only individual tokens, whole text spans are masked with a single [MASK] token and then predicted.
- **Sentence Permutation** – The order of sentences is randomly permuted, and the model tries to predict its original order. This helps the model to learn the relationships between sentences.

Figure 2.2 shows a simplified architecture of BART and the text infilling pre-training objective. The masked document is an input for the bidirectional encoder, and the decoder attempts to predict the original document.

2.4 mBART

The language of data used to pre-train BART was only in English, and thus BART can only be fine-tuned on tasks in English. mBART [Liu et al., 2020] is a multilingual version of BART pre-trained on text corpus consisting of 25 languages, and its main focus was machine translation.

It has the same architecture as BART and even follows the same pre-training scheme. The two models differ only in the input format. mBART appends a special language ID token <LID> to the encoder input to distinguish between different languages. The <LID> of the target language is also used as an initial token for the decoder to predict the sequence. Each sentence is additionally separated using the end of the sentence token </s>.

The model managed to set new state-of-the-art results in machine translation between several languages and demonstrated that multilingual pre-training improves machine translation performance, especially in low-resource language pairs.

2.5 RoBERTa

RoBERTa [Liu et al., 2019] is a model with architecture identical to BERT's. Its authors repeated the study on BERT's pre-training objectives, hyperparameter setting, and training set size and discovered that BERT is significantly undertrained. They found that training the model longer with a much larger batch size over a bigger training set improved its performance significantly. The only changes in the pre-training objectives were

to remove the next sentence prediction and change the mask placement in the masked language modeling objective dynamically during training.

■ 2.6 XLM-RoBERTa

XLM-RoBERTa [Conneau et al., 2019] (XLM-R) is a version of RoBERTa pre-trained on a large multilingual corpus (2.5 TB) containing 100 different languages. The input specification was not changed for XLM-R; thus, the model can distinguish between different languages on its own. It achieved significant improvements on several downstream tasks, especially those in low-resource languages.

Chapter 3

Summarization methods

There are two approaches to automatic text summarization – extractive and abstractive. **Extractive** summarization is the task of choosing the most important sentences from the input text to form a summary. Most methods for solving this task are unsupervised, but we will not deal with these in this work because supervised methods are usually superior when it is possible to use them. However, all extractive methods follow the same two subtasks while generating a summary:

1. **Sentence scoring** – The importance of sentences is usually decided using some scoring function measuring the relevance of each sentence.
2. **Sentence selection** – After scoring each sentence, we will select a few with the highest score to form the summary.

Sentence scoring methods range from straightforward ones, like considering only word frequency, to more complex ones incorporating Transformer-based encoders. Nevertheless, all methods should consider the whole input text when scoring individual sentences in order to be usable. While selecting the most important sentences, we should check whether the sentence we are adding to the summary is not too similar to the ones already selected to avoid redundant information in summary.

Abstractive summarization has the same goal as a human has when summarizing text – to write a brief, coherent summary of the input text. The resulting summary should capture all the critical information but may also contain new words not previously seen in the text. This type of summarization is more challenging than the extractive as it requires the model to have a deep understanding of language and its structure.

Abstractive summarization is the ultimate goal of text summarization. The extractive summarization does not have to be meaningful, even in the best-case scenario.

This chapter will review a few state-of-the-art methods for solving both extractive and abstractive text summarization. In the last section, we will also look into the inference methods of sequence-to-sequence models, as it is an important part of the summarization process.

3.1 BertSum

BertSum [Liu and Lapata, 2019] is a model which uses the BERT model introduced in Section 2.2 for extractive summarization. In order to fine-tune BERT for extractive summarization, we need to create a representation for each sentence. The authors achieved this by adding the special [CLS] token to the beginning of each sentence. Before tokenization, the input is split into individual sentences, and the [CLS] token is prepended

to each sentence. The modified sentences are then concatenated, and this new input is finally tokenized. Each [CLS] token represents the whole sentence and is used to predict whether it should be included in the resulting summary or not. In addition, every token is assigned a segment embedding E_A or E_B depending if the token is from an even or odd sentence. The segment embedding is added with the position and token embedding to form the model’s input.

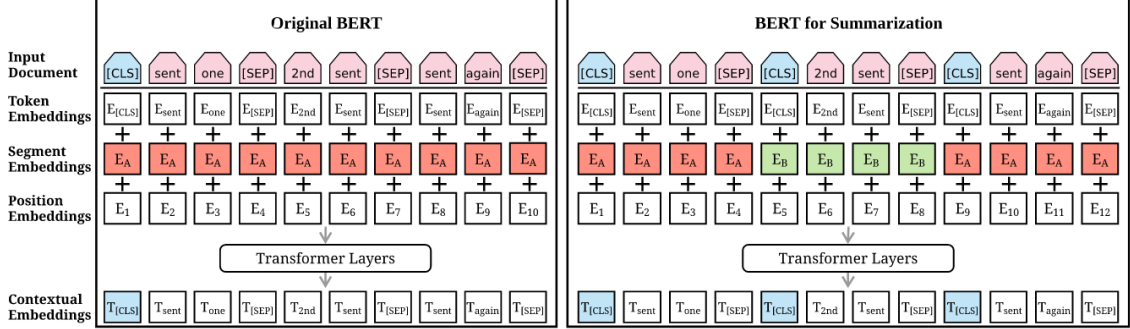


Figure 3.1: BERT and BertSum architecture, taken from [Liu and Lapata, 2019].

Because the maximum number of position embeddings in BERT is 512, the authors expanded the positional embedding layer. The first 512 pre-trained embeddings were copied, and the rest of the embeddings were initialized randomly. This way, the model can process longer inputs.

Finally, a classifier module with sigmoid activation is stacked on top of BERT’s output. The authors used several Transformer blocks as a classifier. A comparison of BERT’s and BertSum’s architecture is depicted in Figure 3.1. Except for the classifier and the extended embedding layer, there are no other differences compared to BERT’s architecture. The main differences lie in the input specification. The model is trained using binary cross-entropy loss between the predictions and labels.

3.2 BART for summarization

Authors of the BART model presented in Section 2.3 already experimented with fine-tuning it to abstractive summarization task and achieved excellent results.

The fine-tuning procedure is relatively straightforward. Given source document D and reference summary S^* , we input D into the encoder part of the model to create its internal representation. The model is then trained using the so-called teacher forcing algorithm. The decoder always predicts the next token conditioned on all previous tokens in the reference summary and the source document. The model’s parameters θ are then updated, optimizing the cross-entropy loss

$$L_{xent} = - \sum_{j=1}^l \sum_s [s = s_j^*] \log p(s | D, S_{<j}^*; \theta) \quad (3.1)$$

where $S_{<j}^*$ is a subsequence of the reference up to the j -th token $\{s_0^*, s_1^*, \dots, s_{j-1}^*\}$ and s_0^* is starting token.

Summarization in languages other than English is also possible using BART’s multilingual version, mBART. It was shown, for example, in [Krotil, 2022] where the model was fine-tuned for the abstractive summarization of Czech news articles.

Fine-tuning of mBART is done similarly as in the previous case. However, the inputs should be correctly preprocessed to match mBART’s input specifications. Because the primary goal of mBART was machine translation, the model expects the source language ID at the end of the encoder’s input and the target language ID as the first token in the decoder. In the case of summarization, both inputs are in the same language; thus, the source and target language IDs are the same.

While generating summaries, the fine-tuned model predicts autoregressively. It is important to note that this differs from the fine-tuning objective because now when predicting the next token, the model conditions on all of its generated tokens and not tokens in the reference summary. Once the generated subsequence deviates from the reference, the generated summary’s overall quality might steeply degrade. This problem is known as exposure bias.

3.3 BRIO

BRIO [Liu et al., 2022] is a novel training paradigm of abstractive summarization models. Until now, models were trained using the maximum likelihood estimate (MLE), maximizing the probability of the next token given previous tokens of the reference summary. In contrast, these models generate new summaries autoregressively during inference, conditioning each token on its previous predictions. This new method tries to reduce the discrepancy between training and inference. The motivation is that model should be able to assign a higher probability to better summaries according to some metric because there always exist more possible summaries of a given document. To force the model to assign a higher probability to summaries of higher quality, the authors use contrastive loss defined as:

$$L_{ctr} = \sum_i \sum_{j>i} \max(0, f(S_j) - f(S_i) + \lambda_{ij}) \quad (3.2)$$

where S_i and S_j are two different candidate summaries satisfying $M(S_i, S^*) > M(S_j, S^*)$ for reference summary S^* and some quality measure M . λ_{ij} is the difference in rank multiplied by the margin hyperparameter λ , i.e. $\lambda_{ij} = (j - i) * \lambda$. And finally, $f(S)$ is the estimated log-probability normalized by the length of the summary

$$f(S) = \frac{\sum_{t=1}^l \log p(s_t | D, S_{<t}; \theta)}{|S|^\alpha} \quad (3.3)$$

with α being the length penalty hyperparameter, D the input document, $S_{<t}$ the summary up to the t -th token and θ the model’s parameters.

However, the model trained using only the contrastive loss would not be able to generate summaries. Thus, the authors combined the cross-entropy and contrastive loss into a multi-task loss L_{mul} .

We can summarize the complete workflow of the BRIO method into three individual steps:

1. **Initial fine-tuning** – The pre-trained model is fine-tuned for abstractive summarization, optimizing the cross-entropy loss. This is the same procedure as described in section 3.2. Let us call this model the base model.
2. **Candidate generation** – Multiple candidate summaries are generated using the base model. To generate more diverse candidates, the authors used an inference method

called diverse beam search [Vijayakumar et al., 2016] and generated 16 candidates for each sample. These candidates remain fixed for the rest of the training.

3. **Multi-task fine-tuning** – The base model is then further fine-tuned using the multi-task loss

$$L_{mul} = L_{xent} + \gamma L_{ctr} \quad (3.4)$$

where L_{xent} is the standard cross-entropy loss as in Equation 3.1 calculated for the reference summary, and γ is the weight of the contrastive loss.

The advantage of this approach is that it is independent of the quality measure M , so we can choose to maximize any metric we want. On the other hand, we need to have a strong base model to generate high-quality candidates.

In the paper, the authors used the fine-tuned BART model as their base model for multi-task fine-tuning. This model significantly improved the state-of-the-art results over the base model.

3.4 Inference of sequence-to-sequence models

During inference of sequence-to-sequence models, we typically aim to maximize the probability of the generated text given the source document. The text is generated autoregressively, meaning the model conditions the probability of the next token on the source document and the previously generated tokens. There are many decoding strategies for text generation, so we will describe only a few most commonly used ones.

3.4.1 Greedy search

The greedy search generates the next token maximizing the probability given the source document and the past tokens at each time step, regardless of what tokens might be generated in the future. The next token w_t at time t given the source document D is given by:

$$w_t = \operatorname{argmax}_w P(w|w_{1:t-1}, D) \quad (3.5)$$

This strategy might be locally optimal at each timestep but gets more suboptimal globally as we generate longer sequences.

3.4.2 Beam search

Beam search is probably the most commonly used decoding method for autoregressive text generation. It considers more candidate sequences at each time step and eventually picks the one having the highest overall probability.

This method is characterized by a single hyperparameter k indicating the number of candidate sequences considered at each time step. These candidate sequences are usually called beams. Initially, we select k tokens having the highest probability. Each of these will be the beginning of the k candidate sequences. In the subsequent time steps, we always consider the most probable k next tokens overall across all beams and keep the top k sequences with the highest combined probability as our k beams for the next step. We continue like this until we reach the end-of-sequence token or satisfy some other condition, e.g., sequence length. Finally, we select the most probable sequence as our output.

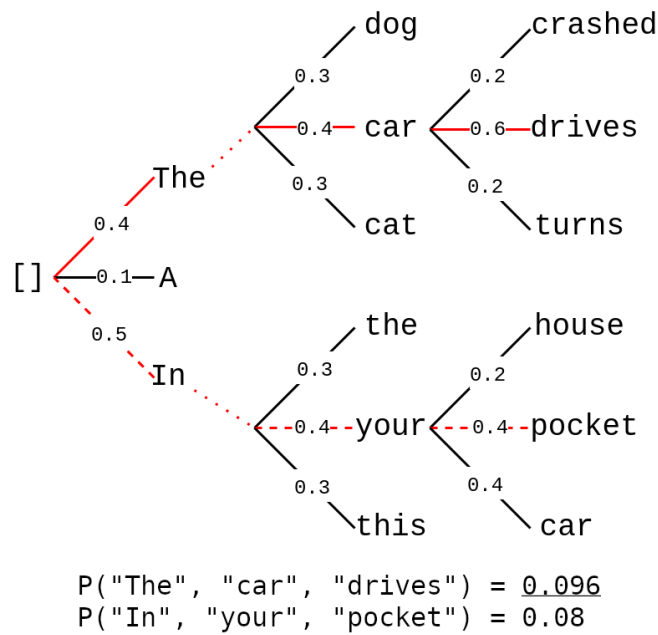


Figure 3.2: Beam search illustration with beam size equal to 2.

Figure 3.2 illustrates decoding using beam search on a simple example with beam size 2. While greedy search would pick the lower branch of the search $\{\text{In, your, pocket}\}$, beam search finds a more probable sequence by also considering a less probable word in the beginning, resulting in the output $\{\text{The, car, drives}\}$.

3.4.3 Sampling

Beam search often leads to generating repetitive texts, or the text tends to be incoherent or too generic. This is because the human text does not follow the high probability distribution of the next tokens as the beam search [Holtzman et al., 2019]. Human text is way more surprising in terms of the probability of the next tokens than the maximization-based decoding methods.

To address this issue, we can introduce randomness into the decoding process via sampling. The two most commonly used sampling-based methods are top-k sampling and nucleus sampling.

■ **Top-k sampling** is a simple sampling-based method. At each decoding step, we select top-k tokens based on their conditional probability. The distribution of these k tokens is re-scaled, and sampling is performed based on this distribution. The problem with this method is the difficulty of choosing the right k . A small value of k might lead to the same problem as maximization-based methods, and too large k could include inappropriate tokens in the generated text.

■ **Nucleus sampling** solves the problem of choosing the right k in top-k sampling by sampling from a set of tokens with variable size. The size of this set is given by the predefined threshold p . At each time step, we select the smallest set of the most probable tokens whose cumulative probability is higher or equal to p . Their distribution is then re-scaled, and the next token is sampled based on this distribution.

3.4.4 Diverse beam search

Another problem arises with beam search when we need to generate multiple diverse sequences. Beam search tends to generate the same or nearly identical text every run. Diverse beam search [Vijayakumar et al., 2016] solves this issue by splitting the beams of a beam search into equal-size groups. Each group then performs a smaller beam search with a beam size equal to B/G , where B is the number of beams and G is the number of groups. The probability of within-group sequences is affected by other groups using a special dissimilarity term to enforce more diverse sequences.

The decoding starts with a step of a regular beam search for the first group. The other groups' probabilities for each token in each sequence are modified using diversity term and diversity penalty hyperparameter λ . For each group g at time step t , the logarithm of probability for token w_t^g given document D is modified as follows:

$$\log(P(w_t^g|w_{1:t-1}^g, D)) = \log(P(w_t^g|w_{1:t-1}^g, D)) + \lambda \sum_{h=1}^{g-1} \Delta(w_t^g, W_t^h) \quad g \in \{2, \dots, G\} \quad (3.6)$$

Where $\Delta(w_t^g, W_t^h)$ measures the dissimilarity between token w_t^g and tokens W_t^h from group h generated at time t . For example, if group h consists of 3 beams, we would compare w_t^g with the three tokens generated in group h at time t . The dissimilarity measure could be arbitrary - e.g., a negative cost for each co-occurring token in the two sequences. The diversity penalty λ controls the diversity of the solutions. The higher it is, the more diversity is enforced, with λ equal to 0 means no diversity.

The beam search in each group then continues as the regular beam search with the modified token probabilities. The output is G sequences, one from each group.

Chapter 4

Evaluation metrics

Automatic evaluation of generated summaries is still not a fully solved task. There exist a few automated metrics that are commonly used for summary evaluation. However, these metrics still have some shortcomings. Current methods are usually based on word overlaps between the reference and generated summary, or they measure the cumulative similarity of tokens using some embedding function and similarity measure. The main disadvantage of such methods is that they evaluate the generated summary with the reference summary, not the source document, as there can be multiple valid summaries. We will present two commonly used metrics for assessing the quality of summaries independent of the source language and two metrics crafted specifically for the Czech language.

4.1 ROUGE

ROUGE [Lin, 2004] is a collection of metrics for automatic summary evaluation. It is a simple overlap-based method that is fast and computationally cheap. Due to this and its interpretability, it is the traditional metric for automatic summary evaluation that moderately correlates with human judgment [Zvára, 2022].

The two most used ROUGE metrics are ROUGE-N and ROUGE-L.

- **ROUGE-N** measures the n -gram overlap between the reference and generated summary. The n -gram represents n consecutive words from the given text. Moreover, we usually calculate n -gram precision P , recall R , and the corresponding F1-score. Given n -grams in the reference W_{ref} and n -grams in the system summary W_{sys} , these measures are calculated as follows:

$$R = \frac{|W_{ref} \cap W_{sys}|}{|W_{ref}|} \quad P = \frac{|W_{ref} \cap W_{sys}|}{|W_{sys}|} \quad F1 = 2 * \frac{R \cdot P}{R + P} \quad (4.1)$$

The most common choice for N is 1 or 2, i.e., unigram and bigram overlaps.

- **ROUGE-L** evaluates the summary by calculating the longest common subsequence (LCS) with the reference. The LCS is obtained using dynamic programming. Given system-generated summary X of length m and reference Y of length n , then the precision P , recall R , and F1-score are given by:

$$R = \frac{LSC(X, Y)}{n} \quad P = \frac{LSC(X, Y)}{m} \quad F1 = 2 * \frac{R \cdot P}{R + P} \quad (4.2)$$

4.2 BERTScore

BERTScore [Zhang et al., 2019] compares two texts by computing pairwise similarities between individual tokens, but instead of exactly matching them like ROUGE, it uses contextual embeddings of the tokens. These embeddings can also consider the context of the words, i.e., the same word can have different embedding depending on its placement in the text.

The contextual embedding of each token is obtained from some pre-trained model by feeding it the tokenized text and then taking the hidden states of one of the encoder layers as the token embeddings.

Pairwise similarities are computed between the token embeddings of the reference and the candidate. The cosine similarity is used as a similarity measure. The embedding vectors are normalized before computing the similarities, so the computation of the similarity of token x_i and x_j reduces to $x_i^T x_j$.

The final BERTScore greedily matches each token in the reference text x to the most similar token in the candidate text y . We can express the score in terms of recall, precision, and F1-score.

$$R = \frac{1}{|x|} \sum_{x_i \in x} \max_{y_j \in y} x_i^T y_j \quad P = \frac{1}{|y|} \sum_{y_j \in y} \max_{x_i \in x} x_i^T y_j \quad F1 = 2 \frac{P \cdot R}{P + R} \quad (4.3)$$

The authors of BERTScore experimented with several pre-trained models as the source of contextualized embeddings and chose the model and model’s layer based on the correlation with human judgment. It is important to note that the model-layer selection was based on a machine translation dataset, so its performance in summarization evaluation might be suboptimal.

4.3 ROUGE-CS

ROUGE is well-suited for English as it does not have many different word forms as some other languages like Czech. It can happen that even semantically identical sentences with slightly different wording may achieve very low ROUGE scores. ROUGE-CS [Zvára, 2022] is a modification of classic ROUGE-N, aiming to maximize correlation with human judgment.

The two texts are compared by first tokenizing the two texts in the same manner as ROUGE, i.e., word by word. Stop-words are removed, and the remaining words are lemmatized. The next step is to generate n-grams from the texts and compare them pairwise. This comparison is based on Word2Vec [Mikolov et al., 2013] embeddings vector similarity. The similarity is counted negatively if the two words are identified as antonyms or the similarity is lower than some predefined threshold. Then we can use the reference n-grams W_{ref} , the system n-grams W_{sys} , and the accumulated similarity to compute recall, precision, and F1-score.

$$R = \frac{similarity}{|W_{ref}|} \quad P = \frac{similarity}{|W_{sys}|} \quad F1 = 2 \frac{similarity}{|W_{ref}| + |W_{sys}|} \quad (4.4)$$

4.4 Memes-CS

Memes-CS [Zvára, 2022] is a model-based metric for automatic summary evaluation against the reference summary. It is built on the XLM-R model fine-tuned on SQuAD 2.0 question-

answering dataset [Rajpurkar et al., 2018]. This checkpoint of the model was further fine-tuned on a combination of two datasets. One is CSFEVER [Ullrich et al., 2023] – dataset for automated fact-checking, and the second is the Grad Cortex dataset developed algorithmically by the model author by systematically corrupting the reference summaries.

The model was stacked with a sigmoid activation on top of its output and was trained by optimizing the binary cross-entropy loss. Its output is a single number between 0 and 1, representing the similarity of the summaries. This model-based metric surpassed ROUGE and ROUGE-CS in correlation evaluation against human-annotated summary pairs.

Chapter 5

Datasets

This work focuses on the summarization of Czech news articles. We chose news articles because they are usually written in the form of a short summary followed by the article’s main text. This gives us ideal training samples for automatic summarization models, with text as the model’s input and summary as the model’s target. Another convenience is that we have access to many already-processed and somewhat clean news articles. This chapter will describe two datasets that we will combine and use as our main training corpus.

5.1 SumeCzech

SumeCzech [Straka et al., 2018] is a large Czech news-based summarization dataset consisting of over one million news articles. The dataset was collected from the CommonCrawl¹, and the authors provide a script for downloading the dataset using CommonCrawl API and cleaning it using several simple rules. More details on the data cleaning can be found in the original paper. Table 5.1 shows sources of data.

Website	Num. documents	Ratio
<i>ceskenoviny.cz</i>	854	0.5%
<i>denik.cz</i>	157 581	15.7%
<i>idnes.cz</i>	463 192	46.2%
<i>lidovsky.cz</i>	136 899	13.7%
<i>novinky.cz</i>	239 067	23.9%
Total	1 001 593	

Table 5.1: Sources of the SumeCzech dataset, taken from [Straka et al., 2018].

Dataset split	Num. documents
train	867 596
dev	44 567
test	44 454
out-of-domain test	44 976
Total	1 001 593

Table 5.2: Statistics of the SumeCzech dataset splits.

The dataset is provided in JSON² Lines format. Each example from the dataset consists

¹<http://commoncrawl.org/>

²<http://jsonlines.org/>

of multiple fields. The fields relevant for us are headline, abstract, and text. The headline is typically one sentence long, the abstract is a few sentences long summary of the article, and the text field is the main text. We used the same dataset splits for training, validation, and testing, as suggested in the paper. The authors also created an out-of-domain test split, where the articles are from different domains than the rest of the dataset. This part of the dataset was found using KMeans clustering. The sizes of these splits can be seen in Table 5.2. Each of the splits has roughly the same coverage of the sources.

5.2 CNC

This dataset was obtained from the Czech News Center (CNC) company³. It is one of the largest media houses in Central Europe, producing multiple popular newspapers in the Czech Republic. It was supplied by the supervisor, who is cooperating with the CNC. This dataset is not publicly available, but we will use it to train our models.

The raw dataset was filtered using a script from [Krotil, 2022]. The techniques to clean the data are similar to the ones used to clean the SumeCzech dataset.

Dataset split	Num. documents
train	675 225
dev	35 000
test	35 000
Total	745 225

Table 5.3: Statistics of the CNC dataset splits.

Each document contains the same three fields as the SumeCzech – headline, abstract, and text. We split the dataset into training, testing, and development splits. There is no information on the exact sources of each document, like in SumeCzech’s case. However, the number of documents reserved for each split and the total number of documents in the dataset is shown in Table 5.3.

5.3 Final training dataset

We used a combination of the training splits of the two previously introduced datasets as our models’ training data. In total, our training dataset has **1,542,821** training examples. The other splits were left as they were. We will often use only a part of the dataset, or we will transform the dataset. The details of the data used will always be provided in relevant sections.

³<https://cncenter.cz>

Chapter 6

Methodology

We have selected multiple methods that might improve the factual consistency of generated summaries. This chapter will briefly describe the selected methods that will be implemented later in this work. We will use one extractive method and two abstractive methods.

6.1 Extractive method

The main reason why we are going to experiment with extractive summarization methods is that extractive models cannot hallucinate and thus should produce only summaries consistent with the source document. On the other hand, extractive summaries are often incoherent because the sentence selection procedure picks the sentences throughout the whole document. We will address the problem with incoherent summaries later in this section using a sequence-to-sequence model trained to rewrite extractive summaries. Extractive methods are also limited to only picking a couple of sentences, so they generally cannot capture all the information from the document.

6.1.1 Model

Our model is inspired by the architecture of BertSum [Liu and Lapata, 2019]. BertSum uses BERT as its backbone model. BERT is a model pre-trained on English-only corpora, so we had to choose a different pre-trained encoder-based language model. While we could use BERT’s multilingual version mBERT released by the authors of BERT [Devlin et al., 2018], we decided to use XLM-R introduced in Section 2.6 as it showed excellent results in various downstream tasks [Conneau et al., 2019].

Same as BertSum, we added a classification token at the beginning of each sentence in order to collect features for this sentence. Our model uses the `<s>` token as the classification token, while BERT uses the `[CLS]` token. XLM-R also does not use segment embeddings because it was not pre-trained with the next-sentence prediction task as BERT, so we did not implement the alternating segment embeddings which were used in BertSum.

Finally, we stacked the classifier module with sigmoid activation on top of the model’s output. Instead of the transformer block used in BertSum as a classifier, we used a simple linear layer because the authors of BertSum achieved comparable results to the transformer-based classifier using a linear layer [Liu, 2019] and because of its simplicity.

Figure 6.1 shows the simplified architecture of our extractive model. The input is first tokenized to match the model’s input specification. We first place the `<s>` classification token at the beginning of each sentence and the end-of-sequence token `</s>` at the end. The input is then tokenized and fed into the XLM-R model. Lastly, the outputted contex-

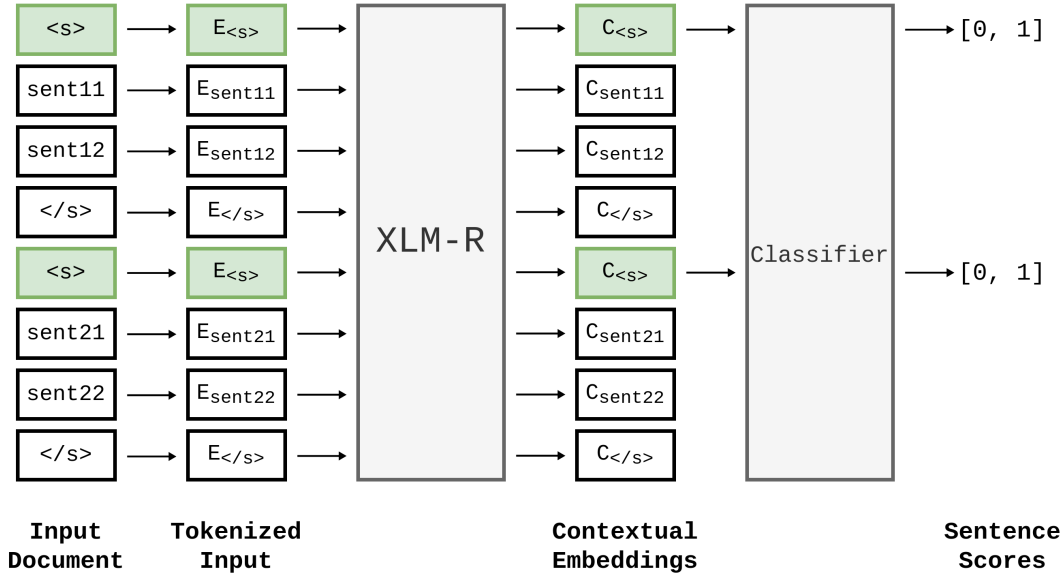


Figure 6.1: Overview of our extractive model architecture.

tual embeddings corresponding to the classification tokens are fed to the classifier. The classifier’s output is a score in the range $[0,1]$ corresponding to each sentence.

6.1.2 Rewriting extractive summaries

A simple idea of dealing with incoherent extractive summaries is to train a model that will rewrite it to a more coherent abstractive summary. Following [Zhang et al., 2023], we will build a dataset consisting of source-target pairs, where the source will be short incoherent text, and the target will be coherent text corresponding to the source. We will describe the exact procedure of building this dataset in Chapter 7.

After we have gathered the data, we will train a sequence-to-sequence model to predict the targets given the source as an input. We decided to use mBART for this task as it is one of the best-performing multilingual encoder-decoder-based models.

This method may improve the coherence of the extractive summaries, but it can also hurt the facticity of the summary. Using a sequence-to-sequence model, we may again bring hallucinations into the summary.

6.2 Abstractive methods

Abstractive summarization methods are still far from perfect, and that is why we mainly focused on them. Abstractive summaries should capture all the information and be coherent and pleasant to read. Unlike extractive methods, abstractive summaries can potentially cover all the critical information within the source document because they are not limited to picking only certain parts of the document.

6.2.1 BRIO-based methods

BRIO, the new method for fine-tuning pre-trained language models, has shown great improvements in terms of ROUGE scores. The authors also tried maximizing BERTScore,

which was also a success, and showed that BERTScore correlates with ROUGE. We hypothesize that maximizing such metrics could also lead to improvements in factual consistency.

The BRIO method requires an already fine-tuned base model for abstractive summarization in order to generate candidate summaries, detailed description of this method is in Section 3.3. The authors experimented with two base models. One of them was the BART model introduced in Section 2.3, fine-tuned on the CNN-DailyMail dataset [Hermann et al., 2015], and the Pegasus model [Zhang et al., 2020] fine-tuned on the XSum dataset [Narayan et al., 2018].

We chose a model HT2A-CS¹ fine-tuned by [Krotil, 2022] as our base model as it is fine-tuned on the same datasets we use. The model’s name represents the task and the dataset it was trained on. HT2A means that the model was trained to predict the abstract of the news article given the concatenation of headline and text. CS stands for the combination of the CNC and the SumeCzech dataset. This model is fine-tuned version of mBART described in Section 2.4. In the next chapter, we will apply the BRIO method to the base model while maximizing various metrics through contrastive loss (Eq. 3.2).

¹<https://huggingface.co/krotimal1/mbart-ht2a-cs>

Chapter 7

Experiments

The aim of this chapter is to present the experiments conducted to evaluate the effectiveness of suggested methods for improving the factual consistency of generated summaries. We will also provide details on the workflow of each selected method from the previous chapter. Through analyzing the chosen methods, we intend to assess the extent to which they enhance factual consistency. Additionally, we will examine the advantages as well as limitations of each method.

Initially, we will provide brief implementation details of our solutions, and then we will explore an alternative approach to automatic summary evaluation. Subsequently, we will provide a comprehensive description of the data preparation and training procedures for each of the suggested methods. Finally, we will present the results of both automatic and human evaluations, which will give insights into the proposed methods' performance and how they compare with other existing techniques.

7.1 Implementation details

We used the Python programming language with well-known libraries to implement the proposed methods. Namely, for vector and matrix calculations, we predominantly used NumPy¹ and PyTorch². For data analysis and plotting, we used a combination of Pandas³ and Matplotlib⁴. The data were preprocessed mainly using the HuggingFace datasets library⁵, which provides functions for fast data preprocessing using multiprocessing. To track the training process, we used the Weights & Biases platform⁶.

All the pre-trained language models used in this work are from the HuggingFace transformers library⁷. This library provides a convenient and intuitive interface for working with these models. The models are fully compatible with PyTorch and can be used as a standard PyTorch module. The library also provides several classes for model fine-tuning, making it very easy to start the training.

7.1.1 Tokenization

To tokenize the input of our models, we used the `PreTrainedTokenizer` class from HuggingFace transformers corresponding to the particular pre-trained model. In our work, we

¹<https://numpy.org/>

²<https://pytorch.org/>

³<https://pandas.pydata.org/>

⁴<https://matplotlib.org/>

⁵<https://github.com/huggingface/datasets>

⁶<https://wandb.ai>

⁷<https://github.com/huggingface/transformers>

utilized only two pre-trained models – mBART and XLM-R. Both of these models use the SentencePiece model [Kudo and Richardson, 2018] for tokenizing their input.

SentencePiece is an unsupervised text tokenizer that is trained directly from raw sentences. During tokenization, it considers spaces as regular characters, unlike some other tokenizers [Sennrich et al., 2015], [Kudo, 2018]. This fact makes decoding tokenized text very easy, as we can just concatenate all the tokens together.

7.2 Source-based evaluation

We argue that the best way to evaluate the quality of a generated summary is to compare it with the source document and not the reference summary, as most previous methods do. There can be multiple correct summaries of the same document, so reference-based evaluation can sometimes be misleading.

There are more reasons why source-based evaluation is a better way of evaluating summaries. We can consider the source document as an objective representation of the information contained in it, whereas a reference summary is a subjective interpretation of that information. As a result, evaluating the summary with the source could eliminate the potential bias present in the reference. Reference summaries may also not capture all the information from the source document. By evaluating with source, we ensure that it contains all relevant information.

We will revisit two automatic metrics for summary evaluation introduced in Chapter 4 and show their potential use in source-based evaluation. We will then evaluate these metrics on a human-annotated dataset.

7.2.1 Annotated dataset

We collected document-summary pairs with summaries generated using multiple different models with human annotations measuring the factual quality of each summary based on comparison with the source document. As an annotation platform, we used an open-source text annotation tool called doccano [Nakayama et al., 2018].

Originally we used the following labels for each document-summary pair:

- **Perfect** – The summary is factual and grammatically correct
- **Sufficient** – The summary is factual with minor grammatical errors
- **Incorrect** – Factually inconsistent summary

Because we focus on improving the factual consistency of generated summaries, we merged the Perfect and Sufficient summaries under one label. Specifically, we used label **1** for Perfect/Sufficient summaries and label **0** for Incorrect summaries.

In total, we collected 445 annotated pairs. However, most of them ended up with label 0, so we only kept the same number of not-factual pairs as factual ones. Finally, we were left with 274 annotated pairs with balanced numbers of each label.

7.2.2 BERTScore revisited

The original BERTScore, introduced in the paper by [Zhang et al., 2019] and its official implementation⁸, was optimized for machine translation. The authors conducted a model-layer search in order to find a combination of a model and its layer, which maximizes the

⁸https://github.com/Tiiiger/bert_score

correlation of BERTScore with human-annotated data. The recommended model to use for the Czech language is bert-base-multilingual-cased⁹ and layer number 9.

BERTScore computes recall, precision, and F1-score. We will only consider precision when using it for source-based evaluation because, using recall or F1-score, we would penalize shorter summaries by dividing the embeddings’ similarity with the size of the document (Equation 4.3).

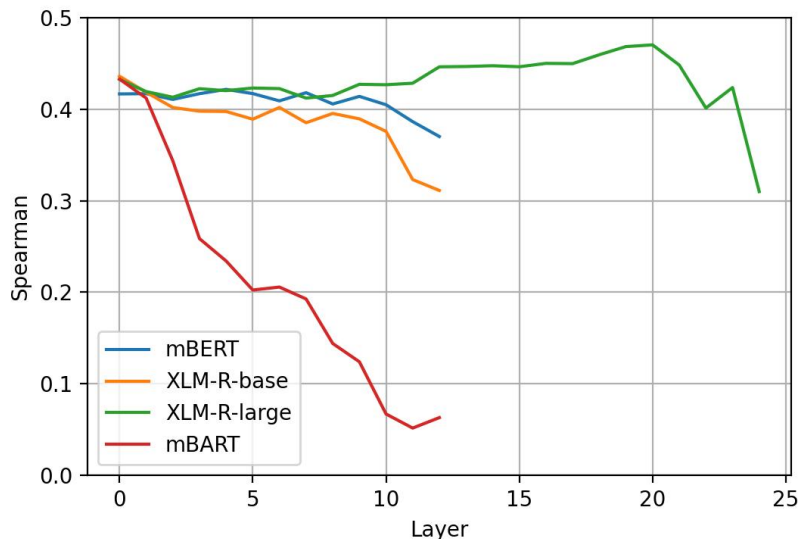


Figure 7.1: Model-layer search of source-based BERTScore.

We performed a similar tuning procedure as the authors of BERTScore, but we tuned the score using the annotated dataset from the previous Section 7.2.1. We selected four multilingual pre-trained models with Czech data in their pre-training corpus and evaluated the score for each layer. We chose three encoders, namely two versions of XLM-R, its base version with 12 layers and its large version with 24 layers, and the multilingual BERT model with 12 layers recommended by the authors. We also tested one encoder-decoder model, mBART. While evaluating mBART, only the first 12 encoder layers were considered.

The results of tuning are shown in Figure 7.1. In this figure, we report the Spearman correlation of the score computed using each model and the corresponding layer with the human-annotated dataset. We can see that the best-performing model is the large version of XLM-R, with the optimal layer being layer 20. Interestingly, the mBART model performed the worst, with a steep decline in performance for deeper layers. The optimal layer for the multilingual BERT is layer number 4, while the recommended layer is layer 9.

When we mention using BERTScore somewhere later in this work, we will be using our optimized version with XLM-R-large and its layer 20. This could be again suboptimal in some cases as we optimized it as a source-based metric using only the precision. However, we hypothesize it could be better optimized for comparing texts in the same language than the default version. We call our optimized version **BERTSource** when used as a source-based metric.

⁹<https://huggingface.co/bert-base-multilingual-cased>

7.2.3 Metrics comparison

The only other metric we found fit to be used for source-based evaluation were ROUGE-N and ROUGE-L. We did not consider ROUGE-CS because of the way the similarity is computed. The vector similarity is computed pairwise between both texts and is sometimes counted negatively. This metric was explicitly constructed for reference-based evaluation, so we did not evaluate it as a source-based metric. Memes-CS was also not evaluated because it is based on XLM-R. Its input size is only 512 tokens, which would be insufficient in most cases when the source document and summary are concatenated.

While evaluating ROUGE-N and ROUGE-L, we only calculated precision because recall and F1-score would penalize shorter summaries. This metric will give us the ratio of n-grams in the summary also occurring in the source to the total number of n-grams in the summary.

Metric	Pearson	Spearman	Kendall
ROUGE-1	0.399	0.390	0.320
ROUGE-2	0.436	0.411	0.337
ROUGE-L	0.437	0.404	0.331
BERTScore	0.422	0.414	0.338
BERTSource	0.474	0.470	0.385

Table 7.1: Evaluation of metrics for source-based evaluation. We show Pearson, Spearman, and Kendall correlations with a human-labeled dataset. ROUGE-1/2/L represents ROUGE precision. We also report an evaluation of BERTScore with the recommended setting and our optimized version.

Table 7.1 shows the evaluation results of the proposed metrics. Our optimized BERTScore achieved the best results in the evaluation, but the improvement over ROUGE is not very significant. We can see that the original BERTScore is comparable with ROUGE.

7.3 Extractive model

We chose XLM-R as our base model for extractive summarization. Specifically, we used the large version of XLM-R¹⁰, and followed the approach proposed in Section 6.1.1. XLM-R accepts only 512 tokens at its input, so we tried to extend the positional embedding layer with extra 512 positions by randomly initializing the remaining embeddings. The models with longer inputs showed no improvements over the base version in our initial experiments, so we used the model with an input size of 512 tokens. This section will provide details on the data preparation and the training procedure of our model.

7.3.1 Extractive datasets

We cannot straight-forwardly use our dataset to train the extractive model. The model needs labels for each sentence in the source document. We use labels 1 and 0. Label 1 means that the sentence should be included in the summary, and 0 means that the sentence should be excluded.

We used a very simple procedure to convert our dataset to the dataset for extractive summarization. Given sentences in the source document D , sentences in the reference

¹⁰<https://huggingface.co/xlm-roberta-large>

summary R , and metric M , we matched every sentence from R with a sentence from D while maximizing M . The label for each sentence in D would be 1 if it got matched with one of the sentences from R and 0 otherwise. In case two sentences from R got matched with the same sentence from D , we took the next best matching sentence.

We applied this procedure to our main dataset and built two such datasets using two different metrics. We used the mean of ROUGE-1, ROUGE-2, and ROUGE-L F1-scores for one dataset and BERTScore F1-score for the second one.

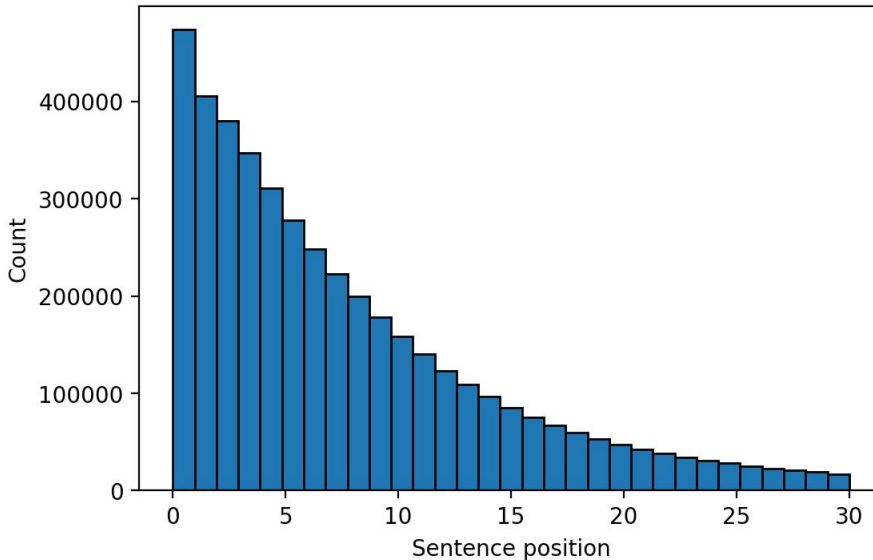


Figure 7.2: Distribution of sentences selected as extractive summary.

Figure 7.2 shows the distribution of sentence positions selected with the mean ROUGE metric. The distribution with the BERTScore looks nearly identical. We truncated the plot at the 30th position, but the trend remains the same. We can see that the sentences selected most frequently tend to be at the beginning of the article. From this, it is apparent that even a simple method as picking only the first three sentences as an extractive summary, could be a strong baseline.

7.3.2 Training

In total, we fine-tuned two extractive models. One model on the dataset maximizing mean ROUGE, and the second, on the dataset maximizing BERTScore. We call our models **EXT-R** and **EXT-B**, respectively.

The training was performed on four NVIDIA A100 40GB GPUs. We kept the hyperparameter setting the same for both models. Specifically, we used a batch size of 8 samples per GPU with gradient accumulation every two samples. Combined with the distributed training on 4 GPUs, this yields an effective batch size of 64 examples. We capped the maximum number of source sentences to consider to 32. We did this in order to have uniformly sized contextual embeddings for the classifier. Also, our metrics tend to select leading sentences, so most sentences labeled 1 will be at the beginning. We used the Adam optimizer [Kingma and Ba, 2014] with a maximum learning rate set to $3e-5$ with 10,000 warming-up steps and linear learning rate decay after that.

We evaluated the models during training on a small subset of the development split of the SumeCzech dataset consisting of 5,000 samples. This evaluation dataset was transformed

in the same manner as the corresponding training dataset. The evaluation was run every 1,000 steps, and the model with the smallest evaluation loss was selected as the final model. The EXT-R model converged after 50,000 steps, and the second model EXT-B after 32,000 steps. The training of each model took roughly one day.

7.3.3 Inference setting

During inference, we have to decide on the number of sentences to pick to form the extractive summary. The average number of sentences in each summary in our training dataset is 2.97, so by default, we used three sentences for the summary, but the number could be arbitrary. We simply pick the top sentences based on the scores given by the classifier. The selected sentences are returned in the document order.

Following [Liu and Lapata, 2019], we employ trigram-blocking in order to reduce redundant information in the extractive summary. While selecting sentences for the extractive summary, we keep track of all the currently included trigrams. In this case, a trigram represents three consecutive words. When appending the next sentence to the summary, we first check whether it has any overlapping trigrams with the previously selected sentences. If there is any overlap, we skip this sentence and try to add the next best one.

7.3.4 Results

In order to have an idea of the upper bound on the achievable ROUGE scores using extractive models, we also computed results for summaries constructed by simply selecting the sentences maximizing the mean ROUGE with the reference. We will call this method the oracle.

Model	R1	R2	RL	RCS1	RCS2	RCS3	RCS4
Oracle	22.9	5.8	15.6	10.4	4.3	-0.6	-6.6
First	14.4	2.1	9.6	2.8	-1.6	-5.3	-12.1
EXT-R	16.8	2.8	11.2	5.6	0.7	-3.2	-9.6
EXT-B	16.1	2.7	10.7	4.8	0.9	-3.4	-9.8

Table 7.2: Overlap-based metric results of extractive models. We show F1-scores of ROUGE-1/2/L, and ROUGE-CS-1/2/3/4 computed for the test split of the SumeCzech dataset. R is shorthand for ROUGE, and RCS for ROUGE-CS. Bold font indicates the best result, not considering the oracle.

Model	BERTScore	Memes	BERTSource
Oracle	28.1	51.4	64.4
First	21.8	55.8	89.1
EXT-R	25.0	51.8	75.9
EXT-B	25.8	52.7	76.9

Table 7.3: Model-based metric results of extractive models. We report the Memes metric, BERTSource, and the F1-score of BERTScore computed for the test split of the SumeCzech dataset. Bold font indicates the best result, not considering the oracle.

We also present the results of the baseline, selecting the first three sentences of the source as the extractive summary, and we call it simply *First* in the results. The results of the overlap-based methods are shown in Table 7.2, and the results of the model-based

metrics are in Table 7.3. All results were computed for the test split of the SumeCzech dataset.

As expected, the EXT-R model achieved the best results on the overlap-based metrics, although EXT-B slightly surpassed it on the 2-gram ROUGE-CS. On the other hand, the results of model-based metrics are quite unexpected. EXT-B performed best on the reference-based BERTScore, but the baseline selecting the first three sentences achieved the best Memes and BERTSource scores. This baseline is the only method that always produces coherent summaries, so we hypothesize that these two metrics prioritize coherent texts.

7.4 Rewrite model

In this section, we will describe the training process of our model for rewriting extractive summaries. As outlined in section 6.1.2, we selected the mBART model as our base pre-trained model. We specifically used the mBART pre-trained on a corpus containing 25 different languages¹¹, including Czech.

We limited the encoder and decoder input size to 256 tokens. This should be sufficient as we will be only inputting short text to each part of the model. The input to the encoder will be a short incoherent text, and the decoder’s input will be a coherent text of similar length.

7.4.1 Rewrite dataset

We use our extractive models to generate summaries with exactly three sentences. In order to fine-tune mBART to rewrite these summaries, we need to have source-target training pairs, where the source is three sentences long incoherent text, and the target is three sentences long coherent text corresponding to the source.

To construct such a dataset, we used a similar approach as in [Zhang et al., 2023]. We applied two strategies to sample the training pairs from our data:

1. **The target is from the abstract, source from the text.** We first selected the first three sentences from the abstract as our target. Then we matched each sentence from the target with sentences from the text of the article to obtain our incoherent source. We used the mean of ROUGE-1/2/L F1 scores to match the sentences. In case the abstract is shorter than three sentences, we skip this example.
2. **The target and source are from the text.** We selected the first three sentences from the text of the article to be our coherent target. Next, we matched the target sentences with sentences from the rest of the text to form the incoherent source. We used the same metric for the matching as in the previous strategy. Articles shorter than six sentences were skipped.

We combined all the training pairs from both strategies. In total, we collected 1,633,561 training pairs. However, this data contains a lot of noise as most of the pairs do not match very well. In some cases, even a sentence with a very low mean ROUGE score can get matched, as there might be no other choice to choose from. This noise in the data could be a source of model hallucinations. To reduce the noise in the dataset, we decided to keep only the top 20% of the pairs produced based on the overall mean ROUGE between

¹¹<https://huggingface.co/facebook/mbart-large-cc25>

Source

Místo je ve vagónu využito opravdu do posledního centimetru. Originální bydlení ve vlaku rozhodně není pro každého. Prostor je zařízen velmi jednoduše.

Target

Bydlet je dnes už možné opravdu kdekoli. Ubytování ve starém vagóně mezi nevšedními příbytky dnešní doby rozhodně není nic až tak atypického. Interiér je velmi útulný a působí příjemným domácím dojmem.

Figure 7.3: Example from the rewrite dataset. English translation can be found in Figure B.2.

the source and target. An example of a training sample is shown in Figure 7.3. We were left with 326,712 samples which we split into training and development splits. We kept 95% of these samples for training and 5% for validation.

7.4.2 Training

We trained the model for 6 epochs on four NVIDIA A100 40GB GPUs. We set the per GPU batch size to 8 samples. This gave us an effective batch size of 32 examples. The maximum learning rate was set to $3e-5$ with 500 warming-up steps and linear learning rate decay after. The model was optimized using AdamW optimizer [Loshchilov and Hutter, 2017] with the weight decay parameter set to 0.01

We evaluated the model every 1,000 steps on the validation split of the dataset and selected our final model as a model with minimum evaluation loss. The training converged after approximately 50,000 steps and took 6 hours.

7.4.3 Results

We did not reserve any data for testing from the generated dataset, as we did not plan to use this model individually but only as a part of a bigger model described in the following section. To give an idea of the model’s performance, Table 7.4 provides ROUGE score results on the development split of the generated dataset. These scores might be biased as we selected the final model based on the mean cross-entropy loss over this dataset split.

ROUGE-1	ROUGE-2	ROUGE-L
28.7	9.8	22.6

Table 7.4: ROUGE score results of the rewrite model on the development split of the generated dataset. Only the F1-scores are presented for each.

7.5 EXT-RW model

In the previous two sections, we trained models for extractive summarization and the model for rewriting extractive summaries. The natural next step is to combine these models into a single pipeline. This way, we will get two abstractive models. We will combine the EXT-R and EXT-B models with our rewrite model and call them **EXT-R-RW** and **EXT-B-RW**, respectively.

7.5.1 Inference setting

When generating summaries using this pipeline, we first extract three sentences from the source document using one of the extractive models. Then we apply the rewrite model to the extractive summary. We use beam search with beam size set to 6 to generate the rewritten summary. We also employ trigram blocking to avoid repetitions in the resulting summary. An example of the generation process using the EXT-R-RW model is depicted in Figure 7.4.

The rewrite model usually attempts to rewrite the text by reorganizing it but also often generates new words. The model still tends to hallucinate, as seen in the example. The last sentence in the rewritten summary, "Silničáři varují před náledím.", translated as "The road workers warn of icy roads." cannot be directly inferred from the extractive summary or even the source document.

Source document

Následující dny by měly charakterizovat teploty do minus osmi stupňů Celsia a ojedinělé sněhové přeháňky. Koncem pracovního týdne může řidičům zkomplikovat život led na silnicích. "V pátek očekáváme polojasno až oblačno, na severu místy sněžení, které bude v nižších polohách přecházet v déšť, při kterém se bude tvořit ledovka," řekl Jovanovič. Sobota by podle něj měla být deštivá, citelné ochlazení přijde až v neděli. "Očekáváme oblačno, místy se sněžením, v průběhu dne polojasno beze srážek. Nejnižší noční teploty -6 až -10 stupňů, nejvyšší denní -8 až -4 stupně," uvedl meteorolog. "V první polovině příštího týdne už bude panovat - pravděpodobně s výjimkou úterka, respektive středy - velmi chladné počasí," doplnil Jovanovič. Teploty by podle něj měly klesnout výrazně pod bod mrazu. "V údolích bude i pod minus dvacet stupňů, odpoledne vystoupí rtuť teploměru na -10 až -5 stupňů."

Extractive summary

Koncem pracovního týdne může řidičům zkomplikovat život led na silnicích. "V pátek očekáváme polojasno až oblačno, na severu místy sněžení, které bude v nižších polohách přecházet v déšť, při kterém se bude tvořit ledovka," řekl Jovanovič. Sobota by podle něj měla být deštivá, citelné ochlazení přijde až v neděli.

Rewritten summary

Řidiči by měli být opatrní při jízdě na silnicích v následujících dnech. V pátek bude převážně polojasno až oblačno, na severu místy sněžení, v nižších polohách se bude tvořit ledovka. Silničáři varují před náledím.

Figure 7.4: Example from the EXT-R-RW pipeline. English translation can be found in Figure B.3.

7.6 BRIO

The authors of BRIO [Liu et al., 2022] came up with the idea of jointly optimizing the cross-entropy and contrastive loss (Eq. 3.2). They showed that this improved the summaries in terms of ROUGE and BERTScore. We will apply BRIO with the base model using multiple metrics to our dataset in order to find whether this method also helps with improving the facticity of the generated summaries or it just exploits some metric-specific characteristics.

First, we will describe the base model we used for the multi-task fine-tuning. Then we will describe the training dataset reduced of potentially noisy examples. This reduction

might further help with lowering the amount of hallucinations. Finally, we will give a description of how we generated candidate summaries and how we trained our models.

7.6.1 Base model

As outlined in section 6.2.1, we selected fine-tuned mBART as our base model. Specifically, we chose the HT2A-CS model from [Krottil, 2022]. This model was fine-tuned on the same corpus we use. It is based on the version of mBART pre-trained on the corpus containing 25 languages.

Before fine-tuning the pre-trained model, the headline and the main text were concatenated. We will use the same approach as the headline might provide additional context and thus reduce noise in the data. Furthermore, the model was already fine-tuned this way, so it only seemed reasonable to stick with this approach.

During training, the maximum input size was set to 512 tokens for the encoder and 128 tokens for the decoder. We will keep the input sizes the same for our models.

7.6.2 Reduced dataset

We empirically found that our dataset contains quite a lot of noise. In our case, by noise, we mean that some information from the abstract is not repeated in the text. A noisy example is shown in Figure 7.5. A common case is that the abstract contains the person’s first and last name, while the main text does not repeat the person’s first name and mentions only their last name. Another typical case is that the person’s age is listed in the brackets after their name is mentioned in the abstract, but it is not again mentioned in the main text. The example shows both of these cases.

<i>Text</i>
S kamarádkou a partou dalších přátel odcestovala oslavit vánoční svátky a příchod nového roku na ostrov Svatý Bartoloměj v Karibském moři, kde si užívá teploty dosahující 30 stupňů Celsia. „Krásné Vánoce z ráje!“ připsala Vojtová k fotografiím z pláže, na kterých předvádí svou štíhlou figuru v bikinách. Dlužno poznamenat, že takové volno je rozhodně příjemnější než zimní počasí v New Yorku, kde topmodelka žije už od roku 2011.
<i>Abstract</i>
Žádné kabáty a huňaté šály. Česká topmodelka úspěšná i v zahraničí Linda Vojtová (31) utekla před zimou do tepla.

Figure 7.5: Example of noisy sample. English translation can be found in Figure B.4.

It is clear that such examples in the training data encourage the model to hallucinate. To reduce the amount of noise, we decided to filter out some examples from our dataset.

In [Filippova, 2020], the author experimented with filtering out noisy examples from the data and studied its effect on the factual consistency of the generated summaries. It was found that filtering the data based on a simple metric, such as the ratio of overlapping words between the text and the reference summary to the number of words in the reference, can improve the faithfulness of the generated summaries.

The method described above is equivalent to filtering the data based on ROUGE-1 precision. We decided to filter our data based on the BERTSource metric, as it better correlates with human judgment.

The authors of BRIO used the CNN / DailyMail dataset [Hermann et al., 2015] for multi-task fine-tuning of their BART model. The training split of this dataset contains less than

300,000 examples, and the authors stated that the training of the BRIO model converged in less than one training epoch. For this reason, we decided to keep only the top 200,000 examples from our training dataset based on BERTSource. This reduced dataset should be sufficient for the training and will significantly reduce the computational overhead from generating candidate summaries.

7.6.3 Generating candidates

The original BRIO used 16 candidate summaries for the multi-task fine-tuning. We used only 15 candidates, as 16 were the maximum number of candidates (not including the reference summary) we could fit into a single GPU memory. This allowed us to compute the multi-task loss (Eq. 3.4) for each sample in a single forward pass.

We used the diverse beam search inference method with 15 beams divided into 15 individual groups to generate the candidate summaries for each sample. The diversity penalty hyperparameter was set to 1.0.

The candidate generation process is the most computationally demanding step of this method. It took over 12 hours to generate the candidates for all 200,000 samples using four NVIDIA A100 40GB GPUs. We used these candidates during all the subsequent experiments.

7.6.4 Training

In total, we trained five BRIO-based models. All models were trained under the same setting. The only thing that differs between the models is the metric used during contrastive loss computation. We experimented with maximizing the mean ROUGE, ROUGE-CS-1, BERTScore, BERTSource, and the Memes metrics. We call our four models maximizing these metrics: **BRIO-R**, **BRIO-RCS**, **BRIO-B**, **BRIO-BS**, and **BRIO-M**, respectively.

Before training each model, we pre-sorted the candidate summaries from the previous section based on the specific metric. This way, we did not have to sort the candidates during each training step.

We kept the hyperparameters the same during each model’s training. We set them mostly according to the original paper. The per-GPU batch size was set to 1 as we could not fit more examples on a single GPU. One example consists of the main text, the reference summary, and the 15 candidate summaries. To deal with the small batch size, we used gradient accumulation over 8 mini-batches and trained the model in a distributed setting on four NVIDIA A100 40GB GPUs. That way, we achieved an effective batch size of 32 examples.

The maximum learning rate was set to $3e-5$. We also used 10,000 warm-up steps and linear learning rate decay after the warm-up phase.

The method-specific hyperparameters were kept exactly the same as in the original paper. The margin parameter of contrastive loss from Eq. 3.2 was set to 0.001, and the length penalty from Eq. 3.3 to 2.0. The weight of the contrastive loss in the multi-task loss (Eq. 3.4) was set to 100.

All models were trained for a single training epoch. We also evaluated each model every 250 steps using the metric it was trained to maximize and saved the checkpoint with the best evaluation score. We used a small subset of the development split of the SumeCzech dataset for evaluation. One training epoch, along with the evaluations, took just under 6 hours.

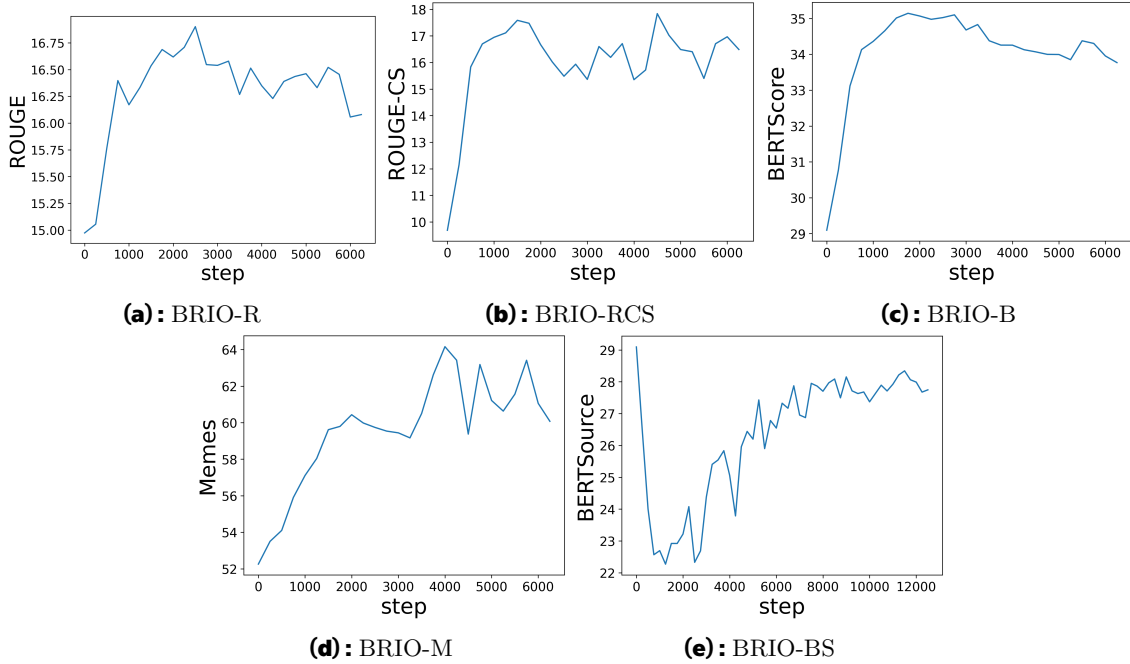


Figure 7.6: Training process of the BRIO-based models.

The training of the BRIO-BS model was not successful, as the BERTScore on the evaluation dataset was not improving throughout the training. The score was steeply declining at the beginning of the training and then began slowly rising but did not surpass the BERTScore of the base model. For this reason, we decided not to evaluate the model in this work any further.

All the remaining models converged within one training epoch (6250 optimization steps). The training process of all models can be seen in Figure 7.6. The figure shows that the evaluation metrics of all models except the BRIO-BS were first rising very quickly and then eventually settled or started to decline. The score at step 0 marks the score of the base model.

We tried to train the BRIO-BS model for two epochs, as it seemed that it might surpass the initial score, but it was not the case. As can be seen in the figure, the score started to settle after the first epoch.

We also found that this method is very sensitive to the hyperparameter setting. When we attempted to change the method-specific parameters to other than the recommended setting, the evaluation score usually diverged from the beginning for all models.

7.6.5 Inference setting

To generate the summaries, we used beam search with six beams. We restricted the generated sequence length to a minimum of 55 tokens and a maximum of 128, as this was the cutoff length of the reference summary during training.

We also employed trigram-blocking, which prohibits three consecutive tokens from being generated more than once in the sequence. This way, we might reduce the repetitions in the generated text, which is a known problem of natural language generation methods [Fu et al., 2021].

We did not try to optimize the inference setting but rather chose the parameters empirically, as optimizing it would be computationally demanding. However, optimizing the

inference parameters might provide further improvements.

7.7 Results

In this section, we will provide the results of our methods. First, we will show the results of the automatic evaluation metrics described earlier in this work. Then we will compare the models based on human evaluation and inspect whether our approaches bring improvements regarding factual consistency. Example summaries generated by our models will be presented at the end of this section.

7.7.1 Automatic evaluation

Here we present the automatic evaluation results for all the summarization models built in this work. We used the test split of the SumeCzech dataset for the evaluation so that we could compare our models with previous results. All models were evaluated on the task of predicting the summary given only the source document. We also present the ROUGE scores of the unsupervised extractive summarization method TextRank [Mihalcea and Tarau, 2004], as well as an abstractive Transformer-based method called tensor2tensor in the results, both obtained by the authors of the SumeCzech dataset. The results for the BRIO base model HT2A-CS were computed mainly by us, as the model’s author provides only the ROUGE scores. Thus, we copied the ROUGE scores computed by the author and computed all the other metrics using the same inference setting we used for our BRIO-based models. The values of all the presented metrics are multiplied by 100 for better clarity and because it is a convention in other works on a similar matter.

Model	R1	R2	RL	RCS1	RCS2	RCS3	RCS4
Extractive							
First	14.4	2.1	9.6	2.8	-1.6	-5.3	-12.1
textrank [Straka et al., 2018]	13.8	2.0	8.9	–	–	–	–
EXT-R	16.8	2.8	11.2	5.6	0.7	-3.2	-9.6
EXT-B	16.1	2.7	10.7	4.8	0.9	-3.4	-9.8
Abstractive							
tensor2tensor [Straka et al., 2018]	11.3	1.0	8.7	–	–	–	–
HT2A-CS [Krotil, 2022]	17.9	4.7	13.4	11.0	4.5	-1.1	-7.0
EXT-R-RW	17.2	3.3	12.0	6.4	2.3	-2.3	-8.2
EXT-B-RW	16.8	3.2	11.6	6.3	2.5	-2.4	-8.3
BRIO-R	21.8	5.4	15.1	9.8	6.6	0.9	-4.6
BRIO-RCS	19.6	4.8	13.8	18.9	6.3	0.0	-5.1
BRIO-M	19.3	4.2	12.9	4.1	3.7	-1.4	-8.0
BRIO-B	20.8	5.0	14.4	9.4	6.7	0.9	-4.5

Table 7.5: Overlap-based metric results. We show F1-scores of ROUGE-1/2/L, and ROUGE-CS-1/2/3/4 computed for the test split of the SumeCzech dataset. R is shorthand for ROUGE, and RCS for ROUGE-CS. Bold font indicates the best result in the respective category.

The results of the overlap-based metrics are shown in Table 7.5. The evaluated metrics include ROUGE-1/2/L and CS-ROUGE-1/2/3/4. Only the F1-scores for each of the measures are presented. We separated the results of the extractive and abstractive methods and highlighted the best results in each category. The model called *First* in the results

is the model selecting the first three sentences of the source document as an extractive summary.

The abstractive models achieved the overall best results on the overlap-based metrics. Namely, the BRIO-R performed best on the ROUGE metrics, and BRIO-RCS reached a very high ROUGE-CS-1 score – the metric it was trained to maximize. However, the best ROUGE-CS-2/3/4 was achieved by BRIO-B. This shows that maximizing only the 1-gram ROUGE-CS does not transfer well to higher n-gram versions of the metric.

The best-performing model in the extractive category was the EXT-R model. It reached the best results on almost all overlap-based metrics among other extractive methods except the ROUGE-CS-2, where it was outperformed by EXT-B.

Model	BERTScore	Memes	BERTSource
Extractive			
First	21.8	55.8	89.1
EXT-R	25.0	51.8	75.9
EXT-B	25.8	52.7	76.9
Abstractive			
HT2A-CS [Krotil, 2022]	29.5	52.2	64.4
EXT-R-RW	26.9	47.3	63.1
EXT-B-RW	27.5	48.3	64.9
BRIO-R	33.3	51.3	54.0
BRIO-RCS	31.4	51.4	54.6
BRIO-M	30.5	62.3	49.0
BRIO-B	35.6	51.9	54.7

Table 7.6: Model-based metric results. We report the Memes metric, BERTSource, and the F1-score of BERTScore computed for the test split of the SumeCzech dataset. Bold font indicates the best result in the respective category.

Table 7.6 presents the model-based metric results. This table shows results on BERTScore, Memes, and BERTSource metrics. We again show only the F1-scores in the case of BERTScore.

The BRIO-B and BRIO-M models performed best on the metrics they were trained to maximize, i.e., BERTScore and the Memes metric. On the other hand, the extractive models outperformed the abstractive ones on the BERTSource metric. However, this seems reasonable as the BERTSource should measure the consistency with the source document, and the extractive summary should always be consistent.

The BRIO-based models scored relatively low on the BERTSource metric compared to the base HT2A-CS model, which might indicate that the summaries produced by these methods are not as consistent with the source.

We can also see that maximizing a single metric using the BRIO method also often leads to improvements in the other metrics, suggesting that the BRIO-based models are not simply exploiting some metric-specific characteristics.

Interestingly, the combined models EXT-R-RW and EXT-B-RW slightly improved upon the extractive models in terms of the overlap-based metrics and BERTScore. They achieved this only by rewriting the extractive summary without any additional context. We hypothesize that this might be because the training data for the rewrite model contain some training pairs where the target is the first three sentences of the reference summary. The reference summaries are often written to be catchier and more entertaining, and the rewrite model might catch some of these properties.

7.7.2 Human evaluation

We also conducted a human evaluation in order to assess the facticity of the summaries generated using our models. Human evaluation is necessary in this case as there is yet no other way to evaluate the factual consistency of the summaries.

We generated summaries for 100 randomly sampled documents from the test split of the SumeCzech dataset. We concatenated the headline with the main text for each document and then generated the summary based on this concatenation. Only the abstractive models were evaluated, as the extractive ones should always produce factually consistent summaries.

The annotation was done in the same manner as in Section 7.2.1. Each document-summary pair was assigned one of the following labels:

- **Perfect** – The summary is factual and grammatically correct
- **Sufficient** – The summary is factual with minor grammatical errors
- **Incorrect** – Factually inconsistent summary

We also obtained 110 document-summary pairs generated by the HT2A-CS model annotated under the same setting. The annotator was also the same as in our case – our colleague, a journalism student from the Faculty of Social Sciences at Charles University. The documents were not the same as ours, but they were also sampled from the test split of the SumeCzech dataset. The inference setting was also different and was set by the author of the model. We still believe the comparison should be fair as the documents were drawn from the same corpus.

Model	Perfect (%)	Sufficient (%)	Incorrect (%)	Factual (%)
HT2A-CS [Krottil, 2022]	29.1	10.0	60.9	39.1
EXT-R-RW	14.0	11.0	75.0	25.0
EXT-B-RW	23.0	21.0	56.0	44.0
BRIO-R	2.0	4.0	94.0	6.0
BRIO-RCS	1.0	6.0	93.0	7.0
BRIO-M	1.0	7.0	92.0	8.0
BRIO-B	4.0	12.0	84.0	16.0

Table 7.7: Human evaluation results. The last column is the sum of the Perfect and Sufficient columns.

Table 7.7 shows the human evaluation results. We also included a column that represents the percentage of the factual summaries. We obtained it by calculating the sum of the Perfect and Sufficient summaries and dividing it by the total number of annotated summaries.

We can see that all the BRIO-based models are significantly underperforming, which is quite surprising given they were the top performers on almost all the automatic evaluation metrics. The best BRIO-based model is the BRIO-B model with 16% factual summaries, which is still far behind the HT2A-CS model, the base model we used for our BRIO models.

The combined models EXT-R-RW and EXT-B-RW produce factual summaries more often but do not bring any significant improvements over current models. The EXT-R-RW is not considerably better than the BRIO models, but the EXT-B-RW is comparable

to HT2A-CS or slightly better in terms of facticity. However, the HT2A-CS still has a higher ratio of the *Perfect* summaries than EXT-B-RW.

Notice that the ranking based on the ratio of factually consistent summaries obtained from the human evaluation roughly corresponds to the ordering given by the BERTSource obtained during the automatic evaluation. This further suggests that BERTSource might be a more appropriate metric to measure the facticity of generated summaries than currently used metrics.

From this evaluation, we can conclude that directly maximizing currently available automatic evaluation metrics using the BRIO method does not lead to any improvement in the factual consistency of generated summaries. We did not manage to train the BRIO model using BERTSource as the underlying evaluation metric. However, we believe that maximizing such source-based evaluation metrics could bring improvements in facticity, and we leave this for future works.

7.7.3 Example summaries

This section shows example summaries generated from all our models. Summaries are presented in Figure 7.7. These summaries were generated based on the headline and the source document. The headline and the source document are also shown, but we truncated the source to clarify the figure.

The abstractive summaries were taken from the annotated dataset we used for human evaluation. All of them seem reasonable at first, and they attempt to capture all the key information from the article. However, only the summary generated from the EXT-R-RW model was marked as *Perfect* during the annotation, except the extractive summaries, as they were not annotated. Other summaries from the abstractive models were labeled as *Incorrect*.

The abstractive models still demonstrate classic problems of all generative language models. Issues such as hallucinations and mixing up entities or numbers are still present, as can be seen in the example.

7.7.4 Discussion

We conducted an extensive evaluation of our models. We evaluated them on multiple overlap-based and model-based automatic evaluation metrics. Our models achieved better results on the automatic metrics than the previous methods. However, after conducting a human evaluation, we found that high scores on currently available evaluation metrics do not necessarily mean that the generated summaries are consistent with the source document.

All current metrics evaluate the generated summaries based on comparison with the reference summary. This approach is insufficient when we are checking for facticity. In order to better capture the factual quality of the summaries, new source-based metrics need to be developed.

We also argue that the training data quality highly influences the resulting model’s performance. We found that the data we used to train our models contain quite a lot of noise. Thorough data filtering or cleaning might provide significant improvements in factual consistency. This could also be a reasonable next step in future works.

Headline + Source text

Silniční vlaky na autopilota. Po Evropě by mohly jezdit už za šest let. Asociace evropských výrobců automobilů (ACEA) vyzvala Evropskou unii, aby do roku 2023 umožnila provoz takzvaných kamionových konvojů. Jde o nejméně dva kamiony za sebou, kdy reálně řídí jen první kamion a další nákladní vozy jedou v těsném sledu za ním a řídí se samy. Řidič je na místě jen jako kontrola. V angličtině se pro takovou technologii vžilo označení platooning, v češtině se mluví nejčastěji o konvoji nebo jízdnicích četách. Výrobci, kteří už technologii několik let testují, chtějí, aby se do běžného provozu dostaly takové kamiony do roku 2025. (...)

EXR-R

Silniční vlaky na autopilota. Po Evropě by mohly jezdit už za šest let. Asociace evropských výrobců automobilů (ACEA) vyzvala Evropskou unii, aby do roku 2023 umožnila provoz takzvaných kamionových konvojů.

EXT-B

Po Evropě by mohly jezdit už za šest let. Asociace evropských výrobců automobilů (ACEA) vyzvala Evropskou unii, aby do roku 2023 umožnila provoz takzvaných kamionových konvojů. Řidič je na místě jen jako kontrola.

EXT-R-RW

Silniční vlaky na autopilota. Po Evropě by mohly jezdit už za šest let. Asociace evropských výrobců automobilů vyzvala Evropskou unii, aby do roku 2023 umožnila provoz takzvaných kamionových konvojů.

EXT-B-RW

Evropské kamiony by mohly jezdit už za šest let. Asociace evropských výrobců automobilů (ACEA) vyzvala Evropskou unii, aby do roku 2023 umožnila provoz takzvaných kamionových konvojů. Řidič je na místě jen jako kontrola.

BRIO-R

Do roku 2025 by po Evropě mohly jezdit kamiony, které se samy řídí. **Takzvané automatické řízení kamionů chce Evropská unie do roku 2023 umožnit do běžného silničního provozu.** A to už za šest let.

BRIO-B

Do roku 2025 by po Evropě mohly jezdit kamiony s automatickým řízením. **Do roku 2023 by mohly na silnicích řídit kamiony samotní řidiči.** Dopravci chtějí, aby se tato technologie rozšířila po celé Evropě.


BRIO-RCS

Do roku 2025 by mohly jezdit po Evropě takzvané kamionové konvoje, které řídí samy kamiony. **Dopravci se snaží, aby se taková technologie dočkala rozšíření do běžného provozu i v Česku a na Slovensku.**

BRIO-M

Do roku 2025 by se v Evropě mohly do běžného provozu dostat kamiony, které jezdí samy za sebou a nepotřebují řidiče. **Řídí je kamiony a další nákladní vozy.** Technologie takzvaného konvojového řízení by mohla začít fungovat už za šest let.

Figure 7.7: Example summaries. Parts of the summaries containing non-factual information are highlighted. English translation can be found in Figure B.5.



Chapter 8

Conclusion

In this work, we reviewed current state-of-the-art methods for abstractive and extractive text summarization, as well as the most commonly used summarization evaluation measures. We then applied these methods or their slight variation on the summarization of Czech news articles, aiming at improving the factual consistency of the generated summaries.

A new facticity evaluation metric called BERTSource was also proposed in this work. This metric compares the generated summary with the source document instead of the reference summary, as all the previous methods do.

We experimented with both abstractive and extractive approaches. Our extractive models were based on the XLM-R model. We trained one model to maximize the ROUGE scores with the reference summaries and the second to maximize BERTScore. The models were evaluated on the SumeCzech dataset and reached higher ROUGE scores than previous extractive methods.

To enhance the fluency of the extractive summaries, we developed a model dedicated to rewriting them in order to improve coherence. By combining the extractive and rewrite models into a pipeline, we obtained two abstractive models. These combined models surpassed the extractive models in terms of both ROUGE and BERTScore.

The fully abstractive models developed in this work were trained using a novel training paradigm called BRIO, which jointly optimizes the cross-entropy loss with the reference summary and some summary quality measure through contrastive loss. Different metrics were considered as the quality measures, including ROUGE, ROUGE-CS, BERTScore, Memes, and BERTSource. However, the BERTSource-maximizing model was not evaluated due to the lack of improvement in the BERTSource score on the evaluation dataset during training. Nonetheless, all other models demonstrated significant improvements in their respective evaluation metrics, often showing enhancements in other metrics as well. This suggests that the models do not rely solely on metric-specific characteristics.

Finally, we conducted a human evaluation in order to assess the factual quality of the summaries generated using our abstractive models. The best performance was observed with the summaries produced by the extractive models combined with the rewrite model. However, no or slight improvement was achieved compared to existing methods. The model optimizing BERTScore performed similarly to current methods, while the model maximizing ROUGE performed poorly in terms of facticity.

Although our BRIO-based models reached state-of-the-art results in terms of all the evaluated automatic evaluation metrics, the human evaluation revealed a significant decrease in the factual consistency of the summaries generated by these models.

Despite the fact that we did not achieve any improvements in terms of facticity, we believe someone may find this work useful. We showed that the currently used evaluation metrics are not a good indicator of factual summaries. Future research should focus on



Bibliography

- [Conneau et al., 2019] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., and Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.
- [Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.
- [Filippova, 2020] Filippova, K. (2020). Controlled hallucinations: Learning to generate faithfully from noisy data. *arXiv preprint arXiv:2010.05873*.
- [Fu et al., 2021] Fu, Z., Lam, W., So, A. M.-C., and Shi, B. (2021). A theoretical analysis of the repetition problem in text generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12848–12856.
- [Hendrycks and Gimpel, 2016] Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.
- [Hermann et al., 2015] Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28.
- [Holtzman et al., 2019] Holtzman, A., Buys, J., Du, L., Forbes, M., and Choi, Y. (2019). The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.
- [Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- [Krottil, 2022] Krottil, M. (2022). Metody sumarizace textu v češtině. B.S. thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum.
- [Kudo, 2018] Kudo, T. (2018). Subword regularization: Improving neural network translation models with multiple subword candidates. *arXiv preprint arXiv:1804.10959*.
- [Kudo and Richardson, 2018] Kudo, T. and Richardson, J. (2018). Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.
- [Lewis et al., 2019] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

- [Lin, 2004] Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- [Liu, 2019] Liu, Y. (2019). Fine-tune bert for extractive summarization. *arXiv preprint arXiv:1903.10318*.
- [Liu et al., 2020] Liu, Y., Gu, J., Goyal, N., Li, X., Edunov, S., Ghazvininejad, M., Lewis, M., and Zettlemoyer, L. (2020). Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.
- [Liu and Lapata, 2019] Liu, Y. and Lapata, M. (2019). Text summarization with pre-trained encoders. *arXiv preprint arXiv:1908.08345*.
- [Liu et al., 2022] Liu, Y., Liu, P., Radev, D., and Neubig, G. (2022). Brio: Bringing order to abstractive summarization. *arXiv preprint arXiv:2203.16804*.
- [Liu et al., 2019] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.
- [Loshchilov and Hutter, 2017] Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.
- [Mihalcea and Tarau, 2004] Mihalcea, R. and Tarau, P. (2004). Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.
- [Mikolov et al., 2013] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.
- [Nakayama et al., 2018] Nakayama, H., Kubo, T., Kamura, J., Taniguchi, Y., and Liang, X. (2018). doccano: Text annotation tool for human. Software available from <https://github.com/doccano/doccano>.
- [Narayan et al., 2018] Narayan, S., Cohen, S. B., and Lapata, M. (2018). Don’t give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745*.
- [Rajpurkar et al., 2018] Rajpurkar, P., Jia, R., and Liang, P. (2018). Know what you don’t know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.
- [Sennrich et al., 2015] Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.
- [Straka et al., 2018] Straka, M., Mediankin, N., Kocmi, T., Žabokrtský, Z., Hudeček, V., and Hajic, J. (2018). Sumeczech: Large czech news-based summarization dataset. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.
- [Ullrich et al., 2023] Ullrich, H., Drchal, J., Rýpar, M., Vincourová, H., and Moravec, V. (2023). CsFEVER and CTKFacts: acquiring czech data for fact verification. *Language Resources and Evaluation*.

- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- [Vijayakumar et al., 2016] Vijayakumar, A. K., Cogswell, M., Selvaraju, R. R., Sun, Q., Lee, S., Crandall, D., and Batra, D. (2016). Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*.
- [Wu et al., 2016] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., et al. (2016). Google’s neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.
- [Zhang et al., 2020] Zhang, J., Zhao, Y., Saleh, M., and Liu, P. (2020). Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR.
- [Zhang et al., 2019] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2019). Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.
- [Zhang et al., 2023] Zhang, Z., Liang, X., Zuo, Y., and Li, Z. (2023). Unsupervised abstractive summarization via sentence rewriting. *Computer Speech & Language*, 78:101467.
- [Zvára, 2022] Zvára, Š. (2022). Ověřování fakticity výstupů metod abstraktivní sumarizace textů. B.S. thesis, České vysoké učení technické v Praze. Vypočetní a informační centrum.



Appendix A

Acronyms

BERT Bidirectional Encoder Representations from Transformers

BART Bidirectional and Auto-Regressive Transformers

MLM Masked Language Modeling

NSP Next Sentence Prediction

RoBERTa Robustly Optimized BERT Pretraining Approach

XLM Cross-lingual Language Model

XLM-R XLM-RoBERTa

BRIO Bringing order to abstractive summarization

ROUGE Recall-Oriented Understudy for Gisting Evaluation

Memes Metric for Evaluating Model Effectiveness in Summarization

CNC Czech News Center

EXT-R Extractive model maximizing the mean of ROUGE-1/2/L

EXT-B Extractive model maximizing BERTScore

EXT-R-RW EXT-R model combined with the rewriting model

EXT-B-RW EXT-B model combined with the rewriting model

HT2A-CS Model from [Krotil, 2022] trained on the combination of SumeCzech and CNC dataset to predict the abstract given the headline and the source text.

BRIO-R BRIO model maximizing the mean of ROUGE-1/2/L

BRIO-RCS BRIO model maximizing ROUGE-CS-1

BRIO-B BRIO model maximizing BERTScore

BRIO-M BRIO model maximizing Memes

Appendix B

Translations of examples

Source document

The US President warned that what is happening in Belarus now could have an impact on the whole region of Central and Eastern Europe: "It makes us less safe." He also added what needs to be done: "It is important that we support the Belarusians in building civil society." Obama was reacting to the fact that a court in Minsk on Thursday sent two unsuccessful presidential candidates, Mikalay Statkevich and Dzmitry Uss, to prison for six years and five-and-a-half years. Both were convicted of organising a demonstration last December after President Lukashenko's re-election. The White House already condemned the convictions on Friday and announced new sanctions against the regime in Minsk. Barack Obama also held talks with Polish Prime Minister Donald Tusk. After his meeting with President Komorowski, the White House chief greeted people who were instrumental in the fall of communism in Poland. Among them were activists of the independent trade union Solidarity, including Tadeusz Mazowiecki and Zbigniew Janas. Absent from Obama's meeting with former Polish dissidents was Lech Walesa, who told Polish media on Friday that he was not comfortable with the meeting. Walesa reportedly found out about the meeting at the last minute. Walesa was criticised for this. "A politician who refuses to meet the leader of a world power, a great ally of Poland, because of offended pride, is making a huge mistake," Gazeta Wyborcza said, adding: "By not speaking even for a moment with the US president, Walesa insulted the guest of Poland's top leaders."

Generated summary

The White House chief Barack Obama warned on Friday after meeting with **Belarusian** President **Bronislaw** Komorowski that the situation in Belarus could have an impact on the whole of Central and Eastern Europe. Obama also held talks with Polish dissidents who were instrumental in the fall of communism in Poland.

Figure B.1: Translation of the example of model hallucination. Red words are hallucinated or wrongly inferred from the source document.

Source

Every centimeter of space in the carriage is used. Original train living is definitely not for everyone. The space is furnished very simply.

Target

It is now possible to live anywhere. Staying in an old wagon among the unusual housing of today is certainly not that atypical. The interior is very cosy and has a pleasant homely feel.

Figure B.2: Translation of the example from the rewrite dataset.

Source document

The following days should be characterised by temperatures down to minus eight degrees Celsius and sporadic snow showers. Ice on the roads may make life difficult for drivers at the end of the working week. "On Friday, we expect partly cloudy to cloudy skies, with snowfall in the north, changing to rain at lower altitudes, which will form ice," Jovanovic said. Saturday should be rainy, he said, with a noticeable cooling coming on Sunday. "We are expecting cloudy skies, with snow in places, and partly cloudy with no precipitation during the day. Lowest night temperatures -6 to -10 degrees, highest daytime temperatures -8 to -4 degrees," the meteorologist said. "In the first half of next week, very cold weather will prevail - probably with the exception of Tuesday and Wednesday," Jovanovic added. Temperatures should drop well below freezing, he said. "In the valleys it will be below minus 20 degrees, and in the afternoon the thermometer will rise to -10 to -5 degrees."

Extractive summary

Ice on the roads can make life difficult for drivers at the end of the working week. "On Friday, we expect partly cloudy to cloudy skies, with snowfall in the north, changing to rain at lower altitudes, which will form ice," Jovanovic said. Saturday should be rainy, he said, with a noticeable cooling coming on Sunday.

Rewritten summary

Drivers should take care when driving on the roads in the coming days. Friday will be mostly partly cloudy to cloudy, with some snow in the north and icy conditions at lower elevations. The road workers warn of icy roads.

Figure B.3: Translation of the example from the EXT-R-RW pipeline.

Text

With a friend and a group of other friends, she went to celebrate the Christmas holidays and the arrival of the New Year on the island of St. Bartholomew in the Caribbean Sea, where she enjoys temperatures reaching 30 degrees Celsius. "Merry Christmas from paradise!" Vojtová added to photos from the beach, showing off her slim figure in a bikini. It's worth noting that this kind of time off is definitely more pleasant than the winter weather in New York, where the top model has been living since 2011.

Abstract

No coats or fluffy scarves. Linda Vojtová (31), a Czech top model who is also successful abroad, escaped the cold to warmth.

Figure B.4: Translation of the example of noisy sample.

Headline + Source text

Road trains on autopilot. They could be running across Europe in as little as six years. The European Automobile Manufacturers Association (ACEA) has called on the European Union to allow the operation of so-called truck convoys by 2023. This involves at least two trucks in a row, with only the first truck actually driving and the other trucks following closely behind and driving themselves. The driver is only there as a check. In English, this technology is known as platooning, while in Czech it is most often referred to as convoys or driving platoons. The manufacturers, who have been testing the technology for several years, want to see such trucks in regular use by 2025. (...)

EXR-R

Road trains on autopilot. They could be running across Europe in as little as six years. The European Automobile Manufacturers Association (ACEA) has called on the European Union to allow the operation of so-called truck convoys by 2023.

EXT-B

They could be running across Europe in as little as six years. The European Automobile Manufacturers Association (ACEA) has called on the European Union to allow so-called truck convoys by 2023. The driver is only there as a check.

EXT-R-RW

Road trains on autopilot. They could be running across Europe in as little as six years. The European Automobile Manufacturers Association has called on the European Union to allow so-called truck convoys to operate by 2023.

EXT-B-RW

European trucks could be running in six years. The European Automobile Manufacturers Association (ACEA) has called on the European Union to allow so-called truck convoys by 2023. The driver is only on site as a check.

BRIO-R

By 2025, self-driving trucks could be driving across Europe. **The European Union wants to make so-called automatic truck driving a reality on the roads by 2023.** And that's just six years from now.

BRIO-B

By 2025, trucks with automatic steering could be driving across Europe. **By 2023, trucks could be driven by drivers themselves.** Transport operators want this technology to spread across Europe.

BRIO-RCS

By 2025, so-called lorry convoys, driven by the trucks themselves, could be running across Europe. **Transport operators are trying to get such technology to become widespread also in the Czech Republic and Slovakia.**

BRIO-M

By 2025, trucks that drive behind each other and do not need a driver could become common in Europe. **They are driven by trucks and other goods vehicles.** So-called convoy driving technology could be operational in as little as six years.

Figure B.5: Translation of the example summaries. Parts of the summaries containing non-factual information are highlighted. English translation can be found in Figure B.5.

Appendix C

Attached files

In this section, we provide the structure of the attached files.

```
src
├── brio
│   ├── brio_utils.py
│   ├── config.py
│   ├── generate_candidates.py
│   ├── sort_candidates.py
│   ├── test.py
│   └── train.py
├── data-filtering
│   ├── calculate_metrics.py
│   └── data-filtering.ipynb
├── extractive
│   ├── build_ext_dataset.py
│   ├── config.py
│   ├── dataset.py
│   ├── model.py
│   ├── test.py
│   ├── train.py
│   └── utils.py
├── metrics-evaluation
│   ├── data
│   │   ├── doccano.jsonl
│   │   └── prep_data.jsonl
│   ├── data-prep.ipynb
│   ├── metric-comparison.ipynb
│   └── model-layer-search.ipynb
├── rewrite-model
│   ├── build-ds.ipynb
│   ├── build-rw-ds.py
│   ├── config.py
│   └── train.py
└── rouge_raw.py
```