UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Meeri-Ly Muru

Document-Level Text Simplification in Estonian Using Large Language Models

Bachelor's Thesis (9 ECTS)

Supervisor:
Eduard Barbu, PhD

Tartu 2025

Document-Level Text Simplification in Estonian Using Large Language Models

Abstract:

The goal of text simplification is to improve text comprehensibility. It can be beneficial for people with reading disabilities or language learners. This thesis investigates Estonian text simplification at the document level by utilizing large language models and different prompt design strategies. The aim is to evaluate how various approaches influence the effectiveness of text simplification. The outputs generated by the models were assessed using both automatic evaluation methods and qualitative analysis.

Keywords: large language models, artificial intelligence, natural language processing

CERCS: P176 Artificial intelligence

Eestikeelse teksti lihtsustamine dokumendi tasemel rakendades suuri keelemudeleid

Lühikokkuvõte:

Teksti lihtsustamise eesmärk on tõsta teksti arusaadavust. See on abiks lugemisraskustega inimestele või keeleõppijatele. Lõputöö uurib eestikeelse teksti lihtsustamist dokumendi tasemel rakendades suuri keelemudeleid ning erinevaid viipe loomise strateegiaid. Eesmärk on hinnata, kuidas erinevad lähenemised mõjutavad teksti lihtsustamise tulemuslikkust. Mudelite väljundeid hinnati automaatsete meetodite ning kvalitatiivse analüüsi abil.

Võtmesõnad: suured keelemudelid, tehisintellekt, loomuliku keele töötlus

CERCS: P176 Tehisintellekt

Contents

Introduction

Text simplification is a valuable natural language processing task that aims to produce a simplified version of the original text to enhance readability. This can benefit those with reading difficulties, like people with autism (Gooding, 2022) or aphasia (Levy et al., 2012), who struggle with complex texts. In addition, even proficient readers can struggle to comprehend specialized texts, and given the exponential growth of specialized content available online, can benefit from more concise versions. Beyond improving accessibility, text simplification has shown promise as a preprocessing step for various downstream natural language processing tasks (Niklaus et al., 2016; Štajner & Popovic, 2016).

Most prior text simplification research has focused on simplifying sentences in isolation. While sentence-level simplification reduces local complexity, it fails to produce a coherent simplification on a larger scale, for example a research paper. The task of document-level text simplification, as formally defined by (Sun et al., 2021), serves as a baseline for the research conducted in this thesis.

Traditionally, text simplification has been done by developing supervised models that learn on parallel corpora of original and simplified texts. As this approach necessitates a lot of data, recent advances in natural language processing have shifted from task-specific models to generalized large language models, that can be instructed using natural language (Liu et al., 2021). Large language models have demonstrated the ability to perform on par with fine-tuned task-specific models, all while requiring little to no additional training data. (Brown et al., 2020). This development has catalyzed substantial research exploring novel prompting techniques, such as multi-agent frameworks and zero-shot prompting strategies, that continually advance natural language processing capabilities (Xi et al., 2023).

This thesis aims to investigate and evaluate these innovative prompting strategies in the context of document-level simplification. Following a comprehensive review of relevant theoretical foundations in text simplification, large language models, and prompting techniques, the thesis details the experimental setup and execution procedures. The goal is to examine how well large language models can perform document-level text simplification in Estonian, a low-resource language, using different prompting strategies.

1. Text Simplification

Text simplification is a natural language processing (NLP) task that reduces the linguistic complexity of a text, without loss of important information, while maintaining its intended meaning (Siddharthan, 2014). The main goal is to make texts accessible to wider audiences.

1.1 Beneficiaries of Simplified Text

Simplified texts can support individuals with learning difficulties or low literacy, including people with conditions such as aphasia, autism spectrum disorder, or dyslexia. Additionally, simplified content can be a valuable resource for language learners, as it enables them to develop their language skills by reading texts in their target language that are written in a simplified format.

Individuals with aphasia, a language impairment that often results from stroke or brain injury, experience difficulties with long texts and uncommon words (Siddharthan, 2014). Even when people with aphasia used to be readers before getting diagnosed, they struggle with complicated texts and have difficulty remembering what they have just read (Kjellén et al., 2017).

Access to simplified texts can help individuals with aphasia to engage with written material that would otherwise be too complex to comprehend. By presenting information in a clearer and more accessible format, simplification enhances their ability to read independently, supports information retention, and promotes autonomy.

In addition to its role in enhancing accessibility, text simplification has also proven beneficial in various NLP tasks such as parsing, information extraction, and machine translation (Chandrasekar et al., 1996). Simplification can be a pre-processing step to reduce ambiguity in input data, thereby enhancing the accuracy of NLP models, particularly in low-resource fields.

1.2 Text Simplification Methods

Text simplification mainly focuses on two tasks: lexical simplification and syntactic simplification.

1.2.1 Lexical

Lexical simplification focuses on vocabulary and replaces uncommon, complex words with more common alternatives that are easier for a wider audience to understand (Štajner & Saggion, 2018).

When readers encounter an unfamiliar word in a sentence, they may miss critical information necessary for full comprehension. This can disrupt the overall understanding of the text, particularly for individuals with limited vocabulary or language proficiency. To address this, lexical simplification not only substitutes individual words with simpler synonyms but can also replace multi-word expressions or phrases with simpler alternatives (Shardlow, 2014). These substitutions aim to preserve semantic integrity while enhancing clarity for the reader.

Lexical substitution is typically carried out in a three-step process: (1) identifying complex or difficult words within the text, (2) generating a list of potential simpler alternatives, and (3) selecting the most appropriate substitution that maintains the original meaning (Paetzold & Specia, 2017). This ensures that simplification enhances accessibility without introducing ambiguity or distortion in meaning.

1.2.2 Syntactic

Syntactic simplification aims to improve readability and comprehension by reducing structural complexity in sentences, often by eliminating or transforming syntactic constructions that are challenging to process (Štajner & Saggion, 2018). This may involve breaking long or compound sentences into multiple shorter ones or reordering the sentence.

1.3 Sentence-Level Simplification

Sentence-level text simplification focuses on simplifying individual sentences in isolation, without considering broader context. Each sentence is processed separately, often using predefined rules or models that assume all input requires simplification. However, this approach has its limitations. Not all sentences benefit from simplification - for instance, a short, five-word sentence may already be in its simplest form. Attempting to simplify such sentences may result in unnecessary paraphrasing that offers no gain in clarity. To address this issue, some systems incorporate filtering parameters, such as minimum sentence length thresholds (e.g., simplifying only sentences longer than ten words), to avoid redundant or counterproductive rewrites. This ensures that simplification is applied only when necessary, preserving the quality and intent of the original text.

1.4 Text Summarization

At first glance, text summarization and document-level simplification may appear similar, as both aim to enhance the accessibility of textual information. However, they differ significantly in their purpose and methodological challenges. Text summarization condenses vital information

by extracting or generating a shorter version of the original document, primarily by removing irrelevant content (Rao et al., 2025). Summarization typically preserves the original phrasing of the text and does not modify the linguistic complexity of the retained text. In contrast, document-level simplification seeks to maintain the full informational content while rewriting the text to make it easier to read and understand.

1.5 Document-Level Text Simplification

Much of the existing research in text simplification is focused on sentence-level approaches. While effective in many cases, this method often overlooks important discourse-level features such as coherence and cohesion (Siddharthan, 2006). In contrast, document-level simplification considers the entire text, preserving meaning across sentences and maintaining the overall structure and flow of the discourse. This approach is increasingly valuable in the modern information age, where individuals are frequently required to process large volumes of complex content. The ability to simplify entire documents - such as scientific articles, technical manuals, or long-form news articles - not only enhances accessibility but also supports efficient information consumption, particularly for time-constrained readers.

Between sentence-level and document-level simplification, several intermediate approaches have emerged. One notable example is extractive summarization, which operates at the document level by identifying and selecting the most salient clauses from a text (See et al., 2017). While this method reduces content volume and highlights key information, it does not always improve the readability of the text. The extracted segments often retain their original complexity, including dense syntax or domain-specific terminology. As a result, while extractive summarization can be a useful pre-processing step, it is often insufficient as a standalone simplification strategy, especially in contexts where both content reduction and linguistic simplification are required for optimal accessibility.

Early attempts at document-level text simplification often relied on applying sentence-by-sentence simplification techniques to longer texts, without considering discourse. However, such approaches proved insufficient for achieving coherent and contextually appropriate simplifications at the document level. Alva-Manchego et al. (2019) demonstrated that simplifying each sentence in isolation isn't sufficient to simplify texts on the document level and proposed five cross-sentence transformation operations: sentence reordering, information addition, sentence joining, content selection, and anaphora resolution. Building on this, Sun et al. (2021) defined the set of document-level simplification operations: sentence joining, sentence splitting, sentence

deletion, sentence reordering, sentence addition, and anaphora resolution. These operations reflect the nuance involved in producing simplified texts that maintain coherence and fluency across an entire document.

1.5.1 Document-Level Simplification Operations

1. **Sentence Joining**

Sentence joining refers to the process of merging two or more sentences into a single, cohesive sentence (Sun et al., 2021). This operation is necessary when redundant or less relevant information has been removed during simplification, leaving fragments that may lack fluency or contextual continuity. Sentence joining improves the overall flow of the text and enhances coherence.

2. **Sentence Reordering**

Sentence reordering alters the original structure of the text (Sun et al., 2021). This operation may involve repositioning certain sentences or clauses. Sentence reordering often works in tandem with other transformations, such as sentence joining and deletion, which can change the organization of the text.

3. **Sentence Splitting**

Sentence splitting is the process of dividing a lengthy sentence into two or more shorter, simpler sentences (Sun et al., 2021). This operation is commonly applied when a sentence contains multiple clauses, embedded structures, or excessive length that may hinder understanding. Similar to techniques used in sentence-level simplification, sentence splitting reduces syntactic complexity and cognitive load for readers.

4. **Sentence Deletion**

Sentence deletion removes sentences that are deemed redundant or non-essential to the core meaning of the text (Sun et al., 2021). This operation is applied when the elimination of a sentence does not compromise the informational content of the document. By discarding irrelevant or repetitive material, sentence deletion contributes to a more concise and focused document.

5. **Sentence Addition**

Sentence addition adds supplementary information into a text to elaborate complex terms or concepts (Sun et al., 2021). Strategically adding explanatory content can bridge knowledge gaps, facilitating a smoother reading experience and deeper understanding.

6. **Anaphora Resolution**

An anaphor is a linguistic expression, such as a pronoun, that refers to a previously mentioned item in a text, and anaphora resolution is the process of identifying and linking these references to their correct antecedents (Mitkov, 2002). This process plays a crucial role in document-level text simplification, where structural operations such as sentence reordering, joining, and deletion alter the structure of the text, disrupting referential coherence. Without proper resolution, these changes may result in ambiguous subjects, missing referents, or unnatural repetition, all of which can reduce the effectiveness of simplification.

1.6 Previous Works on Estonian Text Simplification

So far text simplification of the Estonian language has been explored strictly on the sentence level. Peedosk (2017) developed a web application that simplifies Estonian text lexically utilizing the Estonian Wordnet to find synonyms, word2vec to determine similar words, a frequency list, Estonian foreign word lexicon and vocabulary to assess word complexity. Furthermore, a formula was created to evaluate suitable word replacements based on word complexity and similarity. A questionnaire was conducted to evaluate the simplification results, which received positive feedback. Despite the favorable feedback, Peedosk (2017) himself mentioned that in addition to lexical simplification, the application should also perform syntactic simplification.

Siider (2019) created a text simplifier that worked in two stages: analysis and transformation. In the analysis stage, morphology and syntax were examined using the EstNLTK package to assess the suitability of sentences for simplification. The simplification operations were carried out in the transformation stage. This attempt wasn't very successful, as Siider (2019) conducted a study and only 20% of participants found the sentences easier to understand.

Malva (2024) implemented existing algorithms designed for English text simplification, like DRESS, OpenNMT, fine-tuned T5, and tested them both on English and Estonian sentences. In addition, a diverse data set consisting of machine-translated, human-verified, and AI-generated examples was created. This dataset served as a valuable resource for training and testing simplification models in a low-resource language setting.

2. Large Language Models and Prompt Engineering

In recent years, large language models (LLMs) have revolutionized the field of NLP by demonstrating remarkable capabilities across a wide range of language tasks without the need for task-specific training. This shift has given rise to a new paradigm, prompt engineering, in which carefully designed textual inputs guide pre-trained models to perform tasks.

2.1 Large Language Models

LLMs are large neural networks trained on vast amounts of data, which means they can predict and produce text. These models are typically based on the transformer architecture, introduced by Vaswani et al. (2017), which increased the training speed significantly.

Radford et al. (2019) demonstrated that LLMs built on transformer architectures are successful across various NLP tasks, including text summarization, text translation and text simplification. Due to their advanced contextual comprehension and generative capabilities, transformer-based LLMs have become standard tools for modern NLP research.

One hallmark of LLMs is their scale. They often have billions of parameters (learnable weights), which significantly improves their ability to make correct predictions. For example, OpenAI's Generative Pre-trained Transformer models (GPT series) demonstrated that simply scaling up model size and training data leads to remarkable results (Brown et al., 2020). This few-shot learning ability was a turning point, illustrating how a single large model could generalize to carry out different tasks.

Before the fine-tune and pre-train paradigm gained popularity in 2017, traditional NLP pipelines relied on fully supervised learning (Liu et al., 2021). In the pre-train and fine-tune paradigm, a language model is first pre-trained on a text corpus and then fine-tuned on a particular task with thousands of labeled examples (Brown et al., 2020). This approach, while effective, demands substantial task-specific data and training for each new task, which can be difficult to achieve.

Malva (2024) reports that for Estonian (a low-resource language) sentence level simplification the creation of a custom parallel dataset, which comprised of machine-translated texts and GPT-4–generated simplifications, was not sufficient to carry out the task effectively. Despite these efforts, the limited volume and variability of training data remained a significant barrier to achieving robust NLP performance.

2.2 Prompt Engineering

The rapid advancement of LLMs has made way for a new paradigm in the NLP field - prompt engineering. Prompt engineering refers to the process of designing optimal input templates that guide a pre-trained language model to perform a task effectively, without additional fine-tuning or retraining (Liu et al., 2021).

Kew et al. (2023) tested a variety of top-performing LLMs on sentence simplification and found that they outperformed supervised text simplification models, such as MUSS (Multilingual Unsupervised Sentence Simplification), a fine-tuned model trained on large-scale parallel corpora. The prompted LLMs achieved higher scores on established evaluation metrics such as SARI and BLEU, and their outputs were often judged to be more fluent, grammatically correct, and semantically faithful to the original input. It's become apparent that prompt engineering practices can outperform complex data-driven approaches, such as pre-training and fine-tuning (Feng et al., 2023; Jimenez-Romero et al., 2025). Their research suggests that prompt-based methods are not only more flexible and cost-effective, as they require no additional training, but also capable of generalizing well across tasks and languages, including low-resource scenarios.

However, prompt-based use of LLMs is not without challenges. Designing an effective prompt can be non-trivial – the phrasing and details of the prompt can significantly affect the output. This has led to the development of the field of prompt engineering itself, wherein researchers devise techniques to optimize prompts for reliability and accuracy. Modern NLP is increasingly moving from a paradigm of "collect data and fine-tune" to "use a pre-trained model and prompt it appropriately." This shift also opens new possibilities such as interactive prompts and multi-stage prompting that leverages the model's strengths without altering its weights.

2.3 Prompting Strategies

Prompt engineering encompasses a range of techniques for eliciting the best performance from LLMs. In this section, we highlight two major prompting strategies used in the study: zero-shot prompting and multi-agent prompting.

2.3.1 Zero-Shot

Zero-shot prompting is a straightforward prompting strategy. The model is given no examples in the prompt, only a task description. For example, a zero-shot prompt for text summarization might be: "Summarize the following article in one paragraph:" followed by the article text. The model must rely entirely on its pre-trained knowledge and understanding of the instruction to

generate the correct output. The value of zero-shot prompting is that no task-specific data or fine-tuning is required at all. With the emergence of LLMs, this has become a feasible technique but was unheard of in earlier NLP systems.

Brown et al. (2020) notably demonstrated the zero-shot capability of GPT-3: for many tasks, simply instructing the model produced accurate results without any training on those tasks. The success hinges on the model's extensive pre-training. The effectiveness of zero-shot prompting has been verified in a variety of NLP tasks. For instance, Feng et al. (2023) reported that GPT-3.5 in a zero-shot setting achieved a SARI score of 44.7 on a simplification benchmark, outperforming many prior systems.

Zero-shot prompting has the advantage of no cost to set up (besides creating the prompt) and uses the model as is. The drawback is that it may not always be as accurate as having seen examples – the model might misinterpret the task or produce inferior outputs if the prompt is not precise.

In summary, zero-shot prompting is a cornerstone technique that exploits LLMs' generalization ability. It eliminates the need for any training data, making it a highly efficient approach for various NLP tasks.

2.3.2 Multi-Agent

While zero-shot and few-shot prompting refer to the number of task demonstrations provided within a single prompt, multi-agent prompting represents a more advanced and structured strategy. In this approach, a complex task is decomposed into multiple subtasks, each handled by a separate prompt or agent. This modularity allows for greater specificity and control, as each subtask can be addressed with a tailored prompt that focuses on a narrower objective. Such decomposition has been shown to improve the overall quality and coherence of the output, particularly in tasks that involve multiple layers of reasoning or transformation.

The continued research in this area (Brown et al., 2020; Chen et al., 2023; Liu et al., 2021) has shown a lot of promise in achieving state-of-the-art NLP performance utilizing LLMs. This strategy has been successfully applied to document-level text simplification by Fang et al. (2025), who demonstrated that employing a coordinated set of agents, each responsible for a distinct aspect of the simplification process, led to more structured and semantically faithful outputs. Their work serves as a key inspiration for the present study, which similarly explores multi-

agent prompting as a means of enhancing the performance and consistency of document-level simplification using large language models.

3. Experimental Setup

This chapter outlines the experimental framework used to evaluate document-level simplification in Estonian using large language models. A dataset consisting of Estonian Wikipedia articles was used as example articles. Three LLMs were evaluated on their effectiveness in generating simplified text utilizing a one-shot multi-agent framework.

3.1 Model Selection

For this study, three state-of-the-art LLMs were selected based on their widespread adoption and demonstrated effectiveness across numerous NLP tasks. These models have consistently exhibited strong performance and represent diverse development approaches and deployment models.

3.1.1 GPT-4

GPT-4[1], developed by OpenAI, is currently one of the most widely used LLMs. It has demonstrated strong capabilities across a wide range of NLP tasks, including summarization, translation, and simplification. In this study, GPT-4.1 was deployed.

3.1.2 Llama 3

Llama 3[2] is an open-source language model developed by Meta, which emphasizes high performance combined with broad accessibility for the research community. Due to its openness and flexibility, Llama 3 is particularly suited for research purposes, offering opportunities for customization through fine-tuning and adaptation via prompt engineering. In this study, the model Llama 3.3 70B Instruct was deployed.

3.1.3 Gemini 2

Gemini[3] is a LLM developed by Google DeepMind. Gemini has demonstrated impressive performance across linguistic and reasoning-focused benchmarks. In this study, the model Gemini 2.0 Flash was deployed.

---

[1] https://openai.com/gpt-4/

[2] https://www.llama.com/models/llama-3/

[3] https://gemini.google.com/

3.2 Access and Configuration

This section describes the experimental environment and configuration details employed in this study. All three language models were accessed through the OpenRouter[4] platform, and interactions with these models were managed using custom Python[5] scripts. These scripts automated key tasks such as formatting prompts, querying models, and systematically collecting and storing responses.

3.2.1 OpenRouter

The OpenRouter platform provides a standardized interface for interacting with both commercial and open-source LLMs. Utilizing OpenRouter allowed uniform querying through a single API endpoint, simplifying the integration and comparison process across different model providers. Authentication, request handling, and response retrieval were executed using Python scripts integrated with the OpenAI SDK (Software Development Kit). All of the scripts are available on this GitHub Repository[6]. Since the scope of this research did not include model fine-tuning, model-specific parameters were left at their default settings.

3.2.2 Prompting

A central objective of this research was to examine the effectiveness of an agentic workflow, which necessitates structured, multi-stage prompting. For this approach, prompt design and management were carried out using a combination of plain-text (.txt) files and structured configuration (.json) files. Individual prompt templates were stored as separate .txt files, which allowed convenient iterative editing in the preliminary experiment stage.

During each experimental run, the script CreateAgents.py was executed to integrate any modifications to the prompt templates, ensuring that all updates were consistently reflected in the agent configurations. After each run, a dedicated directory was generated to store the original articles, outputs from each agent, and the corresponding agents.json configuration file. This organizational structure enabled effective version control and transparent tracking of the changes. All interactions with the language models, including prompt-response logging and evaluation

[4] https://openrouter.ai/
[5] https://www.python.org/
[6] https://github.com/meerimuru/ETTextSimpDoc

procedures, were conducted within a controlled scripting environment to ensure reproducibility and accurate documentation of the experimental process.

3.3 Dataset Overview

The dataset employed in this study consists of 4000 Estonian Wikipedia[7] articles, initially collected as part of the EKTB-55 Estonian Text Simplification Project (Barbu et al., 2025). For inclusion in the original corpus, only articles available in both Estonian and English were considered. Alongside the textual content, relevant metadata were also retrieved, including language, title, URL, Wikidata Item ID, page ID, revision ID, and revision date.

3.4 Automatic Evaluation

To evaluate the performance of the language models and prompting strategies on document-level simplification, three different automatic evaluation metrics were employed. Two of the evaluation metrics used in the study (BERT-S and D-SARI) are reference-based, meaning they require a simplified version of each article to serve as ground truth. These metrics assess the quality of the model-generated simplifications by comparing them against reference texts.

3.4.1 Reference Documents

A representative test set was constructed from Estonian Wikipedia articles corpus. From a total of 4,000 articles, 15 articles were randomly sampled for evaluation. The selection was filtered to include texts ranging between 50 and 250 words, ensuring sufficient complexity for meaningful simplification while maintaining manageability for both model processing and human annotation.

Each of the 15 sampled articles was manually simplified by the author. The manual simplification process followed a hybrid approach: first, a guideline was generated using one of the constructed agents to summarize the article's key concepts and domain-specific terminology. This made understanding the article easier.

In addition, the project EKTB-55 (Barbu et al., 2025) produced a simplified version of the Wikipedia article dataset, which included sentence-level simplifications for sentences exceeding 15 words. Where applicable, these pre-existing simplifications were incorporated into the references, provided they aligned with the overall simplification. The guidelines and

[7] https://en.wikipedia.org/wiki/Estonian_Wikipedia

transformations described in by Sun et al. (2021) (sentence splitting, joining, deletion, addition, reordering, and anaphora resolution) were also referenced to ensure coverage of document-level simplification operations.

The decision to simplify the articles entirely by hand was intentional. Using the output of one of the agentic frameworks or the output of a single prompt as a base for manual post-editing could have introduced bias. This could have yielded results that are favorable toward one of the workflows. To maintain neutrality and avoid reinforcing the strengths of a specific prompting framework, the reference simplifications were created independently, without reliance on system-generated outputs.

3.4.2 BERT-S

BERTScore (BERT-S) (Zhang et al., 2020) is an automatic text generation evaluation metric that measures semantic similarity between texts. Unlike traditional metrics that rely on exact token matches, BERT-S computes similarity through BERT's (Bidirectional Encoder Representations from Transformers) contextualized token embeddings and compares these embeddings between the candidate and reference to assess how well the generated text preserves meaning.

Document-level text simplification often involves substantial rephrasing: replacing complex terms with simpler synonyms and altering sentence structure by splitting or reordering sentences, all while aiming to preserve the original meaning. Traditional n-gram overlap metrics like BLEU are brittle in this setting, as they rely on surface-form similarity and therefore may undervalue a good simplification simply because it uses different phrasing than the reference.

As BERT-S evaluates the semantic equivalence between the original and simplified text, it aligns well with the goals of text simplification. A good simplification should convey the same information in a simpler form. That's exactly what BERT-S measures. This semantic focus makes BERT-S a great choice for document-level text simplification evaluation, as it provides a close approximation to human judgments of meaning preservation.

BERT-S was calculated by using the bert_score [8] package.

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

3.4.3 D-SARI

The Document-SARI (D-SARI) metric, proposed by Sun et al. (2021), is an adaptation of the widely used SARI metric Xu et al. (2016), specifically designed for evaluating simplification at the document level. While SARI has become a standard for sentence-level text simplification evaluation, it is limited in its ability to capture simplification quality in longer, multi-sentence inputs. D-SARI addresses this limitation by extending SARI's evaluation framework to handle structured documents and paragraph-length texts. The original SARI metric evaluates simplification quality by comparing the system output not only to the reference simplification but also to the source sentence. It measures performance across three core operations:

1. **Additions**: Words or phrases added in the simplified version that are not present in the source, but appear in the reference. This encourages the addition of clarifying or supportive content.

2. **Deletions**: Words or phrases removed both from the source and the reference. This captures the simplification of content by omission.

3. **Keeps**: Words or phrases retained in the system output and the reference. This reflects preservation of meaning.

Each operation is evaluated using precision, recall, and F1 scores, and the final SARI score is computed as the average of the three operation-specific F1 scores. SARI incorporates the input sentence into the evaluation, making it more suitable for simplification evaluation tasks.

D-SARI extends this methodology to document-level by applying SARI across sequences of sentences, rather than single sentences. D-SARI computes sentence-level SARI scores for each pair of source and simplified sentences within a document and then aggregates the results.

D-SARI was calculated by using the D_SARI.py[9] Python script.

3.4.4 FKGL

The Flesch-Kincaid Grade Level (FKGL) (Kincaid et al., 1975), initially designed for the US Navy personnel is a widely used readability formula that estimates the educational grade level required to comprehend a given text. The score is expressed on a scale that typically ranges from 0 to 12, where lower values indicate simpler texts and higher values more complex material. For

[9] https://github.com/RLSNLP/Document-level-text-simplification/blob/main/D_SARI.py

instance, a score of 6 suggests the text is suitable for readers at a sixth-grade reading level, while a score above 12 indicates advanced academic or technical writing. FKGL is calculated based on two features: average sentence length and average syllable count per word.

FKGL is a non-reference-based metric, as it does not require reference simplifications, making it especially convenient for assessing readability when gold-standard outputs are not available.

In text simplification research, FKGL has been used to evaluate simplicity and readability. FKGL correlates reasonably well with human judgments of textual simplicity (Scialom et al., 2021). However, FKGL is not without limitations. Kew et al. (2023) and Tanprasert & Kauchak (2021) argue that FKGL does not reliably capture true simplification quality and it can be easily manipulated, especially in cases where meaning preservation or grammaticality is compromised in favor of reducing word or sentence length.

Despite these criticisms, FKGL remains a common baseline metric in simplification studies due to its interpretability and ease of use. In this study, FKGL is used to provide a rough, quantitative estimate of output readability, particularly in the absence of human evaluation. While it does not capture deeper semantic or structural qualities of simplification, it offers insight into whether a system produces text that is measurably simpler. When interpreted alongside semantic metrics such as BERT-S and D-SARI, FKGL contributes to the multi-faceted evaluation of simplification quality.

The FKGL score was calculated by using the textstat[10] Python package.

To evaluate all of the simplifications, the run_Eval.py script was run, which can be found in the thesis' GitHub Repository.

[10] https://github.com/textstat/textstat

4. Experiments and Results

This chapter describes the experimental setup and evaluation results for document-level text simplification in Estonian, conducted using three LLMs and two distinct prompting strategies. The first prompting approach, Strategy A, utilizes a single-pass prompt, where the model is instructed to simplify the entire document in one step. In contrast, Strategy B implements a multi-agent prompting framework, wherein the simplification task is split into a sequence of subtasks, each handled by a dedicated agent. Strategy B is further divided into two configurations: one employing a four-agent architecture with a shared guideline-generating agent, and another following a three-agent linear pipeline without an overarching guideline.

This experimental design enables a systematic comparison of prompting strategies and model performance under controlled conditions. All three LLMs were evaluated using the same test set of documents and task prompts. The outputs generated by each model-strategy combination were assessed using both automatic evaluation metrics and qualitative observations.

4.1 Experiment Overview

This study draws upon the framework proposed by Fang et al. (2025), which introduced a multi-agent paradigm for text simplification involving nine distinct agent roles tailored to different subtasks. These roles included: Project Director, Article Logic Analyst, Content Simplifier, Simplify Supervisor, Metaphorical Analyst, Terminology Interpreter, Content Integrator, and Article Architect. In addition to defining agent responsibilities, Fang et al. (2025) proposed two interaction strategies between agents: pipeline-style communication and synchronous communication. Based on their findings, the pipeline-style approach yielded higher-quality simplifications. This study adopts the pipeline-style communication model to structure the agentic workflow. Out of the nine proposed roles, three were taken as a baseline for the agents: Project Director, Content Simplifier and Terminology Interpreter.

To better align with the document-level simplification objectives of this research, the Content Simplifier role was redefined as the Structural Simplifier. Additionally, a new agent, the Anaphora Agent, was introduced to address challenges specific to discourse-level coherence in long-form simplification tasks.

4.2 Agents

This study implements a four-agent framework for document-level text simplification, drawing inspiration from the role-based prompting approach proposed by Fang et al. (2025). Each agent is

assigned a specific task in a pipeline-style communication paradigm, where outputs from earlier agents inform the input of subsequent ones. The agents were designed to support structural, lexical, and discourse-level transformations, enhancing both the fluency and coherence of the documents. The following outlines the five agents employed and their respective responsibilities:

1. **Project Director** (PD): The Project Director is responsible for generating a simplification guideline that serves as the foundation for the prompts used by all downstream agents. Fang et al. (2025) originally proposed seven elements for the guideline. In this study, however, only four elements are implemented: Executive Summary, Terminology, Main Arguments and Evidence, and Cultural and Social Context. The excluded elements (Target Audience, Style, Emotions and Attitudes) were omitted due to contextual irrelevance. Target Audience was excluded because Wikipedia does not define a singular or fixed audience. Emotions and Attitudes, Style and Tone were deemed inappropriate given the formal, scientific nature of Estonian Wikipedia articles.

2. **Terminology Interpreter** (TI): The Terminology Interpreter is tasked with identifying and clarifying complex or domain-specific terms in the source text. The goal is to ensure that uncommon or technical vocabulary is either explained or replaced with simpler alternatives.

3. **Structural Simplifier** (SS): Based on the Content Simplifier role described by Fang et al. (2025), this agent was reconceptualized as the Structural Simplifier to better suit the goals of this study. The Structural Simplifier is responsible for modifying the syntactic structure of the text through operations such as sentence splitting, joining, deletion, and reordering.

4. **Anaphora Agent** (AA): The Anaphora Agent performs anaphora resolution, identifying and clarifying pronouns and other referential expressions to ensure that the simplified text maintains cohesion and referential clarity. This role was motivated by findings from Sun et al. (2021), which emphasize that resolving anaphoric references is a critical step in document-level simplification.

4.3 Prompting Strategy A

Prompting Strategy A adopts a single-prompt approach (see Appendix 1 for the full prompt), wherein the task of document-level simplification is presented concisely within a single instruction . The instructions provided in this strategy were crafted to align closely with the objectives of the more elaborate multi-agent Strategy B, thereby ensuring comparability. Unlike Strategy B, where the simplification process is decomposed into distinct subtasks executed by specialized agents, Strategy A tasks the language model with performing all required

simplification transformations simultaneously. This setup serves as a baseline for assessing the effectiveness and limitations of simplification via single-pass prompting methods.

An example of the simplification results can be seen in Figure 1:

Original: Wohlleben algatas Hümmelis ka kaks eripärast kalmistumetsa inimestele, kes eelistavad matmispaika põlismetsas. Väljaspool neid matmispaiku sai Hümmeli metskond rahvusvahelise metsahoolekogu FSC tunnustuse ja metskond vastab Saksa looduslähedase metsahoolde liidu kriteeriumidele.

↓

SP: Wohlleben rajas Hümmelis ka kalmistumetsad inimestele, kes soovivad matmispaika looduslikus metsas. Hümmeli metskond sai rahvusvahelise metsahoolekogu FSC tunnustuse ja vastab Saksa looduslähedase metsahoolde liidu nõuetele.

Figure 1. Single-Pass (SP) simplification performed by Gemini 2.0 Flash.

4.3.1 Quantitative Results

Model	FKGL	BERT-S	D-SARI
GPT-4.1	10.34	89.76	22.73
Llama-3.3	9.65	89.47	**30.95**
Gemini-2.0	**8.69**	**89.90**	30.41
Mean	9.56	89.71	28.03

Table 1. Results for combined evaluation across simplification metrics. Best values per metric are in bold.

As shown in Table 1, GPT-4.1 outperforms Llama 3.3. Gemini 2.0 outperforms the other models across two out of the three metrics, but only marginally. All models perform well on the BERT-S score, indicating that the single-pass prompting strategy strikes a good balance between simplification, readability, and semantic retention. While GPT-4.1 maintains strong semantic similarity (BERT-S), it underperforms on D-SARI, suggesting it may be too conservative or verbose in its simplifications.

4.3.2 Qualitative Observations

While the quantitative results for the single-pass strategy may appear favorable, the qualitative analysis reveals several shortcomings that limit the practical effectiveness of this approach.

Most notably, the output produced by GPT-4.1, despite achieving high semantic similarity scores (BERT-S), often lacks the structural simplification and clarity expected of document-level simplification. This aligns with the quantitative observation of its relatively high FKGL and low D-SARI scores, suggesting that the model tends to preserve the complexity of the original text without sufficient transformation.

Across all models, there is significant variation in simplification strategies, with no consistent pattern of operations such as sentence splitting, reordering, or deletion. For instance, Llama 3.3 demonstrates the most aggressive transformation behavior, frequently restructuring content to produce more concise and readable outputs. In contrast, GPT-4.1 often returns outputs that are only marginally altered, indicating a tendency toward minimal intervention.

Importantly, none of the models consistently applied sentence reordering, a key operation in document-level simplification aimed at improving coherence and logical flow. The outputs instead reflect a range of simplification behaviors. From largely unchanged reproductions to overly reduced versions, without a clear or reproducible transformation pattern. This inconsistency undermines the reliability of the single-pass approach for producing controlled or predictable simplification outcomes, particularly for more complex documents.

4.4 Prompting Strategy B

Prompting Strategy B implements a structured multi-agent pipeline approach, where each agent's output serves as the input for the subsequent agent. Two distinct pipeline variants were explored to evaluate how different agent configurations influence simplification quality.

Variant I

Variant I employs a pipeline consisting of the following agents: Terminology Interpreter (TI), Structural Simplifier (SS), and Anaphora Agent (AA) (Appendix 2). Fang et al. (2025) initially proposed a pipeline sequence consisting of Content Simplifier (CS), Metaphorical Analyst (MA), and Terminology Interpreter (TI). However, after preliminary experimentation and considering insights from the EKTB-55 project Barbu et al., 2025, which demonstrated that lexical simplification preceding syntactic simplification produced superior outcomes at the sentence-level for Estonian texts, the pipeline sequence was modified.

Specifically, the originally proposed Content Simplifier (CS) was redefined as a dedicated Structural Simplifier (SS) to handle exclusively document-level structural operations such as sentence deletion, splitting, joining, and reordering. The Metaphorical Analyst (MA) role

was excluded to avoid the inadvertent introduction of additional complexity resulting from metaphorical elaboration, which tended to lengthen sentences and reduce readability. The final agent sequence adopted in Variant I is shown in Figure 2:

Figure 2. Pipeline: Original Text (O), Terminology Interpreter (TI), Structural Simplifier (SS), Anaphora Agent (AA), Simplified Document (S)

The process of simplification results across this pipeline can bee seen in Figure 3:

Original: Tekstiil on materjal, mis koosneb loodusliku või kunstliku materjali kiudude (niidi, lõnga) võrgustikust. Tekstiile valmistatakse kangakudumise, silmuskudumise, heegeldamise, viltimise või sõlmimise (makramee) teel.

↓

TI: Tekstiil on riie, mis on tehtud kas looduslikest või tehismaterjalidest kiududest (niitidest, lõngadest). Tekstiile saab teha erinevatel viisidel, näiteks kangast kududes, silmuseid kududes, heegeldades, viltides või sõlmi tehes (makramee).

↓

SS: Tekstiil on riie, mis on tehtud looduslikest või tehismaterjalidest kiududest. Tekstiile saab teha kududes, silmuseid kududes, heegeldades, viltides või makramee tehnikas.

↓

AA: Tekstiil on riie, mis on tehtud looduslikest või tehismaterjalidest kiududest. Tekstiile saab valmistada kududes, silmuseid kududes, heegeldades, viltides või makramee tehnikas.

Figure 3. An extract of the Pipeline-Only simplification process simplified by the Gemini 2.0 Flash model.

As demonstrated, the Terminology Interpreter (TI) agent performs lexical simplifications and elaborations, enhancing clarity by substituting complex terms with simpler alternatives (e.g., ”materjal” to ”riie,” ”koosneb” to ”on tehtud”). Subsequently, the Structural Simplifier (SS) agent refines the paragraph further by removing redundant content. The Anaphora Agent (AA) largely remains inactive during this process, which aligns with its designated role, taking action only if referential ambiguities are detected.

Variant II

Variant II introduces a fifth agent, the Project Director (PD) (Appendix 3), to the previously

described pipeline. Fang et al. (2025) defined the Project Director as an agent responsible for generating comprehensive guidelines to direct the simplification process, highlighting essential content and key elements of the original text. In this variant, the Project Director's guidelines explicitly outline the primary concepts, terminology, main arguments, and relevant contextual details for subsequent agents to follow. An example of the generated guidelines can bee seen in Figure 4:

Guidelines: **1. Tööülevaade**
Dokumendi tööülevaade sisaldab Bernalda linna kirjeldust Itaalias, selle ajaloolist tausta, geograafilisi andmeid ning ülevaadet kohalikust kultuurist. Dokumendi eesmärk on võimaldada lugejatel paremat arusaamist Bernalda kohta ning eskiimi dokumentide lihtsustamisel pea ees hoida linna üksikasju.

2. Terminoloogia
* Comune: Itaalia omavalitsusüksus
* Omavalitsusüksus: Haldusüksus, mis omab teatud tasemel iseseisvust
* Fraktsioon: Haldusüksuse osa, mis võib olla iseseisev asula või piirkond

3. Peamised argumendid ja tõendid
Dokumendi peamised argumendid:
- Bernalda asukoht ning geograafilised andmed
- Linna ajalugu, sealhulgas Metapontumi linn

4. Kultuuriline ja sotsiaalne kontekst
Dokumendis on esindatud Bernalda kultuuriline ja sotsiaalne kontekst läbi linna ajaloo, seal elava rahvastiku ning selle ümbruskonna kirjeldused.

Figure 4. Guidelines generated by the Gemini 2.0 Flash model with the PD agent. Note that some of the text has been reducted.

The Variant II pipeline is structured as follows in Figure 5:

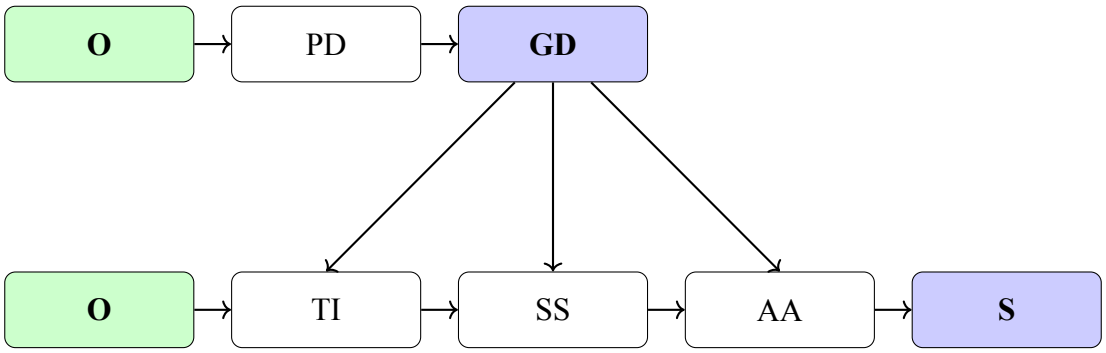

Figure 5. Original Text (O), Project Director (PD), Guidelines (GD), Terminology Interpreter (TI), Structural Simplifier (SS), Anaphora Agent (AA), Simplified Document (S)

By incorporating the guideline-generating Project Director, Variant II aims to enhance consistency and coherence throughout the simplification process, enabling downstream agents to maintain alignment with clearly established simplification objectives.

The process of simplification results across this pipeline can bee seen in Figure 6:

Original: Bernalda on linnaks nimetatav omavalitsusüksus (comune) Itaalias Basilicata maakonnas Matera provintsis. Bernalda sai linnaõigused 1735. aasta 21. juunil.

↓

TI w/ GD: Bernalda on linn, mis on Itaalias kohalik omavalitsus, nagu Eestis vald või linn. See asub Basilicata piirkonnas (Itaalia ”maakond”) Matera provintsis. Bernalda tunnistati ametlikult linnaks (linnaõigused) 21. juunil 1735. aastal.

↓

SS w/ GD: Bernalda on linn Itaalias, Basilicata piirkonnas, Matera provintsis. See on kohalik omavalitsus, nagu Eestis vald või linn. Bernalda tunnistati ametlikult linnaks 1735. aastal.

↓

AA w/ GD: Bernalda on linn Itaalias, Basilicata piirkonnas, Matera provintsis. See on kohalik omavalitsus, nagu Eestis vald või linn, ja tunnistati ametlikult linnaks 1735. aastal.

Figure 6. An extract of the Pipeline-Guideline simplification process simplified by Gemini 2.0. The simplifications were made alongside the guidelines demonstrated in Table 4.

4.4.1 Quantitative Results

Model	FKGL	BERT-S	D-SARI
GPT-4.1	10.05	**88.98**	24.20
Llama-3.3	10.467	88.26	26.48
Gemin-2.0	**9.77**	88.73	**27.37**
Mean	10.10	88.66	26.02

(a) Pipeline-Only

Model	FKGL	BERT-S	D-SARI
GPT-4.1	10.73	86.02	2.02
Llama-3.3	11.41	88.20	21.20
Gemini-2.0	**9.57**	**88.76**	**26.85**
Mean	10.57	87.66	16.69

(b) Pipeline-Guideline

Table 2. Side-by-side comparison of Pipeline-Only (a) and Pipeline-Guideline (b) across simplification metrics. Best values per metric are in bold.

In the Pipeline-Only setup, models demonstrate stronger overall performance across all metrics, as seen in Table 2. It has a slightly lower FKGL score of 10.10, indicating better readability than the PG configuration (mean FKGL: 10.57). Similarly, the mean D-SARI score is significantly higher in the PO setting (26.02) compared to PG (16.69), showing that the PO approach facilitates more effective simplification operations. Gemini 2.0 again stands out with the highest D-SARI (27.37), suggesting it applies the most balanced and impactful simplification strategies under PO.

In terms of semantic preservation, as measured by BERT-S, the Pipeline-Only approach maintains a slight edge overall (88.66 vs. 87.66). GPT-4.1 achieves the highest BERT-S score (88.98) in the PO configuration.

By contrast, the Pipeline-Guideline configuration, though intended to offer structural support and task clarity, results in longer, more complex outputs (higher FKGL) and less transformation (lower D-SARI). The most drastic decline is observed in GPT-4.1, whose D-SARI score drops sharply from 24.20 (PO) to 2.02 (PG), suggesting that the guideline-based prompts may overly constrain the model, causing it to underperform simplification-wise. This trend is consistent across all three models and indicates a systemic effect of the guideline on output style and structure.

4.4.2 Qualitative Observations

The Pipeline-Only approach results in more pronounced structural transformations, including sentence reordering and joining. Outputs generated through this method are consistently more

concise, indicating a higher degree of content condensation. In addition, there are a higher number of elaborations.

The Pipeline-Guideline approach extends the simplification process further by providing more detailed explanations, but this impacts text length, resulting in outputs that are much longer. Overly long sentences were a recurring issue across all models in this configuration.

In addition, GPT-4.1 appears to struggle with task adherence in the Pipeline-Guideline setting. Rather than producing a fully simplified document, it frequently defaults to listing the main points of the article or summarizing the content, deviating from the intended objective. Out of the 15 test documents, only 6 outputs from GPT-4.1 could be described as cohesive and appropriately simplified documents. Similar issues were observed with Gemini-2.0, though to a lesser extent, with 2 documents exhibiting similar summarization behavior. Llama 3.3, in turn, occasionally introduced phrases such as ”Here is the simplified document:”, a stylistic artifact that was not present in the Pipeline-Only approach, suggesting prompt misinterpretation.

These observations suggest that when presented with dense informational content, particularly when structured as a list, the models, especially GPT-4.1, may experience difficulty maintaining both coherence and task adherence.

4.5 Strategy Comparison

Strategy	FKGL	BERT-S	D-SARI
SP	**9.56**	**89.71**	**28.03**
PO	10.10	88.66	26.02
PG	10.57	87.66	16.69

Table 3. Comparison of average scores across three simplification strategies: Single Pass (SP), Pipeline-Only (PO), and Pipeline-Guideline (PG). Best values per metric are in bold.

When comparing the overall performance of the three prompting strategies, the Single-Pass (SP) method outperforms the multi-agent approaches across all three automatic evaluation metrics. As shown in the Table 3, SP achieves the lowest FKGL score (9.56), indicating the highest readability, and the highest BERT-S score (89.71), reflecting strong semantic preservation. It also has the highest D-SARI score (28.03), suggesting that the SP method has the most effective balance of simplification operations (additions, deletions, and content retention).

Despite higher scores, qualitative analysis revealed that the single pass approach tends to preserve much of the original article's structure, resulting in outputs with limited sentence reordering and a frequent occurrence of long, complex sentences. The simplification process in this strategy is relatively conservative, with minimal elaboration or restructuring. For example, only occasional clarifications were observed, such as GPT-4.1 elaborating the term ”kvantitatiivsed meetodid” to ”arvulisi ehk kvantitatiivseid meetodeid”, which slightly improved clarity but was not consistently applied across other terms or documents.

The Pipeline-Only approach typically produced the shortest outputs among the three strategies, focusing more aggressively on structural simplification. It more frequently employed sentence splitting and reduction techniques, resulting in more concise texts overall.

On the other hand, the Pipeline-Guideline approach demonstrated a tendency to over-explain, often expanding on terminology and background information beyond what is necessary for simplification. This led to consistently longer paragraphs and more verbose outputs, which, while informative, occasionally counteracted the goal of enhancing readability and simplicity. This behavior suggests that the additional context provided by the guideline may encourage models to include explanatory content at the cost of conciseness.

4.6 Model Comparison

Model	FKGL	BERT-S	D-SARI
GPT-4.1	10.37	88.25	16.32
Llama-3.3	10.51	88.64	26.21
Gemini-2.0	**9.34**	**89.13**	**28.21**

Table 4. Mean scores across all settings for each model.
Lowest FKGL and highest BERT-S and D-SARI are in bold.

Based on the model evaluation results in Table 4, GPT-4.1 demonstrates the weakest overall performance among the three models across all metrics. This is particularly evident in its mean D-SARI score of 16.32, which indicates limited effectiveness in performing structural and lexical simplification operations. A key factor contributing to this underperformance is the model's poor output quality under the Pipeline-Guideline configuration, where it frequently failed to apply simplification operations and instead produced outputs that resembled summarized key points rather than coherent, simplified versions of the original documents. Across all prompting

strategies, GPT-4.1 also consistently generated the longest outputs, often neglecting opportunities for sentence reduction or reorganization.

In contrast, Llama 3.3 demonstrates greater consistency and adaptability in its outputs. It had solid performance in both semantic preservation and structural editing, Llama 3.3 shows strong capability in restructuring and re-imagining the original documents. Qualitative analysis reveals frequent application of simplification strategies, including sentence reordering, elaboration, and lexical substitution.

However, Gemini 2.0 emerges as the most effective model in this evaluation. It achieves the lowest FKGL score (9.34), indicating the highest output readability, alongside the highest BERT-S (89.13) and D-SARI (28.21) scores. These results suggest that Gemini excels at maintaining semantic integrity while also producing a simplified output. It consistently applies transformation operations across prompting strategies.

5. Analysis and Reflection

This chapter presents a critical analysis of the experimental results, drawing attention to the effectiveness of the prompting strategies, the behavior of the models, and the broader implications of the findings.

5.1 Strengths and Weaknesses of Prompting Methods

Each prompting strategy examined in this study demonstrated specific advantages and limitations, which impacted both the quantitative performance and the qualitative nature of the simplified outputs. The Single Pass approach outperformed the other methods according to automatic metrics, but tended to simplify too conservatively.

The Pipeline-Guideline approach, while designed to enhance coherence and provide structural support to the agents, introduced additional complexity. It often resulted in lengthy outputs and, rather than improving simplification quality, it inhibited coherency and led to summarization-like outputs.

In contrast, the Pipeline-Only approach emerged as the most effective strategy overall. It produced the most concise and structurally simplified outputs, demonstrating stronger document-level transformations such as sentence deletion, reordering, and elaboration where appropriate. This approach achieved a better balance between simplification depth and textual coherence.

5.2 Error Analysis and Key Findings

Several challenges and patterns emerged during prompt development and model evaluation. When working with English-centric LLMs, it became clear that explicitly specifying the output language (Estonian) in each prompt was essential; otherwise, the models defaulted to English or introduced inconsistencies. In addition, it was necessary to define the expected output format clearly to reduce ambiguity in model responses.

An early issue observed during the multi-agent pipeline development was related to the initial implementation of the Content Simplifier, which was responsible for both lexical and structural transformations. The final outputs from this agent often contained overly long and complex sentences, primarily due to excessive elaboration.

The Single Pass strategy tended to be overly cautious, making minimal structural changes and largely preserving the original discourse structure. By contrast, the Pipeline-Only approach yielded the most robust document-level transformations and produced outputs that better

aligned with the goals of simplification. The Pipeline-Guideline configuration consistently underperformed. This was especially evident in the case of GPT-4.1, where outputs frequently diverged from the simplification task, leading to incoherent results. The inclusion of a guideline, in this case, did not enhance output quality and in fact introduced additional errors.

5.3 Limitations of the Study

One of the primary limitations of this study lies in the absence of human evaluation. While automatic metrics such as FKGL, BERT-S, and D-SARI were employed to assess simplification quality, they did not always align with the qualitative observations. In several cases, models that performed well quantitatively produced outputs that were either structurally unchanged or insufficiently simplified, highlighting the limitations of automatic evaluation alone.

Another limitation concerns the nature of the dataset. The corpus consisted exclusively of Estonian Wikipedia articles, which restricted the scope of simplification to factual and encyclopedic text. Narrative or fictional styles were not represented, which meant that certain agent roles, such as those proposed by Fang et al. (2025) for metaphorical analysis, could not be evaluated in this context.

Finally, the reference dataset used for evaluation was relatively small, comprising only 15 documents. Although the simplifications were conducted manually to ensure quality and accuracy, the limited sample size constrains the broader applicability of the findings. A larger and more diverse evaluation set would be necessary for a more comprehensive assessment.

5.4 Further Work

This study opens several promising directions for future research in document-level text simplification in Estonian using large language models and agentic prompting strategies. One key area for extension is the agent workflow architecture. The Structural Simplifier agent introduced in this study encompassed multiple document-level operations, such as sentence splitting, deletion, and reordering. These operations could be further modularized into separate, task-specific agents.

In terms of evaluation, while this study employed established automatic metrics, their limitations became evident when compared with qualitative assessments. Future work should incorporate human evaluation, ideally involving multiple annotators. Furthermore, the evaluation would benefit from a larger reference corpus, including multiple simplified versions per document, to

better support reference-based metrics such as D-SARI and BERT-S. Such enhancements would allow for a more reliable and comprehensive understanding of model behavior.

Lastly, further experimentation across diverse domains and genres would help evaluate the generalizability of the agentic workflow beyond encyclopedic content and explore roles for additional agents.

6. Conclusion

This thesis set out to explore the task of document-level text simplification in Estonian using large language models and modern prompting strategies. Given the limited availability of resources for Estonian, this study aimed to evaluate whether prompt-based approaches, particularly those involving structured multi-agent workflows, could produce high-quality simplifications without the need for task-specific fine-tuning.

Two distinct agentic prompting frameworks were proposed and implemented, alongside a traditional single-pass strategy. Contrary to expectations, the single-pass method performed surprisingly well, producing outputs that were often coherent and reasonably simplified. One of the agentic workflows, the Pipeline-Only approach, achieved superior performance, particularly in terms of structural transformations and elaborations.

A total of 15 Estonian Wikipedia articles were manually simplified to create reference outputs for automatic evaluation. However, the results revealed that automatic evaluation metrics did not always align with qualitative assessments. While metrics such as FKGL, BERT-S and D-SARI provided useful benchmarks, they frequently failed to capture important nuances such as coherence, readability, and informativeness observed during manual analysis.

Overall, the findings highlight that document-level simplification is a complex task, involving not just lexical and syntactic changes but also discourse-level transformations. LLMs and prompt engineering practices show strong potential in the NLP field.

Acknowledgements

I would like to thank my thesis supervisor for their support and guidance throughout the writing of this thesis.

I also extend my sincere thanks to the professors and faculty at the Institute of Computer Science, whose courses and advice have been valuable during my studies.

Lastly, I am grateful to my friends, family, and everyone who supported and cheered me on.

References

Alva-Manchego F., Scarton C., & Specia L. (2019). Cross-Sentence Transformations in Text Simplification. en.

Barbu E., Muru M.-L., & Malva S. M. (2025). Improving Estonian Text Simplification through Pretrained Language Models and Custom Datasets. arXiv:2501.15624 [cs].

Brown T. B., Mann B., Ryder N., Subbiah M., Kaplan J., Dhariwal P., Neelakantan A., Shyam P., Sastry G., Askell A., Agarwal S., Herbert-Voss A., Krueger G., Henighan T., Child R., Ramesh A., Ziegler D. M., Wu J., Winter C., Hesse C., Chen M., Sigler E., Litwin M., Gray S., Chess B., Clark J., Berner C., McCandlish S., Radford A., Sutskever I., & Amodei D. (2020). Language Models are Few-Shot Learners. arXiv:2005.14165 [cs].

Chandrasekar R., Doran C., & Srinivas B. (1996). Motivations and Methods for Text Simplification. *COLING 1996 Volume 2: The 16th International Conference on Computational Linguistics*.

Chen W., Su Y., Zuo J., Yang C., Yuan C., Chan C.-M., Yu H., Lu Y., Hung Y.-H., Qian C., Qin Y., Cong X., Xie R., Liu Z., Sun M., & Zhou J. (2023). AgentVerse: Facilitating Multi-Agent Collaboration and Exploring Emergent Behaviors. arXiv:2308.10848 [cs].

Fang D., Qiang J., Ouyang X., Zhu Y., Yuan Y., & Li Y. (2025). Collaborative Document Simplification Using Multi-Agent Systems. *Proceedings of the 31st International Conference on Computational Linguistics*. Ed. by Rambow O., Wanner L., Apidianaki M., Al-Khalifa H., Eugenio B. D., & Schockaert S. Abu Dhabi, UAE: Association for Computational Linguistics, pp. 897–912.

Feng Y., Qiang J., Li Y., Yuan Y., & Zhu Y. (2023). Sentence Simplification via Large Language Models. arXiv:2302.11957 [cs].

Gooding S. (2022). On the Ethical Considerations of Text Simplification. *Ninth Workshop on Speech and Language Processing for Assistive Technologies (SLPAT-2022)*. Ed. by Ebling S., Prud'hommeaux E., & Vaidyanathan P. Dublin, Ireland: Association for Computational Linguistics, pp. 50–57.

Jimenez-Romero C., Yegenoglu A., & Blum C. (2025). Multi-Agent Systems Powered by Large Language Models: Applications in Swarm Intelligence. arXiv:2503.03800 [cs].

Kew T., Chi A., Vásquez-Rodríguez L., Agrawal S., Aumiller D., Alva-Manchego F., & Shardlow M. (2023). BLESS: Benchmarking Large Language Models on Sentence Simplification. Ed. by Bouamor H., Pino J., & Bali K. Singapore: Association for Computational Linguistics, pp. 13291–13309.

Kincaid J. P., Fishburne J., Robert P. R., Richard L. C., & Brad S. (1975). Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel: en. Tech. rep. Fort Belvoir, VA: Defense Technical Information Center.

Kjellén E., Laakso K., & Henriksson I. (2017). Aphasia and literacy—the insider's perspective. en. *International Journal of Language & Communication Disorders* 52.5, pp. 573–584.

Levy J., Hoover E., Waters G., Kiran S., Caplan D., Berardino A., & Sandberg C. (2012). Effects of Syntactic Complexity, Semantic Reversibility, and Explicitness on Discourse Comprehension in Persons With Aphasia and in Healthy Controls. *American Journal of Speech-Language Pathology* 21, S154–S165.

Liu P., Yuan W., Fu J., Jiang Z., Hayashi H., & Neubig G. (2021). Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. arXiv:2107.13586 [cs].

Malva S. M. (2024). Exploring And Implementing Estonian Text Simplification Using Machine Learning. en. *University of Tartu Institute of Computer Science*.

Mitkov R. (2002). Anaphora Resolution. Oxford, UNITED KINGDOM: Taylor & Francis Group.

Niklaus C., Bermeitinger B., Handschuh S., & Freitas A. (2016). A Sentence Simplification System for Improving Relation Extraction. *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. Ed. by Watanabe H. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 170–174.

Paetzold G. H. & Specia L. (2017). A Survey on Lexical Simplification. en. *Journal of Artificial Intelligence Research* 60, pp. 549–593.

Peedosk M. (2017). Applying Estonian Digital Resources and Technologies in a Text Simplification Program. et. *University of Tartu Institute of Computer Science*.

Radford A., Wu J., Child R., Luan D., Amodei D., & Sutskever I. (2019). Language Models are Unsupervised Multitask Learners. en.

Rao A., Aithal S., & Singh S. (2025). Single-Document Abstractive Text Summarization: A Systematic Literature Review. eng. *ACM Computing Surveys* 57.3. Publisher: Association for Computing Machinery, pp. 1–37.

Scialom T., Martin L., Staiano J., Clergerie É. V. d. l., & Sagot B. (2021). Rethinking Automatic Evaluation in Sentence Simplification. arXiv:2104.07560 [cs].

See A., Liu P. J., & Manning C. D. (2017). Get To The Point: Summarization with Pointer-Generator Networks. arXiv:1704.04368 [cs].

Shardlow M. (2014). A Survey of Automated Text Simplification. en. *International Journal of Advanced Computer Science and Applications* 4.1.

Siddharthan A. (2006). Syntactic Simplification and Text Cohesion. en. *Research on Language and Computation* 4.1, pp. 77–109.

– (2014). A survey of research on text simplification. en. *ITL - International Journal of Applied Linguistics* 165.2, pp. 259–298.

Siider S. (2019). Text simplifier based on syntax analysis. et. *University of Tartu Institute of Computer Science*.

Štajner S. & Popovic M. (2016). Can Text Simplification Help Machine Translation? *Proceedings of the 19th Annual Conference of the European Association for Machine Translation*, pp. 230–242.

Štajner S. & Saggion H. (2018). Data-Driven Text Simplification. *Proceedings of the 27th International Conference on Computational Linguistics: Tutorial Abstracts*. Ed. by Donia Scott, Walker M., & Pascale Fung. Santa Fe, New Mexico, USA: Association for Computational Linguistics, pp. 19–23.

Sun R., Jin H., & Wan X. (2021). Document-Level Text Simplification: Dataset, Criteria and Baseline. en. arXiv:2110.05071 [cs].

Tanprasert T. & Kauchak D. (2021). Flesch-Kincaid is Not a Text Simplification Evaluation Metric. *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*. Ed. by Bosselut A., Durmus E., Gangal V. P., Gehrmann S., Jernite Y., Perez-Beltrachini L., Shaikh S., & Xu W. Online: Association for Computational Linguistics, pp. 1–14.

Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., & Polosukhin I. (2017). Attention Is All You Need. arXiv:1706.03762 [cs] version: 1.

Xi Z., Chen W., Guo X., He W., Ding Y., Hong B., Zhang M., Wang J., Jin S., Zhou E., Zheng R., Fan X., Wang X., Xiong L., Zhou Y., Wang W., Jiang C., Zou Y., Liu X., Yin Z., Dou S., Weng R., Cheng W., Zhang Q., Qin W., Zheng Y., Qiu X., Huang X., & Gui T. (2023). The Rise and Potential of Large Language Model Based Agents: A Survey. arXiv:2309.07864 [cs].

Xu W., Napoles C., Pavlick E., Chen Q., & Callison-Burch C. (2016). Optimizing Statistical Machine Translation for Text Simplification. *Transactions of the Association for*

Computational Linguistics 4. Ed. by Lee L., Johnson M., & Toutanova K. Place: Cambridge, MA Publisher: MIT Press, pp. 401–415.

Zhang T., Kishore V., Wu F., Weinberger K. Q., & Artzi Y. (2020). BERTScore: Evaluating Text Generation with BERT. arXiv:1904.09675 [cs].

Appendices

Appendix 1 Single-Pass Prompt

Single-Pass Prompt

As a text simplification writer, your task is to simplify the given document. Document should aim to retain original information. Elaborate complex terms or replace them with their simpler alternatives. You may simplify the text via document-level simplification operations:

Deleting Sentences: Delete sentences or phrases with non-crucial or repetitive information.

Joining Sentences: Deleting sentences can cause issues with fluency. If necessary, join sentences to improve fluency.

Splitting Sentences: Split long sentences into two or more sentences.

Reordering Sentences: Reorder the sentences if that improves the flow of the document.

Note that these should only be applied where and if necessary. Give only the simplified document in Estonian as output.

Appendix 2 Project Director

Project Director

You are a project director and your task is to create guidelines in Estonian for simplifying Estonian documents. The guidelines aim is to aid subsequent agents in the simplification process. The guideline should be set up as follows:

1. Executive Summary: A brief overview of the document's main points, arguments, and conclusions.
2. Terminology: A list of specialized terms, along with their definitions and contextual meanings. All uncommon words and specialized terms mentioned in the document should be explained.
3. Main Arguments and Evidence: An outline of the document's main arguments and the key evidence or examples that support these arguments.
4. Cultural and Social Context: Information about the cultural, historical, or social context mentioned in the document, to be appropriately addressed during simplification. Output should only contain the guidelines for Estonian document-level simplification.

Appendix 3 Agent Pipeline

The text in bold showcases the extra info added to all Pipeline-Guideline agents.

Terminology Interpreter

You are a terminology interpreter, who's task is simplifying Estonian documents. Document should aim to retain original information. Replace complex words or phrases with simpler alternatives to increase readability of text. Elaborate terminology.

Before simplifying look over the guidelines. The guidelines are to aid the simplification process and keep track of core information, which should not be removed.

Give only the simplified document in Estonian as output.

Structural Simplifier

Your task is to simplify the overall structure of Estonian documents while keeping the original meaning. Document should aim to retain original information. Simplify the text via syntactic simplification methods and document-level simplification operations:

Deleting Sentences: Delete sentences or phrases with non-crucial or repetitive information.

Joining Sentences: Deleting sentences can cause issues with fluency. If necessary, join sentences to improve fluency.

Splitting Sentences: Split long sentences into two or more sentences.

Reordering Sentences: Reorder the sentences if that improves the flow of the document.

Note that these should only be applied where and if necessary.

Give only the simplified document in Estonian as output.

Anaphora Agent

You are an anaphora agent, who's task is to check and edit simplified Estonian documents. Document should aim to retain original information. Go over the document and make sure all of the anaphors are in place and the text is coherent. Elaborate, where necessary. If there is repeating information, remove it.

If everything is in place, do nothing.

Give only the edited simplified document in Estonian as output.

License

Non-exclusive licence to reproduce the thesis and make the thesis public

I, **Meeri-Ly Muru**,

1. grant the University of Tartu a free permit (non-exclusive licence) to

reproduce, for the purpose of preservation, including for adding to the digital archives of the University of Tartu until the expiry of the term of copyright, my thesis

Document-Level Text Simplification in Estonian Using Large Language Models,

supervised by **Eduard Barbu**;

2. grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright;

3. am aware of the fact that the author retains the rights specified in points 1 and 2;

4. confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Meeri-Ly Muru
15/05/2025