TUM Data Innovation Lab

Munich Data Science Institute (MDSI)

Technical University of Munich

&

Zweites Deutsches Fernsehen (ZDF)

Final report of project:

Developing a Small Language Model for News Article Summarization

Authors	Hesham Ghonim, JiWoo Hwang, Tim Nielen, Zeyad Shaheen, Michael Timothy
Mentor(s)	Dr. Carolin Isabel Bauerhenne
TUM Mentor	Dr. Alessandro Scagliotti
Project lead	Dr. Ricardo Acevedo Cabra (MDSI)
Supervisor	Prof. Dr. Massimo Fornasier (MDSI)

Aug 2025

Acknowledgements

We would like to express our sincere gratitude to our project sponsor ZDF and the TUM Data Innovation Lab, whose support made this project possible.

We are especially thankful for the invaluable guidance and encouragement provided by our ZDF mentor Dr. Carolin Bauerhenne and our TUM mentor Dr. Alessandro Scagliotti throughout the course of this work.

We are deeply grateful to our project lead, Dr. Ricardo Acevedo Cabra, for giving us the opportunity to take part in this project.

Finally, we would like to thank the editors at ZDF who generously contributed their time and expertise to the human evaluation of our data.

Abstract

High-quality automatic summarization of German news articles presents a valuable opportunity for streamlining editorial workflows in broadcasting organizations such as ZDF. However, the effectiveness of small and mid-sized language models (SLMs and LLMs) for this task remains underexplored, particularly under resource constraints and domain-specific requirements. In this work, we evaluate the summarization capabilities of four instruction-tuned language models, LLaMA-3.2-1B, Qwen2-1.5B, Teuken-7B, and LLaMA-3.1-8B, exploring different optimization techniques including n-shot prompting and parameter efficient fine-tuning via LoRA. We assess model performance through both automatic metrics and targeted human evaluation, and validate the quality of ZDF's internal article-summary dataset for training and benchmarking.

Our results show that LLaMA-3.1-8B delivers consistently high-quality summaries, even in zero- and few-shot settings, with fine-tuning yielding outputs that rival or exceed human references in fluency and relevance. In contrast, smaller models struggled with hallucinations and coherence issues, highlighting their current limitations for deployment.

Relevance to ZDF and Society: Automated news summarization has broad societal implications, especially in an era of information overload and rapidly evolving news cycles. By enabling the efficient generation of concise, accurate summaries, language models can support journalists, editors, and media consumers alike by helping professionals focus on verification and analysis, while making high-quality information more accessible to the public. For public broadcasters like ZDF, integrating trustworthy AI-driven tools can improve editorial productivity and scalability without compromising journalistic standards. This work demonstrates that such technology is not only feasible with today's open models, but also increasingly practical on consumer hardware, paving the way for transparent, cost-effective AI adoption in public-interest media.

Contents

Abstract **2**

1 Introduction **4**
1.1 Problem Definition and Goals 4

2 Related Works **5**
2.1 Summarization & Evaluation 5
2.2 Large Language Models 6
2.2.1 LLaMa 6
2.2.2 Teuken-7B 6
2.3 Small Language Models 7
2.4 Ethical Considerations 8

3 Methodology **8**
3.1 Data Preparation and Analysis 8
3.1.1 Data Preparation 8
3.1.2 Data Distribution and Metadata 9
3.1.3 Data Splitting 10
3.2 Hardware 11
3.3 Model Choice 11
3.4 Optimization Techniques 11
3.4.1 Prompt Engineering 12
3.4.2 Truncation 12
3.4.3 Fine-tuning 12
3.4.4 Loss Calculation 14
3.4.5 Hyperparameter Tuning 14
3.5 Metrics 16
3.5.1 Human Evaluation Justification 16
3.5.2 Automatic Metrics: ROUGE and F1-BERTScore 16
3.5.3 Metrics After Translation 16
3.5.4 Metric Robustness to Summarization Length 17
3.5.5 Interplay Between Human Evaluation and Metrics 17

4 Experiment Results **18**
4.1 Completion Ratio 18
4.2 Human Evaluation 19
4.3 AEM-based Analysis 19
4.3.1 Correlation of AEMs and Human Ratings 20
4.3.2 Interpretation of n-shot Results 21
4.3.3 Truncation 21
4.3.4 AEMs after Fine-tuning 21
4.4 Qualitative analysis 22

5 Conclusion **23**

1 Introduction

The increasing availability of digital news content across online platforms has introduced challenges for efficient content curation. While manual summarization remains valuable, it is often labor-intensive and difficult to integrate seamlessly into fast-paced editorial environments [20]. The current rise of artificial intelligence, especially in the field of natural language processing motivated ZDF to explore possibilities for Automatic Text Summarization (ATS) to reduce manual effort and to better align editorial workflows. ATS systems aim to algorithmically condense large textual inputs into concise summaries, preserving essential information while minimizing redundancy and noise. These systems offer substantial benefits in terms of speed, scalability, and cost-effectiveness, and are applicable across domains such as journalism, legal analysis, scientific literature review, and intelligence gathering [28, 2].

In recent years, the rise of Large Language Models (LLMs), such as GPT [6], BERT [10], and their successors, has transformed the field of natural language processing (NLP), including summarization tasks. LLMs have demonstrated impressive capabilities in both abstractive and extractive summarization, owing to their high capacity, contextual awareness, and ability to generalize across diverse domains. However, the deployment of LLMs is often constrained by their computational overhead, memory requirements, and inference latency. These limitations pose significant challenges for real-time, on-device, or resource-constrained applications, where efficiency and cost are critical considerations [5].

In light of these constraints, this work investigates the underexplored potential of Small Language Models (SLMs) as an alternative to LLMs for ATS. While SLMs typically underperform LLMs in general-purpose NLP benchmarks, recent research suggests that task-specific fine-tuning, architectural optimization, and domain adaptation can significantly enhance their utility [35, 42, 19]. Our goal is to evaluate whether SLMs, when carefully designed and tuned, can offer a competitive balance between performance and efficiency in the context of news summarization. In doing so, we aim to advance the development of lightweight, accessible, and deployable summarization systems that can operate effectively in low-resource settings without compromising on output quality.

1.1 Problem Definition and Goals

In this project, we address the challenge of automatically summarizing news articles in a manner that balances quality, efficiency, and resource constraints. The problem is particularly relevant given the continuous influx of textual content generated by news organizations and media outlets. To facilitate this research, we collaborated with ZDF (Zweites Deutsches Fernsehen), a major German public-service television broadcaster, who generously provided us with a large dataset of annotated news articles and summaries written by ZDF expert editors in German language. This dataset forms the empirical foundation of our evaluation and experimentation pipeline.

The overarching aim of our study is to systematically explore the capabilities of both

Large Language Models (LLMs) and Small Language Models (SLMs) in the task of news summarization. We formulate three principal goals that guide our investigation:

1. **Assessing the quality of news summarization using language models in German language.** We evaluate how well different language models, regardless of size, can capture and condense the key information in news articles, using both automated metrics (e.g., ROUGE, BLEURT, BERTScore) and human evaluation.

2. **Comparative analysis of SLMs and LLMs.** We compare the summarization quality produced by SLMs against that of LLMs, while also considering associated computational costs such as inference time, memory usage, and hardware requirements. This trade-off analysis is critical for applications where resource efficiency is an important consideration.

3. **Model recommendations based on deployment context.** Based on our findings, we aim to provide informed recommendations on which class of models, SLMs or LLMs and even further which model specifically, is most appropriate under specific conditions, such as real-time processing, offline summarization, or low-resource environments.

Through this multi-faceted evaluation, our project seeks to contribute actionable insights into the practical deployment of summarization models, particularly in media and broadcasting settings where both quality and efficiency are crucial. It is also important to note, while this work was undertaken to advance scientific understanding, any potential applications are intended to support editorial workflows, ensuring human oversight and full alignment with ZDF's principles for the responsible use of artificial intelligence.

2 Related Works

2.1 Summarization & Evaluation

Summaries take various forms, such as headlines, sentence-level condensations, bullet-point highlights, or comprehensive full-text summaries [20]. Additionally, there are multiple ATS techniques, such as by selecting important existing sentences (extractive), by generating new sentences through paraphrasing (abstractive), or by combining both approaches.

For evaluation, human assessment of faithfulness (factual correctness), coherence (logical flow), and relevance (alignment with the source) would yield the most reliable evaluations [48]. Furthermore, several Automatic Evaluation Metrics (AEMs) exist, including ROUGE, BLEURT, BERT-Score, BARTScore, and MoverScore. ROUGE, a traditional and widely used metric, measures n-gram overlap but struggles with semantic equivalence when phrasing differs [24]. BERTScore employs token-level embedding similarity to better handle semantics but can yield high similarity scores even in cases of meaning reversal due to its token-wise nature [49]. MoverScore computes semantic similarity via optimal transport distances between word embeddings, effectively capturing context and paraphrasing but also at higher computational cost [50]. BLEURT leverages fine-tuned transformers to

predict human-like judgments directly, showing high correlation with human assessments, but it is computationally intensive and lacks interpretability [37]. Lastly, BARTScore evaluates summaries by calculating sequence-to-sequence probabilities, providing flexibility including reference-free evaluation but requiring significant computational resources [46].

Recent studies highlight the dataset dependency of AEM reliability. For example, ROUGE-L often correlates better with human evaluation for extractive summarization tasks, whereas BERTScore may excel in abstractive summarization contexts [4]. Moreover, [48] emphasize the critical influence of data quality on AEM performance. Their analysis shows that reference-based metrics sometimes have negative correlations with human judgments regarding faithfulness, particularly in datasets with low-quality reference summaries like CNN/Daily Mail and XSUM, derived from incidental supervision.

2.2 Large Language Models

2.2.1 LLaMa

The LLaMA family of models [41], developed by Meta, is an open-source collection of transformer models offering a range of natural language processing tasks by supporting large-scale deployment and streamlined fine-tuning use cases. Ranging from 7B to 65B parameters at the time of the initial LLaMA release in 2023 [41] and as high as 70B for LLaMA 2 [40], these models perform competitively with the state-of-the-art. While most LLaMA models are LLMs, some smaller models have less than 4B parameters; therefore, they fall under the definition of SLMs.

Among the newer releases, LLaMA 3 [16] supports multilinguality with dedicated training methods for non-English languages. Specifically, Meta trained on a token mix of 90% multilingual content, thus generating a multilingual expert model. In addition, LLaMA 3 introduces support for a token context length of 128K tokens, which is particularly useful for summarization applications with long-document comprehension. In fact, summarization was introduced into the formal training curriculum for LLaMA 3, further enhancing its capability for German summarization.

2.2.2 Teuken-7B

Mehdi Ali [1] and colleagues present Teuken-7B-Base and Teuken-7B-Instruct, two open-source large language models designed to support all 24 official European Union languages. This work addresses a significant gap in the AI landscape, where most models exhibit strong English-centric bias, limiting accessibility and performance for non-English European languages.

The research is motivated by the need for linguistically diverse AI systems that can serve Europe's multilingual population without requiring costly translation or adaptation. The authors emphasize that existing models like BLOOM and LLaMA, while supporting multiple languages, still underperform for many European languages, particularly those with

complex morphology or limited resources.

Methodologically, the models were trained on an extensive 4 trillion-token dataset with 60% non-English content [31], sourced from web crawls, academic texts, and the FineWeb-EDU dataset. A key innovation is their custom multilingual tokenizer, specifically optimized to reduce token fragmentation across European languages. This tokenization strategy particularly benefits morphologically rich languages like Finnish, German, and Hungarian, improving computational efficiency and text processing.

The base architecture consists of a 7-billion parameter transformer-based decoder-only model with a 4096-token [32] context window, employing grouped-query attention and rotary positional embeddings. Following pretraining, Teuken-7B-Instruct underwent instruction tuning focused on high-quality English and German data, including multi-turn dialogues and reasoning tasks.

Evaluation across multilingual benchmarks (EU21-ARC, EU21-HellaSwag, EU21-TruthfulQA, EU21-MMLU) demonstrates competitive performance, with Teuken-7B-Instruct achieving 54.3% average accuracy across benchmarks. The model shows particular strength in commonsense reasoning tasks and maintains consistent performance across both high and low-resource languages.

When compared to similar initiatives like EuroLLM, Teuken-7B models excel in general instruction-following capabilities while maintaining strong multilingual support. However, limitations persist in specialized domains like coding and mathematics, attributed to training data constraints.

2.3 Small Language Models

Smaller Language Models (SLMs) are gaining traction due to their low inference latency, cost-efficiency, ease of deployment, and adaptability, making them ideal for resource-constrained settings and domain-specific tasks. Unlike LLMs, SLMs offer advantages in privacy-sensitive applications and scenarios requiring fast, lightweight fine-tuning [43]. These properties make SLMs particularly suitable for tasks like news summarization.

While there is no universally accepted definition, SLMs (also referred to as sLLMs) are commonly understood as neural, transformer-based language models with $\leq$ 4B parameters. In this paper, we adopt this definition. A straightforward approach to SLM development involves using smaller variants of established LLMs. Models such as Qwen2-1.5B-Instruct, TinyLLAMA, Phi-3-Mini, and LLaMA-3.2-1B-Instruct fall under this category.

When properly selected and fine-tuned with high-quality task-specific data, SLMs can approach or even exceed the performance of larger LLMs. They may perform within 10% of state-of-the-art models like GPT-4o and outperform models such as DS-2, GPT-4o-mini, and Gemini-1.5-Pro in terms of efficiency and task alignment [38]. SLMs, when fine-tuned, have also shown strong results in content moderation and creative writing, key components of news summarization, often matching or surpassing LLMs in this domain

[47, 26, 14, 44].

2.4 Ethical Considerations

Concerns about LLMs are growing, particularly regarding their environmental impact, embedded social biases, and the false sense of understanding they can create. For example, to train such model as GPT-3 can emit as much carbon as five average Americans do in a year [39]. Large, uncurated online datasets that silence minority perspectives and overrepresent dominant opinions are used to train LLMs. For instance, content from websites like Reddit, where conversations frequently express sexist or racist views, is included in the Common Crawl dataset used to train GPT-3 [3]. Even worse, LLMs run the potential of propagating false information since they imitate patterns without understanding their meaning. For example, GPT-3 can produce smooth, persuasive text supporting conspiracy theories, which extremists could weaponize [27].

This connects us to the idea of "stochastic parrots", which holds that LLMs assemble text according to statistical patterns to give the appearance of coherence without any real purpose or significance. Fluent language, even when produced by machines, is inherently meaningful to humans. GPT-3, for instance, is capable of creating a phony Q&A regarding extreme views that appears reasonable but lacks factual support [27]. This "coherence in the eye of the beholder" is dangerous because people trust these outputs as if they were written by humans, not realizing the models lack accountability or grounding in reality.

This raises big questions: Are bigger models always better? For instance, GPT-3 has privacy problems because of its 175 billion parameters, which enable it to remember and replicate private information from its training data [8]. In the meantime, the emphasis on growing LLMs distracts from ethical alternatives. Smaller, more focused models might be safer and greener. For example, models such as DistilBERT [35] or ALBERT [22] employ techniques such as "knowledge distillation" to reduce large models into more manageable, quicker, and energy-efficient variants.

3 Methodology

3.1 Data Preparation and Analysis

ZDF provided us with a large dataset of their text data containing many formats of publications including web content, video descriptions, articles and the corresponding human-produced summaries. Through pre-processing and further analysis, we made sure that we ended up with high quality data, suitable for fine-tuning and model evaluation.

3.1.1 Data Preparation

The original dataset provided by ZDF comprised 248,753 article-summary pairs (*textModule* and *metaDescription*), exclusively containing ZDF-produced news content. Due to limited prior knowledge of the dataset, we iteratively inspected random samples to guide

our filtering process.

First, we removed all entries with *null* article or summary fields, reducing the dataset to 50,825 valid pairs, eliminating roughly 80% of the original data. Next, we discarded entries containing ellipses ("...") in summaries, which typically indicate truncation, and removed cases where the summary was a substring of the article, as these often reflect extractive redundancy.

To ensure content quality, we applied length-based filters: articles with fewer than 100 words and summaries with 15 or fewer words were excluded, as such entries were typically insubstantial or resembled metadata rather than true summaries.

After applying the word count filters, we analyzed the distribution of summary lengths in our dataset and found that the overall average summary length was 350 tokens, indicating that our process successfully shifted toward richer and more informative examples.

Finally, we filtered the dataset based on content type, retaining only entries labeled as "news" while excluding other content types such as "article", "episode", "mobile-news-video", or "foto". This decision ensured stylistic and structural consistency across examples, focusing specifically on news content for our summarization task. We also excluded entries from certain brands like "In eigener Sache" to remove ZDF internal content such as information about "ZDF-Fernsehrat" or job opportunities rather than normal news articles.

This comprehensive filtering process yielded a final dataset of 24,072 clean, high-quality document-summary pairs for downstream model training and evaluation.

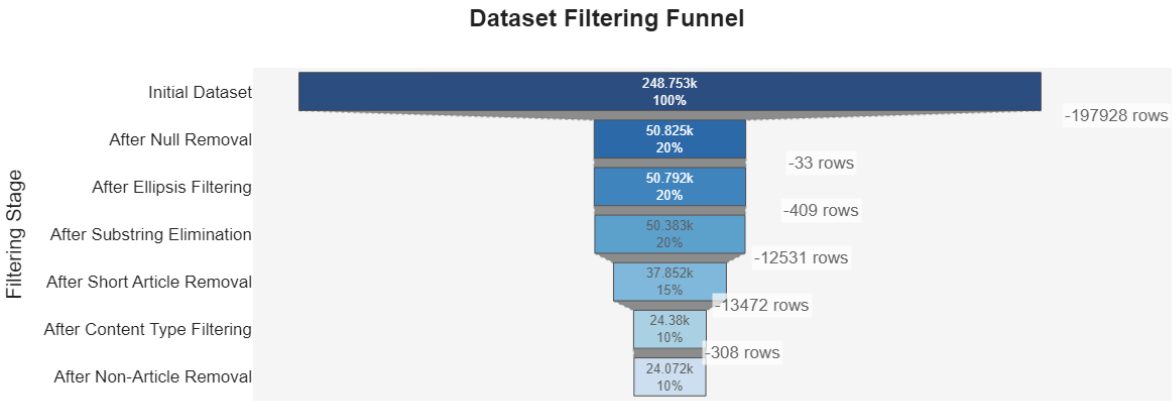

Figure 1: Dataset filtering funnel.

3.1.2 Data Distribution and Metadata

The curated dataset spans a variety of brands. Figure 2 reports the percentage share of each major brand (based on 24,072 rows) and their corresponding average word counts.

The total average word count is 373 vs. 308 on the non-curated data.

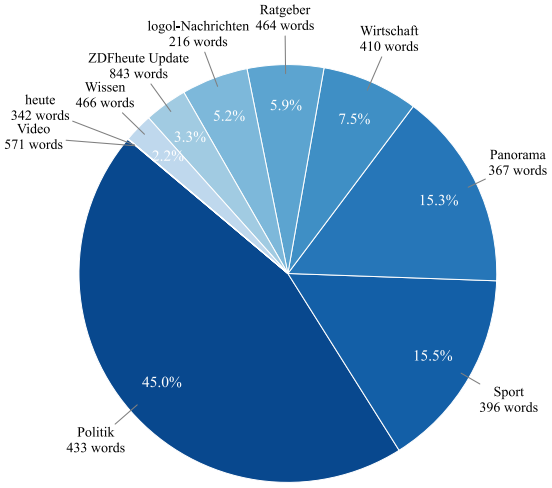

Figure 2: Pie chart of the brand distribution with corresponding average word counts.

3.1.3 Data Splitting

From the cleaned dataset of 24,072 document-summary pairs, we applied a stratified splitting strategy to ensure balanced representation across brands and article lengths.

We first constructed a test set via stratified sampling: for each of the 10 brands, articles were binned into 10 length-based groups, and samples were drawn from each bin. Although the goal was to select 10 articles per brand, some brands had fewer available samples, resulting in a final test set of 90 examples. This process was repeated with manual inspection to refine quality and ensure diversity.

After removing the test set, the remaining 23,982 examples were randomly split 80-20 into training and validation sets:

- **Training set**: 19,185 examples (80%)
- **Validation set**: 4,797 examples (20%)
- **Test set**: 90 examples

This three-way split allows us to train our models on the training set, while tracking generalization performance using the validation set, and evaluate final performance on the held-out test set, following standard machine learning practices for unbiased model evaluation.

3.2 Hardware

We were given access to *Google Colab Enterprise* by ZDF and a limited budget. Through Colab we had access to runtimes with up to eight datacenter GPUs where we could choose between the models L4[1] and A100[2]. The L4 GPU is cheap and efficient with 24 GB of

GPU Model	VRAM	FP16 performance	Cores
L4	24 GB	30 TFLOPS	7424
A100	40 GB	78 TFLOPS	6912

Table 1: Comparison of GPU RAM, FP16 (Half Precision) Performance and number of Cores

VRAM and the A100 GPU is faster but also more expensive with 40 GB of VRAM. To give an estimate of the cost, running an instance with 8 GPUs costs around 7.20 €/h for the L4 and 25.64 €/h for the A100.

Multi-GPU Training

For running and training LLMs, the amount of VRAM plays a vital role as it must fit the model. While in theory our runtime allows for eight GPUs in parallel, splitting the model across multiple GPUs would cut performance dramatically due to the great amount of communication during each execution necessary between the GPUs. We opposed this due to our time and cost limitations. Instead, we restricted ourselves to models that fit into our GPUs. During training we load a copy of the model on each GPU and divide the batch of data equally among all GPUs which they evaluate seperately. Afterwards, gradients are accumulated to perform a training step. We implemented this using *Huggingface Accelerate* which utilizes PyTorch's *DistributedDataParallel* module. This sped up training by a factor of 6 when using 8 GPUs compared to a single one.

3.3 Model Choice

We selected Teuken-7B-Instruct and Llama-3.1-8B-Instruct for the LLMs based on ZDF's interest in Teuken and Llama. These specific models are comparable in size, fit on a 24GB L4 GPU using half precision, and are instruction-tuned.

For the SLMs, we chose Qwen2-1.5B and Llama-3.2-1B. The former shows its promise in multilingual news summarization and the latter to compare directly against Llama-3.1-8B [45]. After selecting the model, we tuned the hyperparameters for each model in order to make sure we have the optimal hyperparameters for our fine-tuning process.

3.4 Optimization Techniques

There are different ways of improving a model's performance on a specific task. We focused on prompt engineering and fine-tuning. In prompt engineering, one optimizes the

[1] `https://www.techpowerup.com/gpu-specs/l4.c4091`
[2] `https://www.techpowerup.com/gpu-specs/a100-pcie-40-gb.c3623`

instruction one gives the model to minimize errors due to ambiguity. In fine-tuning, one starts from a pre-trained model and optimizes parameters, either the model parameters directly or by adding so-called adapter layers.

3.4.1 Prompt Engineering

For consistency and high performance in our experiments, we chose a prompt format based on the findings from Schubiger [36]. In the study, it was discovered that structured prompts significantly improve summarization accuracy for models such as LLaMA2-7B, especially when the prompts are in the same language as the input text. Following their results, we chose the German prompt that has been proven to show strong performance among several different prompt formats:

```
Erstelle eine Zusammenfassung vom folgenden Artikel in 3 oder weniger
Sätzen:
Artikel: {article}
Zusammenfassung:
```

n-shot learning n-shot learning refers to adapting a model to a new task using n labeled examples. In the context of LLMs, this typically involves in-context learning which is providing examples directly in the input prompt without altering model weights [7], or, alternatively, fine-tuning with a small labeled set. This paradigm is particularly useful in low-resource settings and tests a model's ability to generalize from limited supervision.

We evaluated n-shot prompting for $n \in 0, 1, 2$ to assess its effect on summarization performance. All prompts followed a shared structure, incorporating n article-summary pairs sampled from the training set. To encourage stylistic consistency, examples were drawn from the same ZDF brand as the test article.

3.4.2 Truncation

Teuken-7B's has a limited context size of 4096 tokens. This leads empty summaries if its context is completely filled by the prompt alone. This is particularly relevant for one and two shot learning, however, our longest article is around 4600 tokens long, which makes it impossible to fit even for a zero shot prompt. We avoid empty summaries by employing truncation. Specifically, once we encounter a prompt that exceeds a certain length, we remove tokens from the left until the prompt is only 3996 tokens long, leaving space for 100 tokens generated by the model.

3.4.3 Fine-tuning

Contrary to prompt engineering during fine-tuning we are optimizing parameters to teach a model a new skill. The straight forward approach is to fine-tune all parameters of the model using a gradient based optimizer. However, especially for the LLMs, this is quite inefficient as it would increase the amount of VRAM required by a factor of 2-3 as gradients and optimizer states have to be loaded into memory in addition to the model

parameters. Also, full fine-tuning has the risk of catastrophic forgetting [21].

For SLMs, we chose to perform full fine-tuning since they fit within our VRAM constraints and can benefit from additional training, particularly in improving their ability to generate coherent text. In contrast, full fine-tuning of LLMs would exceed our available VRAM and is unnecessary from a linguistic standpoint, as we assume these models already produce coherent and fluent output. Therefore, we used parameter-efficient methods to simply adapt our LLMs to our summarization task.

Parameter Efficient Fine-Tuning (PEFT) allows for fine-tuning a model using only a fraction of memory. The simplest approach would be to only fine-tune certain layers (such as the head) and freeze the rest. Intuitively, it's performance depends on the number of layers fine-tuned, therefore, scaling the memory requirements quickly as well.

Low-Rank Adaptation (LoRA) also allows for fine-tuning selective layers aswell but with a fraction of the memory required [18]. Let $P_0 \in \mathbb{R}^{n \times m}$ be the parameters of a layer we want to optimize. If we were to optimize it using stochastic gradient descent, we would calculate

$$P_{k+1} := P_k - \lambda_k \nabla L_k = P_0 - \sum_{i=0}^{k} \lambda_i \nabla L_i \tag{1}$$

where $\nabla L_i \in \mathbb{R}^{n \times m}$ is the gradient of the batch's loss regarding P_i and $\lambda_i > 0$ is the learning rate in each iteration $i = 0, ..., k$.
In LoRA we write

$$P' = P + BA \tag{2}$$

using matrices $B \in \mathbb{R}^{n \times r}, A \in \mathbb{R}^{r \times m}$ of rank $r \ll min(n, m)$ and optimize P' by freezing P and training these matrices. The number of values in A and B combined should be far less than in P (usually multiple magnitudes): $r \cdot n + r \cdot m \ll nm$.
The intuition behind LoRA is that for $BA \approx -\sum_{i=0}^{k} \lambda_i \nabla L_i$ this should yield a similar result as in common fine-tuning.
Matrices that closely approximate this sum do exist (what validates this approach) because using singular value decomposition (SVD) any matrix $M \in \mathbb{R}^{n \times m}$ of rank r can be decomposed in singular vectors $u_i \in \mathbb{R}^n, v_i \in \mathbb{R}^m$ and its corresponding singular values $\sigma_i > 0$ as

$$M = \sum_{i=1}^{r} \sigma_i u_i v_i^T =: U \Sigma V^T \tag{3}$$

It can be shown that if we truncate this sum to $\hat{r} < r$ this still yields the best approximation of rank $\hat{r}$ for M [11]. Additionally, quantized LoRA (QLoRA) offers even more parameter efficiency by using quantized parameters [9]. Common options are four or eight-bit floats.
Finally, learned embeddings or prompt tuning [23] allow the parameters of a model to be completely frozen. Instead, one introduces new embedding vectors that are appended

to the prompt. During training, only these vectors receive gradients; the standard token embeddings and all other parameters remain frozen. This is the most parameter-efficient method of the ones mentioned, as it only needs to calculate the gradients of these tokens.

As prompt tuning still lacks behind LoRA-based methods in absolute performance [23] and we are able to employ QLoRA on our hardware, we opted for this technique when fine-tuning our LLMs.

3.4.4 Loss Calculation

In order to optimize parameters through a gradient based method, we must first define a loss function. Since all of our models are so-called CausalLMs they are trained on next token prediction which allows us to train them using the cross-entropy-loss. Specifically, given a prompt and completion of token indices $p_0, ..., p_n$ and $c_0, ..., c_m$ respectively, we want to train the model to produce c_i given an input of $(p_0, ..., p_n, c_0, ..., c_{i-1})$.

CausalLMs produce one output for each input token, where the output is a distribution of floating point numbers $x_{i,j}$ resembling the probability of the token of index $j \in [N]$ succeeding the ith input token where $N \in \mathbb{N}$ is the vocabulary size. CausalLMs are autoregressive and therefore employ attention masking which makes sure each token only attends to the tokens preceding it. This allows us to train the model on a complete example in a single run rather than running it once per token.

The input is the sequence $(p_0, ..., p_n, c_0, ..., c_{m-1})$ and the labels are defined by the the sequence $(p_1, ..., p_n, c_0, ..., c_m)$. For the ith output-label pair $\big((x_{i,j})_{j\in[N]}, y_i\big)$, the cross-entropy-loss is defined as

$$l_i = -log(x_{i,y_i}) \tag{4}$$

i.e. the higher the probability of producing y_i the lower the loss. Usually, we obtain $x_{i,j}$ using the Softmax activation function

$$\sigma(\vec{z})_j = \frac{e^{z_j}}{\sum_{k=0}^{N} e^{z_k}} \tag{5}$$

where $\vec{z}$ is the output of the last layer for token i, the so called logits. Therefore, the loss affects all outputs not only the y_ith component.

To ensure the model only trains on the completions rather than the whole sequence, the first n losses are masked out. Finally, all the remaining losses are averaged to obtain the final loss for the example. Fortunately, Huggingface's "Supervised Fine-tuning Trainer" handles these processes internally, minimizing the chance of misimplementation.

3.4.5 Hyperparameter Tuning

To systematically optimize model performance, we used Optuna for hyperparameter tuning. Optuna eliminates the trial-and-error process of hyperparameter tuning by automatically searching for the best combination of hyperparameters based on an optimization

target we define. Instead of manually testing different configurations, Optuna systematically explores the hyperparameter space to find optimal values that maximize or minimize our chosen performance metric [33].

Our objective function for Optuna includes the hyperparameters shown in Table 2 with their corresponding ranges and distributions. The parameter ranges were chosen based on established best practices and empirical observations from the literature. For learning rates, we used different ranges for LLMs and SLMs due to their distinct parameter scales and training dynamics. Training a LoRA from scratch for the LLMs requires a higher learning rate than tuning all the pre-trained parameters of the SLMs. The LoRA-specific parameters (rank, alpha, dropout) are set within commonly used ranges that balance model expressiveness with parameter efficiency. The rank values {16, 32, 64} represent typical low-rank decomposition sizes that maintain performance while reducing computational overhead. The choice of `num_of_epochs` is mostly to define the curve of the learning rate scheduler. We stop the trail after 3 epochs either way to maintain consistency. Of course, in the final fine-tuning run, we do not cut off early.

Parameter	**Type**	**Range/Values**	**Notes**
learning_rate	log-uniform	LLMs: 1×10^{-5} to 3×10^{-4} SLMs: 5×10^{-6} to 3×10^{-5}	Higher for LoRA (LLMs) Lower for SLMs
lora_rank	categorical	{16, 32, 64}	Low-rank decomposition size
lora_alpha	float	0.5–2.0	Multiplied by rank for final alpha
lora_dropout	float	0.0–0.3	Regularization for LoRA layers
total_batch_size	categorical	{8, 16}	Hardware memory constraints
warmup_ratio	float	0.01–0.05	Learning rate schedule
weight_decay	float	0.0–0.05	L2 regularization
num_of_epochs	categorical	{3, 5}	Training duration
adam_beta1	float	0.8–0.95	Adam momentum parameter
adam_beta2	float	0.95–0.999	Adam second moment parameter
adam_epsilon	log-uniform	1×10^{-8} to 1×10^{-6}	Numerical stability
neftune_noise_alpha	float	0.0–10.0	Noise injection for training
max_grad_norm	float	0.5–2.0	Gradient clipping
lr_scheduler_type	categorical	{cosine, polynomial}	Learning rate decay strategy

Table 2: Hyperparameter search space for Optuna optimization.

We ran 30 Optuna trials per model, balancing search efficiency and compute budget. Each trial used 15% of the training data. The objective metric for optimization is validation loss, reflecting generalization performance.

Each trial involves testing a specific set of parameters across multiple training epochs. We considered two strategies for selecting the final parameter state in each trial: using the set of parameters that achieved the lowest validation loss during the trial ("best") or using the parameters from the final epoch ("last"). While intuitively, it might make more sense to choose the "best" parameters within a trial, we observed signs of overfitting shortly after the point of minimum validation loss. To address this, we conducted an additional Optuna optimization run using the "last" parameters from each trial, with the aim of identifying configurations that maintained more consistent performance and

reduced overfitting. The best configuration from each study was then used for final model training and evaluation.

3.5 Metrics

3.5.1 Human Evaluation Justification

Human evaluation is critical for assessing the performance of language generation models, particularly for tasks involving summarization, translation, and open-ended generation. While automatic metrics provide efficiency and reproducibility, human judgment is essential for capturing subjective qualities such as coherence, relevance, fluency, and factual consistency [17]. Human evaluators serve as the gold standard, especially in cases where automatic metrics are misaligned with human preferences.

3.5.2 Automatic Metrics: ROUGE and F1-BERTScore

To evaluate summary quality comprehensively, we combine standard and advanced metrics: ROUGE [24], BERTScore [49], and BLEURT [37], each capturing different aspects of performance. ROUGE, widely used in summarization, measures lexical overlap via ROUGE-1 (unigrams), ROUGE-2 (bigrams), and ROUGE-L (longest common subsequence), but often fails to reflect semantic adequacy, particularly in abstractive settings.

To address this, we include BERTScore, which computes similarity using contextual embeddings from pretrained Transformer models. It aligns tokens based on embedding distance and provides F1 scores that better correlate with human judgments on meaning preservation. We report BERTScore F1 using three different BERT variants, namely:

(1) `bert-base-multilingual-cased`

(2) `T-Systems-onsite/german-roberta-sentence-transformer-v2`

(3) `FacebookAI/xlm-roberta-base`.

3.5.3 Metrics After Translation

There are multiple metrics only available in English language, such as BLEURT [37], MOVERScore [50], and BARTScore [46]. Since the dataset is in German, it is conceptually possible to use another large language model for translation and evaluate the reference summarization translation with the model summarization translation. Some scientific papers support this method empirically [29, 25]. However, this is not without a cost. In practice, a larger resource will be required. In theory, the summarization error will be further affected by the translation infidelity/error.

In order to exhaust all possible metrics, we are using `facebook/nllb-200-distilled-600M` as a translation model and then BLEURT on top of that for the evaluation metric. We chose this model as it is quite popular on Huggingface currently and produced convincing translations that did not seem to change the structure of the original summary.

3.5.4 Metric Robustness to Summarization Length

When selecting evaluation metrics for summarization, it is critical to consider their robustness to variations in summary length particularly in abstractive summarization. Table 3 shows the robustness of the selected metrics with regards to length of summarization based on theoretical point of view of the practical implementation of these metrics. This however does not influence the reliability of the metrics in summarization. How fitting a metric is will be determined by its correlation to the human evaluation. [30, 13, 49, 37].

Metric	Type	Robustness to Length	Notes
ROUGE	Overlap-based	Low	Biased toward longer summaries; favors extractive systems due to reliance on n-gram overlap
BERTScore	Embedding-based	Moderate	Captures semantic similarity via contextual embeddings, but still somewhat sensitive to token count
BLEURT	Learned / Regression	Moderate	Fine-tuned on human judgments; robust to paraphrasing and summary length variability

Table 3: Robustness of evaluation metrics with respect to summary length variation. Color indicators denote qualitative robustness levels.

3.5.5 Interplay Between Human Evaluation and Metrics

While automatic metrics enable scalable and consistent evaluation, they often fail to fully capture linguistic nuance, especially in creative or complex texts. Human evaluations are therefore commonly used to complement metrics, ensuring that high metric scores reflect truly fluent and useful outputs. This combined approach yields a more robust evaluation framework [15].

To quantify alignment between human judgments and automatic metrics, we use statistical correlation analysis. Specifically, we compute *Kendall's tau* (τ), which measures rank correlation between two ordered lists, for example, human quality rankings and metric-based rankings of summaries [34]. Higher τ values indicate stronger agreement between metrics and human preferences.

We calculate Kendall's τ per article to respect annotator-specific judgments and avoid cross-annotator bias. To assess overall statistical significance, we combine the resulting p-values using Fisher's method [12], a standard technique for aggregating independent significance tests.

$$X^2 = -2\sum_{i=1}^{k} \ln(p_i) \qquad \text{where } X^2 \sim \chi^2_{2k} \tag{6}$$

This allows us to test whether the overall correlation across all 28 articles is unlikely to be due to chance, even if individual article-level p-values are not significant on their own. Overall, this methodology enables us to identify which metrics most faithfully reflect human judgments and to prioritize them in model assessment and recommendation.

4 Experiment Results

Model	n**-shot**	**Complete Summaries**
Llama-3.2-1B	0	0.22
Llama-3.2-1B	1	0.76
Llama-3.2-1B	2	0.78
Qwen2-1.5B	0	0.02
Qwen2-1.5B	1	0.43
Qwen2-1.5B	2	0.54
Llama-3.1-8B	0	0.06
Llama-3.1-8B	1	0.73
Llama-3.1-8B	2	0.90
Teuken-7B	0	0.57
Teuken-7B	1	0.77
Teuken-7B	2	0.76
Teuken-7B (trunc)	0	0.49
Teuken-7B (trunc)	1	0.78
Teuken-7B (trunc)	2	0.83

Table 4: Ratio of completed summaries across models and n-shot settings. We consider a summary complete if the model's generation did not need to be cut off after 100 tokens.

4.1 Completion Ratio

The average number of words per summary is about 20 in our data and we prompted our models to produce at most 2-3 sentences. Therefore, we decided to limit the number of tokens the model is allowed to produce to 100 which equals roughly 60-80 words as this should be plenty to produce a reasonable summary. Generation terminates once this number of tokens is exceeded and we say that the model overproduces. We call a summary complete if the model does not overproduce.

Table 4 shows the ratio of completed summaries on the test set across models and n-shot settings. We observe that all models tend to overproduce in the 0-shot setting, especially Qwen2-1.5B and Llama-3.1-8B completing only 2 and 6 percent of their respective summaries. Increasing the number of shots, i.e. the number of examples given, this ratio increases for nearly all models. For Llama-3.1-8B it even jumps up to 90% in the 2-shot setting. After fine-tuning, all models reach a completion ratio close to 1. However, we did not include those results in Table 4 to keep it readable.

4.2 Human Evaluation

Model	n-shot	Faithfulness	Relevance	Coherence	Average
Llama-3.2-1B	0	2.14	2.29	1.66	2.03
Llama-3.2-1B	1	2.16	1.95	1.82	1.98
Llama-3.2-1B	2	2.33	2.09	2.26	2.23
Qwen2-1.5B	0	1.87	1.67	1.54	1.69
Qwen2-1.5B	1	1.98	2.14	1.97	2.03
Qwen2-1.5B	2	2.25	2.23	1.84	2.11
Llama-3.1-8B	0	4.08	3.29	2.59	3.32
Llama-3.1-8B	1	3.58	3.08	3.22	3.30
Llama-3.1-8B	2	3.74	3.27	3.58	3.53
Teuken-7B	0	3.43	3.04	3.20	3.22
Teuken-7B	1	3.57	3.02	3.12	3.24
Teuken-7B	2	2.94	2.65	2.99	2.86
Reference		**4.09**	3.21	**4.12**	**3.80**

Table 5: Average human evaluation scores across models and number of shots.

Multiple human annotators rated random articles of our test set on faithfulness, relevance and coherence. In total, 456 summaries were evaluated spanning 28 articles in total. Some articles were reviewed by multiple annotators due to randomness. Due to a network error, however, we can only recover 409 usable reviews.Table 5 shows the results of the human evaluation of the models in different n-shot settings. We highlight the best and worst SLM and LLM in green and red respectively.

Llama-3.1-8B in the 0-shot setting achieves close or even better results compared to the reference in faithfulness and relevance. This hints towards the quality of our references. They tend to be rather short and abstract including only a single highlight of the article in some cases. In the 0-shot setting, Llama-3.1-8B is very verbose which the annotators seem to prefer regarding the content. However, they punish the model's coherence which is likely due to its low completion ratio as shown in Table 4.

Llama-3.1-8B in the 2-shot setting seems to have the best trade-off between content and completion reaching an average score of 3.53 which is already quite close to the reference. The same seems to hold for our two SLMs, especially the LLama model. However, they do not seem to reach the same level as the LLMs yet. Teuken-7B seems to benefit less from increasing the number of shots which is due to its context size. As it was trained on only 4096 tokens it produces nothing or gibberish if this number is exceeded.

4.3 AEM-based Analysis

Next, we compute AEMs on the generated summaries by the models in different n-shot scenarios (see Table 6). The resulting values have completely different distributions across metrics, making interpretations of single values and comparisons between different metrics useless. However, within a metric we can at least rank models accordingly. However, as

Model	n-shot	ROUGE-L	BS (1)	BS (2)	BS (3)	BLEURT
Llama-3.2-1B	0	0.1239	0.5350	0.4743	0.9853	-0.4754
Llama-3.2-1B	1	0.1256	0.5309	0.4319	0.9852	-0.5482
Llama-3.2-1B	2	0.1199	0.5294	0.4396	0.9852	-0.5385
Qwen2-1.5B	0	0.1226	0.5249	0.4795	0.9852	-0.4369
Qwen2-1.5B	1	0.1262	0.5402	0.4715	0.9856	-0.4806
Qwen2-1.5B	2	0.1405	0.5482	0.4875	0.9858	-0.5047
Llama-3.1-8B	0	0.1490	0.5532	0.5206	0.9859	-0.3790
Llama-3.1-8B	1	0.1580	0.5682	0.5175	0.9861	-0.3554
Llama-3.1-8B	2	0.1681	0.5709	0.5171	0.9862	-0.3991
Teuken-7B	0	0.1306	0.5364	0.4787	0.9853	-0.4641
Teuken-7B	1	0.1307	0.5331	0.4800	0.9746	-0.5385
Teuken-7B	2	0.1269	0.5140	0.4444	0.9523	-0.5814
Teuken-7B (trunc)	0	0.1230	0.5259	0.4566	0.9630	-0.4972
Teuken-7B (trunc)	1	0.1379	0.5484	0.4887	0.9856	-0.4690
Teuken-7B (trunc)	2	0.1413	0.5400	0.4770	0.9639	-0.5316

Table 6: Model performance with varying n-shot settings.

Table 6 shows, this ranking still differs from one metric to another. To shine some light on which metric to trust, we compute the correlation between these AEM based rankings and the human rankings.

Metric	tau	p-value
ROUGE-L	0.1550	0.0404
BERTScore F1 (1)	0.2322	0.0366
BERTScore F1 (2)	0.2498	0.0010
BERTScore F1 (3)	0.1536	0.0444
BLEURT	0.1955	0.0266

Table 7: Kendall's tau between AEMs and average human scores. The tau values are averaged over 28 articles and the aggregated p-values are obtained using Fisher's method.

4.3.1 Correlation of AEMs and Human Ratings

The tau values in Table 7 suggest that BERTScore models (1) and (2) generally achieve stronger rank correlations with human judgments compared to the others. Both metrics' scores are far from the ideal tau value of 1. This is likely due to the fact that our metrics are reference-based and as our human evaluation showed, our references are imperfect relevance-wise. However, even these moderate values validate the use of the AEM as a first indicator of a model's performance and allows us to identify certain trends. To interpret whether the computed correlations are statistically meaningful, we report the aggregated p-values obtained via Fisher's method as well. All metrics achieve low p-values, surpassing a common threshold of < 0.05, validating the statistical significance of our test.

4.3.2 Interpretation of n-shot Results

Similar to the human evaluation, the AEMs have a tendency towards Llama-3.1-8B as Table 6 shows. While BERTScore model (1) prefers the 2-shot version, model (2) prefers the 0-shot version. One possible intuitive explanation could be that model (2) focuses more on faithfulness and relevance as shown in Table 7 while model (1) weights coherence slightly stronger.

Considering that Llama-3.1-8B only has a 6% completion ratio in the zero shot setting (see Table 4), rendering it effectively useless and the potential inaccuracy of our human evaluation we tend to trust model (1) more than (2) as it prefers the 2-shot version and aligns with the majority vote in most cases.

Moreover, for the SLMs the AEMs seem to prefer 2-shot Qwen2-1.5B over Llama-3.2-1B which is contrary to the human evaluation. Again, possibly because they focus more on faithfulness and relevance rather than coherence. If we only consider the average of these two, 2-shot Qwen2-1.5B performs best on the human setting as well.

4.3.3 Truncation

Now we observe the effect of truncation for Teuken-7B. As Table 6 shows, most metrics agree that the non-truncated version in the 2-shot setting performs the worst, partially even worse than the SLMs. This is due to the fact that many 2-shot prompts exceed the models designed context length of 4096 leading to an empty output which receives a score of zero in BERTScore models.

This is a slightly different result compared to the human evaluation where Teuken-7B clearly outperforms SLMs. Likely, the reason is that a score of zero is a significant outlier for the AEMs having a greater impact on the average compared to human evaluation.

After applying truncation, the results improve, especially for the one and two shot settings. One shot with truncation yields the best results for Teuken-7B, likely as it benefits from an example but does not need to truncate as often as the two shot version leading to incomplete, possibly misleading, examples.

4.3.4 AEMs after Fine-tuning

Figure 3 contains the best scores of the non-fine-tuned models from Table 6, denoted by the `model-name`, as well as the results of the two respective fine-tuned versions denoted by `model-name-ft-best` and `model-name-ft-last`. The names correspond to the respective hyperparameter optimization objectives “best” and “last” explained in Section 3.4.5. For the fine-tuned versions, increasing the number of shots did not yield better results, which is why we are considering only the 0-shot versions here. Colors are chosen as a heatmap to make the differences easier to observe. Again, we choose green for the best model and red for the worst, interpolating between them.

As shown in Figure 3, fine-tuning Llama-3.1-8B seems to be the best model overall.

Model	ROUGE-L	BS (1)	BS (2)	BS (3)	BLEURT
Llama-3.2-1B	0.1256	0.535	0.4743	0.9853	-0.4754
Llama-3.2-1B-ft-best	0.1711	0.5616	0.4815	0.9866	-0.6349
Llama-3.2-1B-ft-last	0.159	0.5642	0.4858	0.9865	-0.6328
Qwen2-1.5B	0.1405	0.5482	0.4875	0.9858	-0.4369
Qwen2-1.5B-ft-best	0.1496	0.5599	0.4807	0.9864	-0.6604
Qwen2-1.5B-ft-last	0.1487	0.5506	0.4769	0.9862	-0.6555
Llama-3.1-8B	0.1681	0.5709	0.5206	0.9862	-0.3554
Llama-3.1-8B-ft-best	0.173	0.5737	0.5043	0.9867	-0.5731
Llama-3.1-8B-ft-last	0.1764	0.5856	0.5198	0.987	-0.5579
Teuken-7B	0.1413	0.5484	0.4887	0.9856	-0.469
Teuken-7B-ft-best	0.1763	0.5693	0.5009	0.9866	-0.599
Teuken-7B-ft-last	0.1751	0.5664	0.5018	0.9757	-0.5826

Figure 3: AEM scores of the different fine-tuned models compared with each models best score before fine-tuning.

However, the non-fine-tuned version is still quite competitive. Teuken-7B benefits from fine-tuning as well but lacks behind Llama-3.1-8B slightly. None of the fine-tuned SLMs has competitive performance in comparison to the bigger ones. They are comparable to the non-fine-tuned Teuken-7B. Even then, the non-fine-tuned Llama-3.1-8B still is the better option based on the metrics.

4.4 Qualitative analysis

While the AEM based analysis provides insights on the performance differences between different models, their actual usability in practice remains uncertain. Based on our n-shot and fine-tuning results (Table 6, Figure 3), we selected seven representative models for closer inspection: LLAMA-3.1-8B-Instruct in zero-shot, two-shot, and fine-tuned (ft-last) variants; Teuken-7B-ft-best; and three SLMs: Qwen2-1.5B-ft-best, Llama-3.2-1B-ft-last, and Llama-3.2-1B-ft-best.

Evaluation on 10 randomly sampled articles revealed consistent trends. The SLMs showed frequent hallucinations and occasional incoherence. Teuken-7B produced abstract, creative, and largely faithful summaries with minimal hallucinations, though its style occasionally introduced confusion by deviating from the source. All Llama-8B variants generated coherent, faithful summaries with minimal hallucinations, showing reliability across prompting and fine-tuning setups. The Llama-8B variants usually highlighted similar aspects of the article as the reference, while Teuken-7B occasionally prioritized stylistic flair over strict adherence to the source.

Among the Llama-8B models, the zero-shot version attempted exhaustive coverage, often resulting in incomplete outputs. The two-shot variant was more elaborate but generally stayed within length constraints. The fine-tuned version (ft-last) consistently produced concise, relevant, and sometimes superior summaries. Based on these findings, we

strongly recommend LLaMA-3.1-8B-Instruct for high-quality summarization, ideally using our fine-tuned version ft-last.

5 Conclusion

This study evaluated the capabilities of recent language models for the task of summarizing German news articles, with a focus on applicability within ZDF's editorial workflows. We examined two small language models (SLMs), LLaMA-3.2-1B-Instruct and Qwen2-1.5B-Instruct, and two larger models, Teuken-7B-Instruct and LLaMA-3.1-8B-Instruct, under various optimization strategies including n-shot prompting and parameter-efficient fine-tuning via LoRA.

Our evaluation employed three automatic metrics, whose reliability was verified through a targeted human evaluation using Kendall Tau's correlation analysis. Additionally, we also evaluated the underlying data quality of ZDF's article-summary corpus and found it well-suited for supervised training, shown by the performance increase after finetuning.
Our results demonstrate that n-shot learning alone substantially improves model adherence to summarization prompts, particularly in terms of output length and focus. Nonetheless, both SLMs suffered from significant hallucinations and coherence issues, even after fine-tuning, rendering them unsuitable for production use.
Teuken-7B, while more capable than the SLMs, exhibited context window limitations and occasional hallucinations, resulting in an inconsistent performance profile. These limitations, along with its qualitative shortcomings, preclude its recommendation for use in ZDF environments.
In contrast, LLaMA-3.1-8B-Instruct consistently delivered faithful, coherent, and relevant summaries across all configurations. The 0-shot and 2-shot variants already performed competitively, and the fine-tuned model produced outputs comparable to or better than reference summaries. It also demonstrated robustness in minimizing hallucinations, making it a strong candidate for real-world deployment. According to our qualitative analysis, after fine-tuning LLaMA-8B produced very good summaries, comparable or even better than the reference.

Overall, our findings highlight that modern LLMs, particularly LLaMA-8B, are capable of high-quality summarization of German news articles, and can be effectively optimized even with lightweight methods such as prompt tuning. This shows the practical feasibility of deploying conservatively sized models within ZDF's editorial infrastructure without compromising output quality.

Looking ahead, a possible direction involves leveraging knowledge distillation techniques to transfer the performance gains of larger models (e.g., LLaMA-8B) into smaller, more efficient SLMs. While not pursued in this work, knowledge distillation had been considered as a secondary option, with fine-tuning initially prioritized as the preferred strategy. However, due to time constraints and limited computational resources, this direction remained unexplored. Nevertheless, this approach may help mitigate the hallucination and

coherence issues currently observed in small models, while retaining much of the performance improvements gained through fine-tuning.

References

[1] Mehdi Ali et al. "Teuken-7B-Base & Teuken-7B-Instruct: Towards European LLMs". In: *arXiv preprint arXiv:2410.03730* (2024). URL: https://arxiv.org/abs/2410.03730.

[2] Mehdi Allahyari et al. "A brief survey of text mining: Classification, clustering and extraction techniques". In: *arXiv preprint arXiv:1707.02919* (2017).

[3] Emily M. Bender et al. "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?" In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. ACM. 2021, pp. 610–623. DOI: 10.1145/3442188.3445922. URL: https://dl.acm.org/doi/10.1145/3442188.3445922.

[4] Manik Bhandari et al. "Re-evaluating evaluation in text summarization". In: *arXiv preprint arXiv:2010.07100* (2020).

[5] Rishi Bommasani et al. *On the Opportunities and Risks of Foundation Models*. 2022. arXiv: 2108.07258 [cs.LG]. URL: https://arxiv.org/abs/2108.07258.

[6] Tom Brown et al. "Language models are few-shot learners". In: *Advances in neural information processing systems* 33 (2020), pp. 1877–1901.

[7] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL]. URL: https://arxiv.org/abs/2005.14165.

[8] Nicholas Carlini et al. "Extracting Training Data from Large Language Models". In: *Proceedings of the 30th USENIX Security Symposium*. USENIX Association. 2021, pp. 2633–2650. URL: https://www.usenix.org/system/files/sec21-carlini-extracting.pdf.

[9] Tim Dettmers et al. "Qlora: Efficient finetuning of quantized llms". In: *Advances in neural information processing systems* 36 (2023), pp. 10088–10115.

[10] Jacob Devlin et al. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*. 2019, pp. 4171–4186.

[11] Carl Eckart and Gale Young. "The approximation of one matrix by another of lower rank". In: *Psychometrika* 1.3 (1936), pp. 211–218.

[12] RC Elston. "On Fisher's method of combining p-values". In: *Biometrical journal* 33.3 (1991), pp. 339–345.

[13] Alexander R Fabbri et al. "Summeval: Re-evaluating summarization evaluation". In: *Transactions of the Association for Computational Linguistics* 9 (2021), pp. 391–409.

[14] Xue-Yong Fu et al. *Tiny Titans: Can Smaller Large Language Models Punch Above Their Weight in the Real World for Meeting Summarization?* 2024. arXiv: 2402.00841 [cs.CL]. URL: https://arxiv.org/abs/2402.00841.

[15] Sebastian Gehrmann et al. *The GEM Benchmark: Natural Language Generation, its Evaluation and Metrics*. 2021. arXiv: 2102.01672 [cs.CL]. URL: https://arxiv.org/abs/2102.01672.

[16] Aaron Grattafiori et al. "The llama 3 herd of models". In: *arXiv preprint arXiv:2407.21783* (2024).

[17] David M Howcroft et al. "Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions". In: *13th International Conference on Natural Language Generation 2020*. Association for Computational Linguistics. 2020, pp. 169–182.

[18] Edward J Hu et al. "Lora: Low-rank adaptation of large language models." In: *ICLR* 1.2 (2022), p. 3.

[19] Xiaoqi Jiao et al. "Tinybert: Distilling bert for natural language understanding". In: *arXiv preprint arXiv:1909.10351* (2019).

[20] Wafaa S El-Kassas et al. "Automatic text summarization: A comprehensive survey". In: *Expert systems with applications* 165 (2021), p. 113679.

[21] James Kirkpatrick et al. "Overcoming catastrophic forgetting in neural networks". In: *Proceedings of the national academy of sciences* 114.13 (2017), pp. 3521–3526.

[22] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *arXiv preprint arXiv:1909.11942* (2019). URL: https://arxiv.org/abs/1909.11942.

[23] Brian Lester, Rami Al-Rfou, and Noah Constant. "The Power of Scale for Parameter-Efficient Prompt Tuning". In: *arXiv preprint arXiv:2104.08691* (2021). URL: https://arxiv.org/abs/2104.08691.

[24] Chin-Yew Lin. "Rouge: A package for automatic evaluation of summaries". In: *Text summarization branches out*. 2004, pp. 74–81.

[25] Ziheng Lin et al. "Combining coherence models and machine translation evaluation metrics for summarization evaluation". In: *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2012, pp. 1006–1014.

[26] Guillermo Marco, Luz Rello, and Julio Gonzalo. *Small Language Models can Outperform Humans in Short Creative Writing: A Study Comparing SLMs with Humans and LLMs*. 2025. arXiv: 2409.11547 [cs.CL]. URL: https://arxiv.org/abs/2409.11547.

[27] Kris McGuffie and Alex Newhouse. *The Radicalization Risks of GPT-3 and Advanced Neural Language Models*. Tech. rep. Middlebury Institute of International Studies at Monterey, Center on Terrorism, Extremism, and Counterterrorism, 2020. URL: https://www.middlebury.edu/institute/sites/default/files/2020-09/gpt3-article.pdf.

[28] Ani Nenkova, Kathleen McKeown, et al. "Automatic summarization". In: *Foundations and Trends® in Information Retrieval* 5.2–3 (2011), pp. 103–233.

[29] Huyen Nguyen et al. *A Comparative Study of Quality Evaluation Methods for Text Summarization*. 2024. arXiv: 2407.00747 [cs.CL]. URL: https://arxiv.org/abs/2407.00747.

[30] Jekaterina Novikova et al. "Why we need new evaluation metrics for NLG". In: *arXiv preprint arXiv:1707.06875* (2017).

[31] OpenGPT-X. *Progress Report: Towards European LLMs*. Tech. rep. OpenGPT-X, 2024. URL: https://opengpt-x.de/en/models/teuken-7b/.

[32] openGPT-X. *Teuken-7B-instruct-commercial-v0.4 Model Card*. Hugging Face. 2024. URL: https://huggingface.co/openGPT-X/Teuken-7B-instruct-commercial-v0.4.

[33] Optuna Development Team. *Efficient Optimization Algorithms*. https://optuna.readthedocs.io/en/stable/tutorial/10_key_features/003_efficient_optimization_algorithms.html. Accessed: 2025. 2023.

[34] Llukan Puka. "Kendallâ€™s tau". In: *International encyclopedia of statistical science*. Springer, 2011, pp. 713–715.

[35] Victor Sanh et al. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter". In: *arXiv preprint arXiv:1910.01108* (2019). URL: https://arxiv.org/abs/1910.01108.

[36] Roy Schubiger. "German summarization with large language models". MA thesis. ETH Zurich, 2024.

[37] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. "BLEURT: Learning robust metrics for text generation". In: *arXiv preprint arXiv:2004.04696* (2020).

[38] Neelabh Sinha, Vinija Jain, and Aman Chadha. *Are Small Language Models Ready to Compete with Large Language Models for Practical Applications?* 2025. arXiv: 2406.11402 [cs.CL]. URL: https://arxiv.org/abs/2406.11402.

[39] Emma Strubell, Ananya Ganesh, and Andrew McCallum. "Energy and Policy Considerations for Deep Learning in NLP". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics. 2019, pp. 3645–3650. URL: https://aclanthology.org/P19-1355/.

[40] Hugo Touvron et al. "Llama 2: Open foundation and fine-tuned chat models". In: *arXiv preprint arXiv:2307.09288* (2023).

[41] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).

[42] Iulia Turc et al. "Well-read students learn better: On the importance of pre-training compact models". In: *arXiv preprint arXiv:1908.08962* (2019).

[43] Fali Wang et al. *A Comprehensive Survey of Small Language Models in the Era of Large Language Models: Techniques, Enhancements, Applications, Collaboration with LLMs, and Trustworthiness*. 2024. arXiv: 2411.03350 [cs.CL]. URL: https://arxiv.org/abs/2411.03350.

[44] Borui Xu et al. *Evaluating Small Language Models for News Summarization: Implications and Factors Influencing Performance*. 2025. arXiv: 2502.00641 [cs.CL]. URL: https://arxiv.org/abs/2502.00641.

[45] Borui Xu et al. "Evaluating small language models for news summarization: Implications and factors influencing performance". In: *arXiv preprint arXiv:2502.00641* (2025).

[46] Weizhe Yuan, Graham Neubig, and Pengfei Liu. "Bartscore: Evaluating generated text as text generation". In: *Advances in neural information processing systems* 34 (2021), pp. 27263–27277.

[47] Xianyang Zhan et al. *SLM-Mod: Small Language Models Surpass LLMs at Content Moderation*. 2025. arXiv: 2410.13155 [cs.CL]. URL: https://arxiv.org/abs/2410.13155.

[48] Tianyi Zhang et al. "Benchmarking large language models for news summarization". In: *Transactions of the Association for Computational Linguistics* 12 (2024), pp. 39–57.

[49] Tianyi Zhang et al. "Bertscore: Evaluating text generation with bert". In: *arXiv preprint arXiv:1904.09675* (2019).

[50] Wei Zhao et al. "MoverScore: Text generation evaluating with contextualized embeddings and earth mover distance". In: *arXiv preprint arXiv:1909.02622* (2019).