UNIVERSITY OF GOTHENBURG

DEPARTMENT OF PHILOSOPHY,
LINGUISTICS AND THEORY OF SCIENCE

Expert in the Loop: LLM Assistance for Technical Documentation Writing

Case Study at Saab AB

Anni Nieminen

Master's thesis:	30 credits
Programme:	Master in Language Technology, 120 credits
Level:	Advanced level
Semester and year:	Spring, 2025
Supervisors:	Staffan Larsson & Asad Sayeed
Company supervisors:	Åsa Hedenskog & Erik Helander
Examiner:	Sharid Loáiciga
Keywords:	Technical Writing, Fine-tuning, LLM, Simplified Technical English, Instruction-tuning

Abstract

This study explores the potential of LLMs in the technical writing process at Saab Aeronautics. The technical writing process is investigated by interviewing technical writers, collecting insights regarding the most challenging tasks and areas where AI-assistance could be beneficial. These experts are involved in this research project in several stages with the aim of investigating how an LLM could facilitate their tasks. A demonstration dataset is collected with the help of the experts. Additionally, a parallel corpus consisting of technical procedures is created. Supervised Instruction Fine Tuning (SIFT) method is implemented for the fine-tuning of an LLM (Mistral-7b-Instruct-v.02), combining Quantized Low-rank Adaptation (QLoRA) and Low-rank Adaptation (LoRA) in order to perform the fine-tuning memory-efficiently. Sampled generations are investigated qualitatively in addition to a small-scale hyperparameter search. Both the experts involved in the data collection as well as held-out experts are involved in the evaluation stage. The results show that the fine-tuned model's outputs are preferred over the base model outputs 68% of the time. Analysis of the experts' comments reveals that the fine-tuned model outperforms the base model specifically in terms of adhering to the Simplified Technical English (STE) writing standard and by containing fewer hallucinations. This study suggests potential in fine-tuning LLMs with small, but high-quality datasets. Additionally, this study highlights the significance in involving human expertise in such processes for domain-specific needs, such as those at Saab.

Acknowledgements

I would like to thank Saab for this opportunity. My special thanks go to my thesis supervisors at Saab Aeronautics, Åsa Hedenskog and Erik Helander, whose support, encouragement, and trust in me has been amazing.

I would also like to warmly thank Anna Nygårdh-Birgersson, Mats Hermansson, and all the professionals from the technical writing team who took part in my study.

I send my warm thanks also to my supervisors at Gothenburg university, Staffan Larsson and Asad Sayeed.

Glossary

AECMA European Association of Aerospace Industries

AMP Aircraft Maintenance Publication

ASD Aerospace and Defence Industries Association of Europe

GPU Graphics Processing Unit

LLM Large Language Model

LoRA Low-rank Adaptation

PEFT Parameter-efficient Fine-tuning

QLoRA Quantized Low-rank Adaptation

RAM Random Access Memory

SIFT Supervised Instruction Fine-tuning

STE Simplified Technical English

Contents

1 Introduction 1
1.1 Research Questions 1
1.2 Thesis Organization 2
2 Background & Related Work 3
2.1 Technical Writing at Saab 3
2.1.1 ASD-STE100 Simplified Technical English 3
2.2 Large Language Models 5
2.3 Instruction Fine-tuning LLMs 7
2.4 Expert Post-editing of LLM Outputs 8
2.5 Framework for Generative AI in Technical Writing 9
2.6 Previous Thesis Work 9
3 Methodology 13
3.1 Data 13
3.1.1 Creation of the Parallel Corpus 13
3.1.1.1 Observations of the Source Data 14
3.1.1.2 Observations of the Target Data 15
3.1.2 Finalized Parallel Corpus 15
3.1.3 Meetings with the Technical Writers 16
3.1.4 Demonstration Data Collection with the Technical Writers 19
3.1.5 Finalized Demonstration Dataset 20
3.1.6 Final Complete Dataset 21
3.2 Model Choice 22
3.3 QLoRA and LoRA - Parameter-efficient Fine-tuning 22
3.4 Hyperparameter Settings and Output Sampling Results 23
3.5 Evaluation Method 26
3.5.1 Why Human Evaluation? 26
3.5.2 Evaluation Setup 27
3.6 Methodological Differences Compared to Bengtsson and Habbe (2024) 28
4 Results 29
4.1 Model Preference 29
4.1.1 General Accuracy 29
4.1.2 Helpfulness 31
4.1.3 Format 32
4.1.4 Simplified Technical English (STE) 33
4.1.5 Inter-rater Agreement 34

4.2 Feedback on Gold Outputs (AMPs) 35
4.3 Pre-study Questionnaire Results 35
5 Discussion 36
5.1 Delimitations of the Study 37
5.2 Ethical Aspects of the Study 38
6 Conclusion & Future Work 39
6.1 Addressing Research Questions 39
6.2 Future Work 40
References 42
Appendices 45
Appendix A: Pre-study Questionnaire 45
Appendix B: Example Evaluation Formulas 46

1 Introduction

Saab, founded in 1937, is a Swedish producer of various defense and civil security products and solutions (Saab, 2025). Saab operates in over 30 countries on all continents, and the company is spread across many different business areas, such as Surveillance (i.e. threat detection and localization), Dynamics (weapons, sensors, and simulation systems) and Kockums (submarines and surface vessels). This thesis is conducted in collaboration with **Saab Aeronautics**, most known for its multi-role fighter aircraft, Gripen, and various aircraft systems.

Ensuring the correct use and maintenance of Saab's defense products, it is vital that Saab's products are accompanied by high-quality technical documentation. For a majority of these products, the compiling and writing of the technical documents are done manually by specialized professionals, who at Saab are known as technical writers (*tekniska skribenter* in Swedish). With the goal of ensuring both safety and compliance with both governmental and Saab's company-specific regulations, the technical writers are required to follow various writing guidelines. When it comes to the language of the documents, the aerospace industry follows an international writing standard known as ASD-STE100 Simplified Technical English.

Following different general writing guidelines as well as project-specific rules combined with tight delivery schedules imposes high expectations on the technical writers. Additionally, as the technical writers are not necessarily required to have a background in technical work, understanding the source documents written by the maintenance engineers can sometimes be time-consuming and may require additional correspondence with them.

Our thesis explores how a domain-specific instruction-tuned Large Language Model could assist the technical writers in the writing process. To this end, together with the technical writers (also referred to as *experts* in this study), we curate an expert-annotated dataset that will be used for the instruction fine-tuning of an LLM. Our goal throughout the project is to keep the potential end users, namely the experts, in mind: we will investigate the technical writing process as a whole, interview technical writers at Saab, and collect their feedback and insights also at the evaluation stage.

We will also be investigating and presenting our solution to the source-target data alignment problem discussed in Bengtsson and Habbe (2024). This entails a careful and detailed analysis of the structure and content of the data provided for our study. The resulting parallel corpus will be combined with the expert-annotated dataset mentioned above, and the resulting dataset will be used to instruct fine-tune Mistral-7b-instruct-v.02.

1.1 Research Questions

The research questions of this study are the following:

- To what extent can fine-tuning of an instruction-tuned LLM with expert-annotated data and a curated parallel corpus produce expert-level outputs?
- What kind of qualitative evaluation do the experts provide about the fine-tuned model's performance, and in what ways does their feedback differ from that of the base model?
- By adopting a more hands-on approach, what type of solution can we offer for the source-target data alignment problem in Bengtsson and Habbe (2024)?

1.2 Thesis Organization

This study will start with an overview of related research and work in Chapter 2. We will introduce the writing standard ASD-STE100 (Simplified Technical English), a generative AI framework for technical writing, Large Language Models (LLMs), and also provide an overview of the fine-tuning process of such models. Essential to many of the aspects and decisions made throughout our study, the previous thesis work (Bengtsson and Habbe, 2024) conducted on the topic at Saab will also be presented.

Subsequently, we will introduce our study methodology in Chapter 3. This chapter will include description of the data used in our study, the training process as well as the evaluation method.

The results will be presented in Chapter 4 and discussion will follow in Chapter 5.

Lastly, Chapter 6 is where we conclude our study and discuss the delimitations of this study along with future possibilities.

2 Background & Related Work

In this chapter, we will discuss the technical writing process at Saab, with the focus on the ASD-STE100 standard. Furthermore, we will introduce Large Language Models (LLMs) and instruction fine-tuning. Then, we will briefly discuss human expertise in the fine-tuning processes of LLMs. We will also present a framework for the use of Generative AI in technical writing. This chapter will be concluded with an introduction of the study of Bengtsson and Habbe (2024), the authors of the previous thesis work at Saab.

2.1 Technical Writing at Saab

The main task of the technical writers (writing for Gripen C and D) at Saab Aeronautics is to convert various source documents into their final publication form, in terms of content, language, and structure. This thesis focuses on a specific task of the technical writers, the creation of Aircraft Maintenance Publications (AMP). During this process, the technical writers receive source documents from maintenance engineers and transform those into new publications, which then get distributed to Saab's customers. Outside this specific task category, it is important to note that format of the source documents the writers receive varies, and at times the technical writers are ordered to only add some additional information to existing publications or edit sub parts of them.

The technical writers need to also comply with both company and project-specific guidelines and writing requirements. However, due to the confidential nature of these documents, we will be introducing only Simplified Technical English (ASD-STE100).

2.1.1 ASD-STE100 Simplified Technical English

Simplified Technical English, STE, is a controlled language used in the writing of various technical documents, created in the late 70s by the Aerospace, Security and Defense Industries Association of Europe (ASD) (Aerospace and Defence Industries Association of Europe, 2025). It was originally created and intended specifically for the aerospace industry. The main motivation behind developing STE is to improve safety by simplifying the language used in various aircraft maintenance documents, given that a majority of the professionals who read and work with these technical documents do not speak English as their native language. In essence, STE guidelines aim to prevent any sort of ambiguity or confusion. The first AECMA Simplified English Guide was published in 1986. Today STE is maintained by STEMG, ASD Simplified Technical English Maintenance Group and is widely adopted as a language specification also in other industries.

The ASD-STE100 document is divided into two main parts, the first one specifying various writing rules. These rules concern both the grammar and style of writing. Table 1 illustrates some of these writing rules. The reader is asked to note that each section contains several rules, but we have picked only one for illustrative purposes.

Table 1. Example Rules from ASD-STE100, issue 9

Section	**Example rule from the section**	**Example of *not* approved usage**	**Example of approved usage**
Section 1 – Words	Rule 1.2 – Use approved words from the dictionary only as the specified part of speech.	Dim the lights. ("Dim" is approved only as an adjective.)	Set the lights to the dim position.
Section 2 – Multi-word nouns	Rule 2.1 – Write multi-word nouns of no more than three words.	Install the forward turbine overheat thermocouple terminal tags.	Install the terminal tags on the forward overheat thermocouple of the turbine.
Section 3 – Verbs	Rule 3.4 – Do not use auxiliary verbs to make complex verb constructions.	The temperature must be adjusted.	Adjust the temperature.
Section 4 – Sentences	Rule 4.3 – Use a vertical list for complex text.	The wheel assembly comprises the tire, the tube, the spokes, the spoke fittings, the valve, and the hub.	The wheel assembly has these parts: -The tire -The tube -The spokes -The spoke fittings -The valve -The hub.
Section 5 – Procedural writing	Rule 5.2 – Write only one instruction in each sentence unless two or more actions occur at the same time.	Set the TEST switch to the middle position and release the SHORT-CIRCUIT TEST switch.	A. Set the TEST switch to the middle position. B. Release the SHORT-CIRCUIT TEST switch.
Section 6 – Descriptive writing	Rule 6.3 – Write short sentences. Use a maximum of 25 words in each sentence.	-	-
Section 7 – Safety instructions	Rule 7.1 – Use an applicable word (for example, "warning" or "caution") to identify the level of risk.	CAUTION: EXTREME CLEANLINESS OF OXYGEN TUBES IS IMPERATIVE. (The word "caution" is to be used when there is a risk of damage only to machines, tools, or equipment).	WARNING: MAKE SURE THAT THE OXYGEN TUBES ARE FULLY CLEAN. OXYGEN AND GREASE MAKE AN EXPLOSIVE MIXTURE. AN EXPLOSION CAN CAUSE INJURY OR DEATH.
Section 8 – Punctuation and word count	Rule 8.1 – You can use all standard English punctuation marks but not the semicolon (;).	The battery is not user-replaceable; it can only be replaced by an approved service station.	Users cannot replace the battery. Only specialists at approved service stations can replace it.
Section 9 – Writing practices	Rule 9.4 – When you select terminology or wording, always use a consistent style.	Lubricate the two bolts (10) with oil. Torque-tighten the two bolts to 4 Nm. […] Apply a small quantity of oil to the threads of the four bolts. Torque each of the four bolts (12) to 0.8 Nm.	Apply a small quantity of oil to the threads of the two bolts (10). Torque each of the bolts to 4 Nm. […] Apply a small quantity of oil to the threads of the four bolts. Torque each of the four bolts (12) to 0.8 Nm.

The second main part of STE guidelines consists of the dictionary. It defines the set of approved and not approved words. The general rule is that each word may only have one meaning and belong to a single word class. Table 2 illustrates examples from the STE dictionary.

Table 2. Examples from the STE Dictionary

STE-Approved word and POS	**STE-approved sentence**	**Not STE-approved word and its POS**	**Not STE-approved sentence**	**Reason**
Connect (v)	Connect the towing arm to the nose gear.	Hook up (v)	Hook the towing arm up to the nose gear.	"Connect" is a simpler word option to "hook up".
Do (v)	If the test does not give satisfactory results, do the fault isolation procedure.	Implement (v)	If the test does not give satisfactory results, implement the fault isolation procedure.	"Do" is a simpler word option to "implement".
Make sure (v)	Make sure that it is safe to supply electrical power.	Check (v)	Check that it is safe to supply electrical power.	"Check" is marked as a noun in the dictionary, thus usage of it as verb is not allowed.

We can see from Table 2 that whenever there are several verbs with similar meanings, the user is told to choose the simpler option. Additionally, as STE highlights that each word usually belongs to a single part of speech, it is important for the user to verify the correct use. For instance, if the user wants to use the word "check" with a similar meaning as "make sure", the sentence will have to be formatted so that "check" is used as a noun ("Do a check of…").

2.2 Large Language Models

Large Language Models (LLMs) are deep neural networks that have been pre-trained in a self-supervised manner with massive quantities of text data scraped off the internet. For instance, in the training methodology of previous-generation **GPT-3** (Brown at al., 2020), a filtered version of the Common Crawl[1] dataset was used as a part of the complete training data. This Common Crawl subset alone consisted of around 410 billion tokens. Many of today's LLMs are multimodal, meaning they have also been trained with images.

LLMs are most often pre-trained to *model* language, in other words to predict the next token given a history (Qiu et al., 2020). In their essence, language models thus calculate probability distributions over a vocabulary. Having not only revolutionized the whole field of Natural Language Processing (NLP), LLMs are also transforming the ways we search for information, understand and produce content, and optimize our work. Their striking abilities in assisting humans in a multitude of tasks only keep developing as new methods are continuously being innovated to improve the performance of these models.

[1] The Common Crawl is a vast open-source corpus containing raw text from webpages, as well as metadata and text extracts (Common Crawl, 2025).

The transformer architecture and its ground-breaking self-attention mechanism, originally presented in the landmark paper by Vaswani et al. (2017), served as the foundation for several different types of modern LLMs. The transformer architecture allows the model to create contextual token embeddings by attending to all of the surrounding tokens, leading to a vast contextual understanding between the input and the output. The multi-head attention mechanism allows this process to run in parallel, with each attention cell's different set of weights capturing different types of dependencies, structures, and relations between the words. Furthermore, cross-attention allows the model to keep track of relations between the input and the output.

In decoder-only models, the encoder block (see left side of Figure 1) is removed. Since the input and output sequences are no longer fed separately, the self-attention differs in that the current token can only attend to itself and the previous tokens, known as causal or masked self-attention. This means that the encoding and decoding processes are combined in a single unit, as the cross-attention between the outputs from the encoder and the decoder is not used (see arrows connecting encoder and decoder blocks in Figure 1).

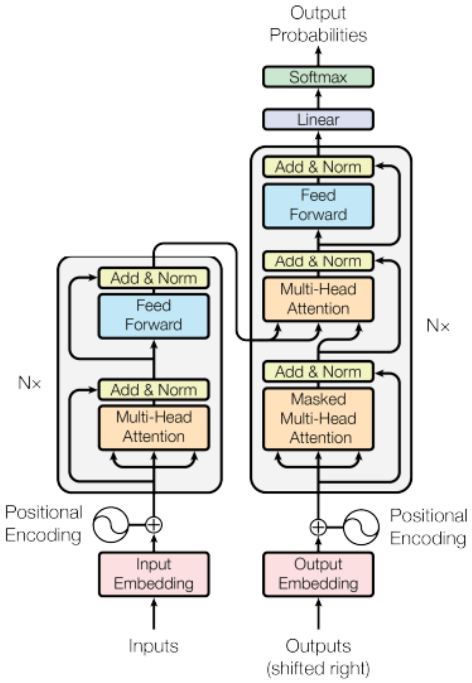


Figure 1. The Encoder-Decoder Transformer Architecture from Vaswani et al. (2017, p.3). Decoder-only models rely on purely the decoder block, represented on the right side of the figure.

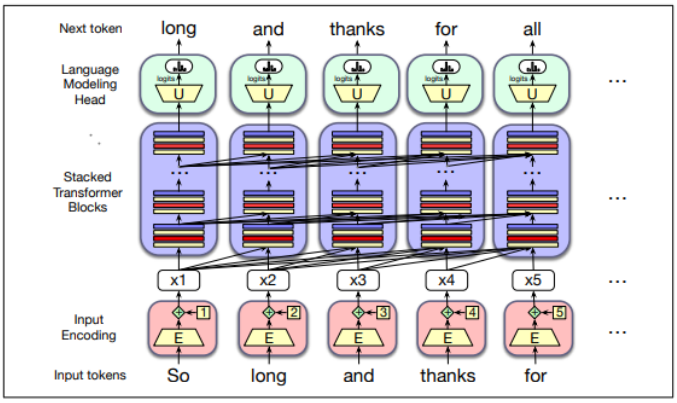


Figure 2. The Architecture of a Causal Transformer Model (Decoder-only) from Jurafsky and Martin (2025, Figure 9.1).

Figure 2 illustrates a causal transformer architecture, with stacked transformer blocks, allowing the capturing of various types of relations between the input tokens.

In essence, decoder-only LLMs perform conditional generation. The prompt that is fed to the model works as the starting point for the model when it starts the generation of the output. Then, the model continues producing tokens based on both all the tokens it has already generated and the original prompt (Jurafsky and Martin, 2025). This decoding process can be done in various ways, which are referred to as sampling algorithms.

2.3 Instruction Fine-tuning LLMs

Fine-tuning Large Language Models refers to the further training process in which the pre-trained LLM is being fed domain-specific data in order to better align its performance with a down-stream task. In other words, the weights of the pre-trained model are updated with a task-specific supervised dataset (Brown et al., 2020). As the training process of an LLM from scratch is often both computationally expensive as well as time-consuming, fine-tuning has emerged as an efficient method of adjusting the LLMs to various down-stream tasks.

As already stated above, LLMs have been pre-trained with massive amounts of text and their ultimate task is to predict the next tokens given the previous ones. However, when we want the LLM to provide a helpful response to our instruction or command, these base LLMs might not be the best option, as the objective of the LLM is not directly to follow the user's instruction (Ouyang et al., 2022).

Instruction-tuned models are LLMs that have been specifically fine-tuned to understand human instructions, with the goal of better executing various tasks (Sun et al., 2023). These tasks can be for instance question answering or content summarization. The fine-tuning process happens by training the model in a supervised manner with instruction prompt-response pairs, illustrating the desired behavior of the model. Using cross-entropy loss, the goal is for the model to learn to predict the desired output from the instruction prompt (Jurafsky and Martin, 2025).

When it comes to the amount of data needed for LLM fine-tuning, although data quantity required is considerably less than during the pre-training phase, the performance of the fine-tuned model usually increases along with the size of the dataset. Fine-tuning with too little data can lead to over-fitting, where the model does not learn to generalize to unseen data, but instead relies on simply memorizing the training data. In order to counteract overfitting, hyperparameter tuning and regularization techniques can be explored, as suggested in Vieira et al. (2024).

There exist, however, studies where fine-tuning with very little data has yielded good results. For instance, Zhou et al. (2023) achieved remarkable results when fine-tuning a LlaMa (65B) language model with only 1000 prompt and response pairs. During human evaluation, the outputs from their

model were rated as equivalent or better than those of GPT-4 in 43% of the cases. The corresponding rates for comparisons against Bard and DaVinci003 were 58% and 65%, respectively. In addition to highlighting the power of pre-training, this shows that also data quality plays a big role in the fine-tuning process. Additionally, with main motivation on cost-reduction, Chen et al. (2023) experimented with Low Training Data Instruction Tuning (LTD) in task-specific scenarios. Having focused on Natural Language Inference (NLI) in the testing phase, the authors noted that in cases where the task is pre-defined and restricted, fine-tuning the model with as little as 0,5% of the original data can yield satisfactory results.

Parameter-efficient fine-tuning, a popular and cost-effective method for fine-tuning, will be introduced in Section 3.3.

2.4 Expert Post-editing of LLM Outputs

Human or expert intervention, such as performing corrections, adding details, or other ways of refining LLM outputs is still often needed, especially in domain-specific contexts.

Occhipinti et al. (2024) investigated the impact of human post-editing of LLM-generations on the fine-tuning process of decoder-only LLMs (Pythia in 1.4B, 6.9B, and 12B). The authors found that when trained with human-edited data, the average accuracy at N^2 of the LLMs increased with the 12B model, and the quality of the outputs was rated higher on both naturalness and comprehensibility by human evaluators. Interestingly, the authors noted that the number of human edits performed on the LLM-generated outputs correlated positively with the higher ranking on these aspects. Lastly, the authors observed that the impact of the human post-editing is amplified with smaller LLMs.

Human demonstration data collection is also the first step of the reinforcement learning from human feedback, as presented in Ouyang et al. (2022). During this stage of this method, a labeler illustrates the desired performance of the model by generating an output to a sampled prompt. That data is then used to train an LLM in a supervised manner. Subsequently, the fine-tuned model is used to generate different outputs, which then get ranked by the labelers. This step generates the preference dataset, which is then used at the last stage in order to update the policy of the fine-tuned model. This can be formatted in a continuous loop, where the model's performance gets improved over use.

Achieving accurate and reliable datasets through human annotation comes often at the cost of increased time and resource use. However, in closed domain and precise contexts these demands can be seen as compensated to some extent by the reliability and quality of the data collected.

[2] "[…] prediction is counted as correct if the correct token occurs within the top N most probable words given by the model." (Welch et al., 2022)

2.5 Framework for Generative AI in Technical Writing

The field of technical writing can be seen as a subfield to technical communication, covering multiple different domains including editing, reviewing, and content strategy. However, when discussing the writing process specifically, Getto et al. (2025) suggest their own Generative AI technical writing framework, divided into the following four heuristics:

- **Defining Tasks.** The assessment of tasks within the workflow that are suitable for automation. Acknowledging the workflow tasks in which automation is not the solution.
- **Prompting Strategies**. Providing information regarding different ways of prompting and their suitability and effectiveness. Learning about model-specific prompting.
- **Quality Control Strategies**. Cross-checking the factualness of AI-generated information. Possible subject-matter expert consultation.
- **Scaling and Automation Strategies**. Developing guidelines and framework on a larger scale regarding the tasks in which AI-automation could be adopted, after having collected feedback and insights from AI-automation experiments.

The authors also discuss the Human-in-the-loop-approach when adopting generative AI into the writing process, emphasizing the importance of continuous human judgement while recognizing the utility of generative AI in some tasks. This view highlights the use of AI as an **enhancing tool**, not as an author itself, which is the view we also share in this study.

2.6 Previous Thesis Work

Bengtsson and Habbe (2024) launched the exploration of LLM's potential assistance in the process of technical writing at Saab. In their thesis, they explored fine-tuning an instruct-tuned model (Mistral-7b-instruct-v.02) with two different training methodologies, Supervised instruction fine-tuning (SIFT) and direct preference optimization (DPO).

Their data was composed of 380 corresponding source and target technical documents extracted from Saab's internal software. The source documents consisted of the files that the technical writers receive (compiled by the maintenance engineers) and the target documents consisting of the final technical publications prepared by them. As the final technical publications follow the S1000D[3] standard, the content of the target documents was stripped from their XML-format into pure text by the authors.

As for the data preprocessing, the authors faced two issues in preparing the data for the training. Firstly, training the model with the complete document pairs (consisting of concatenated subtasks, which usually consist of one or a few sentences) was not possible due to the memory limitations of the GPU. Consequently, when they investigated the possibility of training the model with those shorter subtasks, new difficulties arose regarding the aligning of these *sequences* between the source and target data. On average, a source document consisted of 54 sequences and a target document of 65 sequences, meaning they didn't have one-to-have correspondence. The authors attempted aligning the sequences by calculating the cosine similarity of the words' BERT embeddings, but this approach resulted in a very poor outcome. This led to the authors deciding to generate synthetic source data from the target sequences. For this, they designed the few-shot prompt illustrated in Figure 3.

[3] XML-based standard for the production of technical publications (S1000D, 2025).

You are a writer, and your expertise is to rewrite sentences so that the sentences do not adhere to Simplified Technical English.

You rewrite and paraphrase the sentence within <<< >>>.

You keep references and numbers in their original form.

You only provide the augmented sentence.

You do not explain your changes.

You do not provide alternatives.

Example 1:
<<< *Example input* >>>
Response: *Response*

Example 2:
<<< Example input >>>
Response: *Response*

Example 3:
<<< *Example input* >>>
Response: *Response*

Example 4:
<<< *Example input* >>>
Response: *Response*

The sentence you will change:
<<< *input* >>>

Figure 3. The prompt used in Bengtsson & Habbe (2024) to generate the synthetic source sequences.

The generated synthetic source sentence was compared against the real source data in terms of sentence length, number of sentences as well as semantic similarity (BERT-score). The synthetic data contained more tokens and sentences than the real source data. Furthermore, The BERT F1-score between the synthetic source data and the target data was higher (0.828) than the score between the synthetic source data and the real source data (0.6579). These results are not particularly surprising, since the synthetic source data was generated from the target data.

All in all, their data preprocessing resulted in 25,000 synthetic source – target sequence pairs (extracted from those 380 documents) that were used in the training of both of the models (SIFT and DPO). Both of the models were trained with the prompt presented in Figure 4.

Your task is to translate a sentence into Simplified Technical English (STE), a controlled language designed to ensure clarity in technical documentation. It is commonly used in aerospace and other technical fields where safety and clarity are paramount. The sentence you will translate is at the end of this prompt.

Please adhere to the following guidelines in your translation:

Active Voice: Use active voice so that the subject performs the action stated by the verb, which simplifies understanding.

Short Sentences: Break down complex ideas into shorter sentences. Ensure each sentence contains only one main idea for easier comprehension. One sentence may be a maximum of 20 words long.

Noun Clusters: Noun clusters should be no more than three nouns.

Avoid Technical Jargon: When possible, use simple and general language. If you must use technical terms, provide clear definitions.

Single Solution: Provide only one translation that best adheres to the principles of STE.

Verbs: Verbs should be limited to their infinitive, imperative, simple present tense, simple past tense, past participle, and future tense forms. Do not use the "-ing" verb form, unless it is a technical term.

Maintain Documentation References: Keep any references to other documentation intact in the translation to preserve the context and utility.

Original Sentence for Translation:
Input Sentence

Directives for Output:

Translate the sentences into Simplified Technical English according to the rules specified above.

Provide only the translated sentences, with no additional notes, any notes regarding your changes, explanations, or commentary.

Figure 4. The prompt used during the training of the models in Bengtsson & Habbe (2024).

We can observe from the prompt in Figure 4 that the authors had explicitly specified several STE-rules, such as writing in active voice, producing short sentences and only specific verb forms.

The results were evaluated against multiple quantitative metrics, measuring among other things sentence length, semantic similarity (BERT), simplification (SARI), and number of edit operations. The authors referred to these as the technical writing evaluation. Additionally, the performance of the models was also evaluated in terms of how well they followed company-specific guidelines. This covered aspects such as compliance with STE-rules, clarity of the text, inclusiveness, and terminology. For this, they used Acrolinx, a supporting tool that the technical writers use in their work.

In terms of the technical writing evaluation, when testing the model with real source data the authors found that the SIFT model outperformed the DPO model in semantic similarity (BERT F1 0.7413 vs. 0.6571), text simplification (SARI 46.9310 vs. 35.8986) and number of edit operations (TER 83.7301 vs. 102.0834). Regarding the compliance with the company-specific rules, the SIFT model outperformed the DPO model in STE, consistency, and terminology. DPO outperformed SIFT in clarity and correctness, and they obtained identical scores in inclusiveness and tone.

Interestingly, **despite the SIFT model's better performance on almost all of the metrics, a visual analysis proved that the outputs contain frequent model hallucinations, repetitions, and other unwanted behavior**. The DPO model, however, seemed to produce more stable, coherent, and organized outputs. The same applied even to the base model, which performance positively surprised the authors. The authors witnessed that the performances of the models deteriorated when they tested them with real source data, which is most likely due to the differences between the synthetic and real source data that we discussed above. Furthermore, as approximately 70% of the source data was in Swedish, for the testing phase they translated it in English using MADLAD-400-3B-MT translation model, which was loaded into the development environment at Saab. It is likely that having used this machine translated data during the testing had a further impact on the results.

A deeper qualitative evaluation of the models' outputs in the work of Bengtsson and Habbe (2024) would have been interesting as it would have provided more insight into the models' performance and their differences. The authors did do some visual inspections - both in the preprocessing and evaluation phase - but those covered only a very small number of the documents.

It is also important to note the practical limitations of their work – aspects that are mainly tied to the confidentiality issues and security precautions of Saab. These limitations are also present in our own work. First of all, the choice of model was tied to a very restricted array of options, as the model needs to be available in the development environment of Saab. This is most likely also a reason for the authors not being able to explore alignment models for the source and target data matching. Secondly, the confidential nature of the data meant that no real examples were shown in the thesis. Thirdly, the data represented only remove and install procedures of JAS 39 Gripen. Lastly, the authors only had access to a single GPU (NVIDIA A100, with 80GB of VRAM) that was shared between multiple Saab employees, adding more challenge in planning the training methodology within these specific memory limitations.

3 Methodology

We will start this section by presenting the data used in our study. We will first discuss the creation of the parallel corpus in Section 3.1. After, in Section 3.1.4, we will present the demonstration data collection performed together with the experts. Subsequently, we will discuss the model fine-tuning settings starting from Section 3.2 and evaluation methodology in Section 3.5. We will end this section with a brief summary of the main methodological differences between our study and that of Bengtsson and Habbe (2024).

3.1 Data

The data was provided to us by Saab, consisting of maintenance documentation for the JAS 39 Gripen, specifically 380 source and target document pairs. The data consists of purely remove and install procedures of the C/D model, and the source-target document-level alignment was performed by Bengtsson and Habbe (2024).

The source data are written by maintenance engineers, and it illustrates a possible subset of the document types that the technical writers receive. These are the documents that they will transform, following the STE and Saab-specific guidelines and rules, into the final aircraft publications. The source documents had been compiled by concatenating the text describing the subtasks (referred to as *sequences* in Bengtsson and Habbe (2024)) extracted from Saab's internal software.

The target data, written by the technical writers, had been extracted by Bengtsson and Habbe (2024). Since the final Aircraft Publications are in XML-format, the authors decided to strip the text from inside the tags and only use that in their work. This stripped data is what we are also using in our study.

As we mentioned earlier, one of the biggest challenges of Bengtsson & Habbe (2024) was the data alignment between the inputs and outputs. The authors managed to align the data on file-level, but due to memory limitations of the GPU, training the model on these complete documents was not possible. Additionally, matching of the sequences (aka different subtasks, most often single sentences of the install and remove procedures) proved to be challenging. One of the most notable challenges in aligning the source and target sequences was that the a source sequence was often mapped to several target sequences.

3.1.1 Creation of the Parallel Corpus

After having investigated the source and target data files carefully, we decided to explore if aligning the contents of the source and target documents was possible on a level other than the sole sequences. We found out that, despite considerable differences in the way the source documents were structured, certain headings were present in the source as well as the target documents. The body text under these headings consisted of several subtasks and aligning them between the source and target documents seemed much more doable than simply aligning the single subtasks.

The text sections (also referred to as **procedures**) that we divided the documents into were the following:

- **To prepare the aircraft / Förberedelser**,
- **Remove procedure / Borttagning**,
- **Install procedure / Montering**,
- **To do a check after the installation / Kontroll efter montering,**
- **To restore the aircraft / Återställning**.

3.1.1.1 Observations of the Source Data

A qualitative analysis into the source and target data allowed us to gain insight into their structure and characteristics. As the maintenance engineers producing the source documents are not required to follow any strict writing guidelines themselves, we believe this investigation was necessary to create a high-quality parallel dataset that would also be used in the fine-tuning process.

Since we had observed a certain combination of characters and capitalized words often marking these headings, we started the process by extracting these headings and the text under them from the source files with the help of regular expressions. However, due to frequent inconsistencies in the formatting, it was necessary to perform manual checking and corrections on the resulting data structure (heading and corresponding text content from procedures formatted on lines). At times, in order to categorize the text under the correct procedure heading, we consulted the target data documents. This manual work allowed us to drastically improve the quality of the resulting data structure.

The most notable inconsistencies in the source documents were the following:

- Problems with headings: missing headings, differences in the names of the headings, differences in the hierarchy of the headings, and sometimes two headings were combined into one.
- Differences in the organization of the documents concerning aircraft model applicability: most of the source documents were organized by first stating the name of the procedure, for instance “REMOVE” or “INSTALL” and only then listing the applicability (Gripen C or D). However, a considerable number of the documents had the inverse order, where the procedure titles were listed twice in the document, different for Gripen C and D. In the data structure we created, we listed applicability only under the text content of the procedure, meaning that each document only contains one REMOVE and INSTALL procedure.

Additionally, we gained insight into the language aspect of the source documents. A vast majority of the source documents were written solely in Swedish, as noted also by Bengtsson and Habbe (2024). Approximately 25% of the documents were written in pure English, and the rest was a mix of both languages. As for the documents where the content was in both languages, it could be that the headings were written in one language and the content in another, some sentences could have been written in Swedish and the next in English, or the names of some tools or materials/supplies might have been written in Swedish although the rest of the document was written in English. We performed a manual annotation of the document language as well.

Bengtsson and Habbe (2024) decided to machine translate the source documents that were written in Swedish. This data was used in the testing phase, since they conducted the testing with complete source documents (training was performed with synthetic sequences). In our quest to decide which input data to include in our work, we also conducted our own visual assessment of the machine translated data. Only a brief analysis showed that the translations leave a lot of room for improvement. In addition to simple mistakes in the vocabulary, such as ***fellokalisering*** (error localization/troubleshooting) having been translated as ***mislocation***, one of the most prominent issues was inconsistencies in the word choices. For instance, an integral aircraft component, ***lucka***, which translates to ***access panels*** in Saab-context, was translated very inconsistently as doors, covers, gaps, lids, or even holes.

Furthermore, the translations were unable to translate the numerous Saab-acronyms correctly into English, which causes a lot of meaning to get lost. Letters indicating important aspects such as side of the item (right or left side) also are not correctly translated. Lastly, we were able to identify at least one instance where the original English source document had mistakenly been translated into English, leading to some changes in its content.

As we want to emphasize high-quality real-world data and we are dealing with technical documents where precision and attention to detail is very important, we decided to only include source documents written in their original language.

3.1.1.2 Observations of the Target Data

As for matching these source document procedures with their respective targets, we were able to rather quickly perform the alignment manually thanks to the detailed analysis into the structure of the documents discussed earlier. The target documents contained mainly the same main headings as the source documents, mainly following the ***Prepare-Remove-Install-Check-Restore-***structure. We feel that a manual inspection and work was necessary also with the target data due to the following issues that we encountered:

- With certain documents, the use of headings in the target documents was not aligned with those of the source documents. Some of the target documents contained sub or sub-subheadings that were not present in the source documents. This required careful alignment of the corresponding text.
- As with the source documents, some of the target documents followed the inverse applicability-heading order mentioned above in 3.1.1.1. It was necessary to manually confirm that the structure of the target document will match that of the source.
- Due to the aircraft model applicability being marked with the XML-tags in the target documents (which were not visible in the extracted text versions we used), correctly placing the content under the headings sometimes required consulting the XML-files.

3.1.2 Finalized Parallel Corpus

At the end of the alignment process, we performed a visual inspection of the source-target procedure pairs. With some of these pairs, either with the source or with the target data, the only content under a heading was a reference to another document or to another task. For example, the content under the source document's heading "To do a check after the installation" could have been "Refer to document XX". With these types of cases, we excluded the instance from the dataset.

In total, our parallel corpus consists of 502 English-STE procedure pairs and 302 Swedish-STE procedure pairs. As we use an already instruction-tuned model[4] in our study, we then planned the instruction prompts that the source data from the parallel corpus would be combined with.

The English source data was randomly matched with followings prompts:

- *"Translate this text into STE."*
- *"Rewrite this text using Simplified Technical English."*
- *"Transform this text into STE."*
- *"Transform this text into Simplified Technical English (STE)."*
- *"Here is text transform it into Simplified Technical English (STE)."*

[4] Model discussed in chapter 3.2.

The Swedish source data was randomly matched with similar prompts, only with an additional specification concerning the source language:

- *"Transform this Swedish text into Simplified Technical English (STE)."*
- *"Rewrite this Swedish text using Simplified Technical English."*
- *"Transform this Swedish text into STE."*
- *"Translate this Swedish text into STE."*
- *"Here is a text in Swedish transform it into Simplified Technical English (STE)."*

Table 3 offers a fictional example of a training instance.

Table 3. Fictional training instance example.

Instruction prompt	**Input text**[5] **(non-STE)**	**Output (STE)**
Transform this text into Simplified Technical English.	Install procedure. Set a new O-ring on the O-ring guide, drive the O-ring guide onto the shaft and then place the ring gear into the gear case cover.	**Install** a new O-ring on the O-ring guide. **Put** the O-ring guide onto the ring gear shaft. **Install** the ring gear into the gear case cover.

As illustrated in Table 3, the instruction prompt was combined with the input text to form the input to the model. We can observe that the text content in the Output (STE)-column showcases wanted behavior by following the STE writing guidelines: it has only one procedural work step per sentence, and the verb forms and their meanings are accepted in the STE-dictionary.

When not accompanied with detailed prompting, we are interested in seeing up to which level the model is able to implicitly learn what Simplified Technical English refers to, the changes it is expected to make, and the vocabulary it is supposed to adopt. This differs from the approach of Bengtsson and Habbe (2024), who trained the model with a very comprehensive prompt and explicitly fed the model the whole STE-dictionary at inference.

3.1.3 Meetings with the Technical Writers

Before starting the demonstration data (the expert-annotated data) collection, we had an initial meeting with three technical writers at Saab Aeronautics. The goal of this first meeting was to learn more about the general job description of the technical writers, become more familiar with the tasks and tools they use, as well as to collect insights concerning their views on AI and how it could facilitate their tasks. The meeting allowed us to also learn about the technical writing process — the steps that take place from receiving a source document from the maintenance engineers, all the way up to releasing a final Aircraft Maintenance Publication.

The technical writers also introduced the tools that they use in their work, one of them being Acrolinx, an automatic content planning and evaluation software (Acrolinx, n.d.), which aids the technical writers in complying with both the STE and company-specific guidelines. Acrolinx is able to alert the writers in case the text they are working on contains not-STE approved words, word forms, or other project-specific deviations. It also scores the written text on specified metrics (ASD-STE100, clarity, consistency, correctness, inclusive language, terminology, and tone).

[5] Edited from Honda Shop Manual CB550SC/CB650SC Nighthawk.

The main takeaway from our first meeting with the writers was learning that often the most time-consuming part of their tasks is understanding the source documents written by the maintenance engineers. These source documents frequently contain ambiguous or unclear sentences. Furthermore, even though the technical writers are not required to have a high level of technical know-how, the writers we interviewed expressed that having a general understanding of the content of the source documents helps considerably in the writing process. In other words, what they would look for in a chatbot assistant is an ability to capture the context and meaning of the source documents. Another aspect was that the source documents can be written in many different ways, structured in different ways, as the maintenance engineers are not strictly required to follow any guidelines when writing the technical source documents. The writers explained that a chatbot from whom they could ask summarizations, explanations, clarifications of the source documents would be very helpful in their work and speed up the decoding process. However, it was also mentioned that it could be useful to have the chatbot work as "rough" STE-translator in the writing pipeline – its outputs could then be refined and edited to achieve high quality end results.

Even though the writers did not express that following the STE-guidelines or the *writing* process itself as the most time-consuming aspect of their work, we were still curious about this aspect, too. When specifically asked about the STE-aspect of the writing process, the writers mentioned that the most common changes they need to perform on the source documents have to do with breaking down long sequences into shorter ones, transforming passive voice into active, and finding suitable words to replace not-approved words. It was mentioned that they are sometimes obliged to deviate from the list of approved words listed in the STE-dictionary, and thus having a supplementary tool alongside Acrolinx to suggest other words could be useful.

During the second meeting with the three technical writers, we were able to present both preliminary instruction prompts that we had designed based on the insights from the first meeting. We also showcased the performance of Mistral-7b-instruct-v.02 before any fine-tuning with some randomly selected inputs from the source dataset. Although we were not able to analyze these outputs in a very detailed manner together with the technical writers, they were mostly positively surprised by the outputs, describing their format of the outputs as something that they could be looking for in the final model. Table 4 presents some examples of the prompts that were introduced to the writers during the second meeting. Note that the examples are not from the real dataset used in this study.

Table 4. Examples of the Prompts Designed Based on the Meeting with the Technical Writers

Instruction prompt	**Input**[6]	**Output**
What does "high flash" mean in the context of this text?	Clean all parts with non-flammable or high flash point solvent.	In this context, high flash…
Summarize this text. What are the main sections of this text? What is this text about?	Assemble the crankcase halves, aligning the shift fork claws with the grooves in M2, M3, C5, and C6 gears. Also align the alternator chain slider and cam chain slider with the grooves in the lower case. Place the special assembly pins, bolt holes first through the very front outer holes of the upper case. Thread 6 x 20 mm bolts into the pins from the lower case side. Torque the assembly pins/bolts to 10-14 N-m (1.0-1.4 kg-m, 7-10 ft-lb). This will precisely align the crankcase halves with each other.	Here is a summary of… Here are the main sections… This text is about…
What tool can be used to tighten the lock nut?	Place the bearing holder in a vise with soft jaws and, apply engine oil to the bearing holder and lock nut threads. Tighten a new outer race lock nut to the specified torque value.	In this context, the tool that will be used is most likely…

Additionally, we also conducted an interview with a Technical Information Engineer working with publication and STE quality standards at Saab. Our goal was to complement the information we had received from the technical writers with the views of a professional who has a deep understanding of both STE and the final AMPs that Saab produces. The main point we learnt from this meeting was that according to them, in order to produce high-quality technical documentation, it is important that the technical writers understand the content of the input documents. The writers receive only general training on Gripen, but no training on how technicians work on it. Additionally, they emphasized that the quality of the source documents can have an impact on the quality of the technical publications. "Rewriting text that one doesn't understand can be difficult", they summarized. These notes seem to go hand in hand with the insights we learnt from the meetings with the technical writers.

[6] Extracted from Honda Shop Manual CB550SC/CB650SC Nighthawk.

3.1.4 Demonstration Data Collection with the Technical Writers

After having met the technical writers a few times and having designed the prompts based on the insights from those discussions, we were able to start organizing the data collection in more detail. The data to be used in the demonstration data collection was divided into three categories, mainly based on the length of the inputs and the goal of the data:

1. **Instruction prompt + complete source document + [expert output]**
   This data is used to teach the language model to summarize, reformulate, and help the user to understand the Gripen-related source documents. Each input is a complete source document.

2. **Instruction prompt + source text procedure + [expert output]**
   This data is used to teach the language model to summarize, reformulate, and help the user to understand the Gripen-related source documents. Each input text is a procedure from the source documents.

3. **Instruction prompt + complete source document / source text procedure / source text sentence(s) + [expert output]**
   This data is used to teach the language model to answer questions related to the content of Gripen source documents, answer STE-related questions, suggest fixes to STE-issues, and answer technical questions related to tools, for instance. The input text can be either a complete source document, a procedure, a single or a few sentences.

As for the prompts that we designed, we kept in mind that varying the phrasing can improve the performance during inference phase (Jurafsky and Martin, 2025), as it may allow the model to generalize better. If we follow the case categories of different prompts from Ouyang et al. (2022), our prompts mainly fell under open QA, rewrite, summarize, and extract.

After having presented the annotation task to the writers, considering the time and work hour limitations, we decided to first generate outputs with the base model (Mistral-7b-instruct-v.02) in a zero-shot manner. The writers were then instructed to review the base model's outputs, add or remove information, edit the content or the structure, or even completely rewrite the output. The writers were explicitly instructed that **the output they produce should reflect their technical writing expertise, encouraging them to add domain-specific information.** Additionally, the writers were told that modifying the prompts was also allowed, as long as they made sure the resulting prompt + input text combination matched the output. Lastly, the writers were also able to leave comments under the instances regarding the annotation process. It is important to note that using the base model's outputs as a "starting point" can naturally also introduce bias in the outputs but given the number of instances to annotate and our time constraints, we were obliged to find a way to make the task less overwhelming.

The data was divided into three and shared to the writers as Excel files. The Excel files consisted of the following columns: prompt, input, technical writer output, base model output, and file name. As mentioned, an additional column was added for the writers to leave their optional comments in. We instructed the writers about the annotation process during another meeting and also provided them with written details about the data and about the annotation process.

Furthermore, we also requested the technical writers to provide qualitative feedback on three of the machine-translated source documents from the work of Bengtsson and Habbe (2024). Like mentioned in previous sections, the authors decided to machine translate most of the source data from Swedish to English with another LLM, and this machine translated data was used during the testing phase.

The feedback we received stated the translations to be generally adequate and fluent; however, they did contain translations errors and sometimes did not capture the context, leading to misinterpretations. For instance, one of the writers noticed that in one of the documents, the Swedish word ***avtankning*** (meaning ***de***fueling) had been translated into ***re***fueling. This further confirmed our decision not to use any of the machine-translated data.

3.1.5 Finalized Demonstration Dataset

In total, we were able to collect **113** expert-annotated data instances. 73 instances out of those had mainly to do with summarizing, rephrasing, clarifying, or helping with the understanding of either whole source documents or procedures. Table 5 illustrates some of these types of instances from the dataset.

Table 5. Example Instances from our Dataset Regarding Rewriting, Summarizing, or Explaining Content

Instruction prompt	**Input**	**Output**
Can you summarize this text to me in a clear way?	Complete source document	Expert output
What is this text about?	Install procedure from a source document	Expert output
Clarify this paragraph.	Remove procedure from a source document	Expert output

The remaining 40 instances out of 113 covered different aspects such as STE, specific items, tools, or sentences in the source documents. Table 6 illustrates some of these instances.

Table 6. Example Instances from our Dataset Regarding STE, Technical Details, and Explaining Content

Instruction prompt	**Input**	**Output**
Rewrite this sentence using active voice.	Sentence from a source document	Expert output
What words or phrase could I use to replace [NOT-STE APPROVED TERM] in this context?	Sentence from a source document	Expert output
Format this text to working steps. If suitable, also incorporate sub steps. Write the steps in numerical order. One step should only contain one action, if the actions are not required to be done at the same time in the context.	Complete source document	Expert output
What do you think [SENTENCE] means in this context?	Install procedure from a source document	Expert output

The comments left in the annotation files allowed us to gain insight into the importance of performing the demonstration data collection. Examples of expert comments regarding the **quality of the base model output** are presented in the bullet points below.

- The aircraft model is wrong.
- There is redundant or unnecessary information.
- The meaning given to an acronym is wrong.
- The amount of information provided doesn't correspond to the prompt.

As mentioned, to reduce the writers' workload and considering the time restrictions, the annotators were shown the base model's outputs as a starting point. It is important to note that the three technical writers who participated in the demonstration data collection performed the work alongside their own tasks. In order to produce high-quality, expert-enhanced annotations, the annotation work required at times consulting the corresponding source and target documents from their own databases. With only three annotators we were not able to collect many annotations.

3.1.6 Final Complete Dataset

We combined the parallel corpus and expert demonstration dataset to create the final dataset for the model fine-tuning. In total, this final dataset consisted of 917 instances. Table 7 illustrates the final dataset split.

Table 7. *Data split.*

Train set	80%	733 instances
Validation set	10%	92 instances
Test set	10%	92 instances

3.2 Model Choice

Following the model choice of Bengtsson and Habbe (2024), we used Mistral-7b-instruct-v.02, open-source, Apache 2.0-licenced model developed by the French company Mistral AI. This model is the instruct fine-tuned version of the Mistral-7b model, and it is a decoder-only transformer model. As the model is already instruction-tuned, it is important to note that our work constitutes further tuning on top of that. The model in question was already uploaded into Saab's internal development environment, which was a prerequisite given the classified nature of the data. The architecture of the model is presented in Table 8.

Table 8. Mistral-7b-Instruct-v.02 Model Architecture

Parameter	Value
Embedding dimension	4096
Number of layers	32
Head dimension	128
Hidden dimension	14336
Number of heads	32
Number of key/value heads	8
Window size	4096
Context length	32000
Vocabulary size	32000

3.3 QLoRA and LoRA - Parameter-efficient Fine-tuning

Parameter-efficient fine-tuning methods were developed in order to fight against the memory usage issues that often come with fine-tuning large language models. Fully fine-tuning a large language model means having to save both the gradients and optimizer states of each parameter (Sun et al. 2023), as all the weights of the model are updated.

Hu et al. (2022) proposed one of the most known and most popular methods for parameter-efficient fine-tuning of models, Low-Rank Adaptation (LoRA). During LoRA, the weights of the pre-trained model are kept completely frozen, and no gradient updates are done on them. Instead, dimensionally smaller trainable decomposition matrices are "injected" into the layers of the model. These matrices are also referred to as adapters, which get merged into the base model after the training. Figure 5 presents the LoRA formula.

$$h = W_0 x + \Delta W x = W_0 x + BAx$$

Figure 5. LoRA formula, where W is the original model weights, and B and A are the low-rank matrices.

Quantized LoRA (QLoRA) further optimizes the memory efficiency of Low-Rank Adapters by compressing the frozen weights of the pre-trained LLM with 4-bit quantization (Dettmers et al., 2023). In other words, these weights are transformed into a 4-bit representation. Essentially, by lowering the number of bits, the precision of the values is lowered, which reduces the memory usage. New adapter (LoRA) layers are then inserted into the model and updated with a higher precision during the training phase. This drastically lowers the memory usage during the training. QLoRA uses 4-bit NormalFloat as data type for storage, and 16-bit BrainFloat as the computation data type. This means that the base model weights are *de*quantized when performing the necessary gradient computations. As LoRA adapter weights do not get quantized in the first place, they do not need to be dequantized.

It is important to note that choosing a suitable base model is vital when conducting parameter-efficient fine-tuning. For instance, Sun et al. (2023) obtained better results in their instruction fine-tuning process when fine-tuning an already instruction-tuned model, compared to when they applied LoRA on a base LLM.

Saab's development environment contains at the moment of writing this thesis three GPU accelerators, which are a shared resource across users. There is thus a strong incentive to explore parameter-efficient ways of fine-tuning the model, as it will allow fine-tuning models on a single GPU, with a lower RAM usage. We performed the training on L40S Accelerator with 48GB of memory.

3.4 Hyperparameter Settings and Output Sampling Results

As mentioned, in order to quantize the model, we opted for QLoRA. In practice, this meant using the BitsAndBytes library for Python. Our settings are presented in Table 9.

Table 9. Quantization Settings

Parameter	Value
load_in_4bit	True
bnb_4bit_compute_dtype	Torch.bfloat16
bnb_4bit_use_double_quant	True
bnb_4bit_quant_type	Nf4 (normalFloat4)

As for LoRA, we used the PEFT library by Hugging Face. We explored two different rank and alpha combinations. This is illustrated in Figure 6.

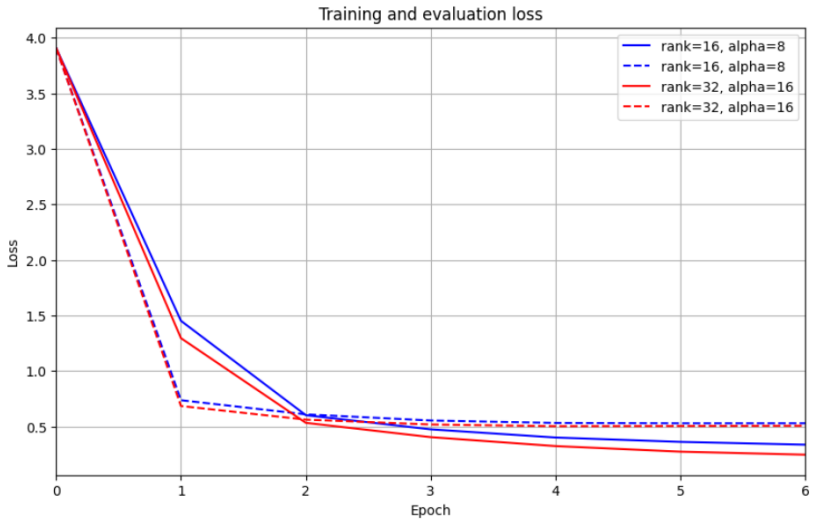


Figure 6. Training and validation losses with two different rank and alpha values. The learning rate was set to 2e-05.

We are able to see from Figure 6 that the explored rank and alpha value combinations did not seem to have an impact on training and validation losses.

In our exploration of suitable values for LoRA, we also had to consider the size of our dataset. Higher rank and alpha values can make the model rely more on the trained adapters, which can easily lead to

overfitting and “overpowering” the general capabilities of the base model. This was something we wanted to avoid.

Additionally, we investigated specifying only the query (“q_proj”) and value (“v_proj”) projection layers (with a learning rate of 2e-05, rank set to 32, and alpha set to 16). We compared the performance of the model by sampling generations to when all the target modules were specified. Although the qualitative differences were not very notable, we did notice the outputs to be slightly more accurate when all the target modules were specified: the English translations given for Swedish source terms were more aligned with the target output, and AMP references were more often kept. The structural formatting of the outputs was similar.

The final settings for the PEFT along with explanations of the parameters are presented in Table 10.

Table 10. PEFTConfig Settings

Parameter	**Value**	**Explanation**
`rank`	16	The dimension of the low-rank matrices. Controls the n of parameters, and thus the “depth” of the features it can learn.
`alpha`	8	Scaling factor for the low-rank matrices. Controls the amount of change that the adaptation layers (low-rank matrices) have on the base model weights. The final scaling is calculated by dividing alpha by rank (resulting in 0.5 in our case).
`lora_dropout`	0.05	Dropout probability for regularization for LoRA weights.
`bias`	none	Whether base model and/or LoRA biases are updated.
`task_type`	CAUSAL_LM	Specification for the task type of the model.
`target_modules`	['k_proj', 'gate_proj', 'v_proj', 'up_proj', 'q_proj', 'o_proj', 'down_proj']	Specification for the linear (attention) layers of the model that will be updated. We specified all.

By lowering the alpha value to half of the rank our objective was to keep the LoRA adapters trained with our small dataset from completely overriding the base model’s behavior.

For other training hyperparameters, we used the Hugging Face Transformers TrainingArguments API, and for the training loop we used the Trainer[7] API. We faced a CUDA out of memory error when we increased the batch size to over 2, which was likely due to the length of our training instances. In order to achieve a higher simulated batch size, we set gradient accumulation steps to 4.

As for the learning rate and number of epochs to run the training for, our plan was to perform a hyperparameter grid search covering different combinations. However, due to the small size of our dataset we first explored if changing these hyperparameters seem to create fluctuations in the training and validation losses. Having tested learning rates 1e-05, 2e-05, and 2e-06, we noticed that the model converged already around epoch 3, the additional epochs bringing only minor decreasing in the training and validation loss. This proved no additional value in training the model for longer. Figure 7 illustrates these findings.

[7] Training API optimized for Transformer models (Hugging Face, n.d.)

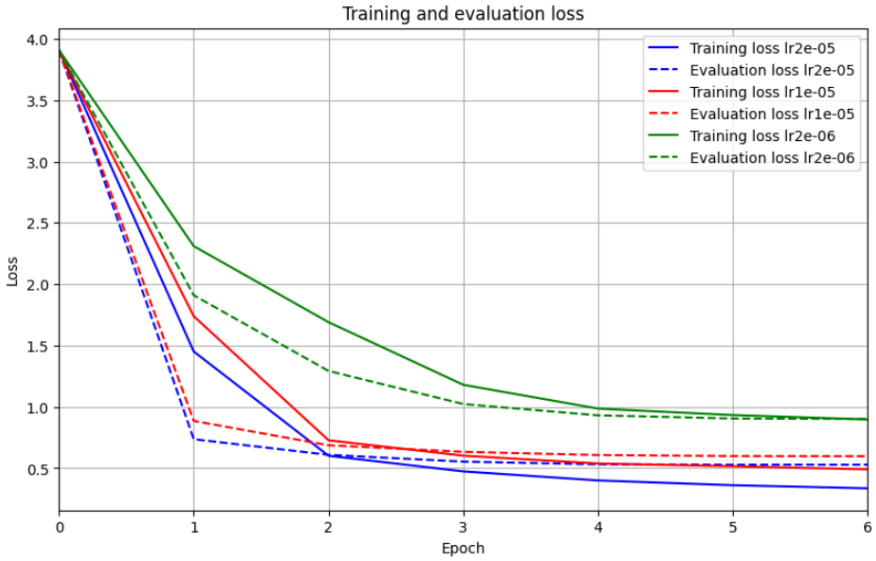


Figure 7. Training and validation losses from our learning rate search. Rank was set to 16 and alpha to 8.

We can see from Figure 7 that the model converges early, around epochs 3-4. The training and validation losses are slightly higher when the model was trained with a learning rate of 2e-06.

We also sampled more generations from the model in order to not rely only on the numeric losses during training and validation. We believe that this was needed, especially due to the small size of our dataset, in order to monitor possible overfitting (which very low losses can be an indicator of). Although our hyperparameter search was not fully exhaustive due to resource limitations, this allowed us to gain valuable information into how those settings actually impact the model's performance.

As for the learning rates, we observed an interesting trade-off in the model performance and the format of the outputs when we tested generating outputs with different models trained with 1e-05, 2e-05, and 2e-06 learning rates. The first two models generated outputs that corresponded to the gold (target) outputs well: the references to other documents were mostly correct, the necessary changes were made in order to comply with STE vocabulary, but the model also managed to refrain from replacing already STE-approved words just for the sake of translating them (we observed this type of behavior often in the base model outputs).

On the other hand, the model trained with a significantly lower learning rate of 2e-06 produced outputs that were structurally more similar to the base model's outputs and thus possibly more pleasant for the user's eyes: sentences were formatted with new lines and more of the content was in a list format. However, the quality of these outputs content-wise was not as good as those generated with the lr 1e-05 and lr 2e-05 models. For instance, references to different tools and items were at times wrong, alongside inconsistent word choices with references to Aircraft Publications and other re-occurring terms. Additionally, the model was at times making changes into more complicated verb forms, for instance changing "start" into "initiate" and "stop" into "terminate".

For the training, we defined our own child class inheriting from the Hugging Face Supervised Fine-Tuning Trainer (SFTTrainer), which itself inherited from the parent class Trainer. The main reason for this was to be able to log the training and validation losses during training, since the default Tensorboard logging was disabled in the development environment we used. For the formatting of the data into the instruction format required by Mistral-7b-Instruct-v.02, tokenizing, and generation of the outputs we were able to partly re-use functions written by Bengtsson and Habbe (2024). However, we did perform necessary modifications on these in order to match them with the format and characteristics of our own data.

Table 11 lists the final hyperparameters used in the training of our model.

Table 11. Hyperparameters Used in Model Fine-tuning

Learning rate	2e-05
N of training epochs	5
Train and validation batch size	2
Gradient checkpointing	True
Gradient accumulation steps	4 (simulated batch size 8)
Bf16	True
Optimizer	paged_adamw_32bit
Warmup ratio	0.1
Lr scheduler type	cosine
Evaluation strategy	epoch

3.5 Evaluation Method

In this section, we will describe the evaluation method adopted in this study. We will start by motivating our choice to rely on human evaluation, specifically performed by experts. We will then describe the evaluation setting. This chapter is concluded with a brief summary of the main methodological differences between our work and Bengtsson and Habbe (2024).

3.5.1 Why Human Evaluation?

The results of Bengtsson and Habbe (2024), which showed major inconsistencies between the quantitative and qualitative analysis. In other words, the quantitative evaluation showed that the SIFT model was outperforming the DPO model, but a visual inspection showed that the authors actually preferred the DPO models outputs. This aligns with Van Der Lee et al. (2019), who claim that automatic metrics do not always correlate with human evaluations. The authors noted that while human evaluations might be more susceptible to subjectivity, they can at the same time be more comprehensive and accurate. In the findings of de Souza et al. (2025), a moderate correlation was established between the human and LLM assessment of LLM outputs, but not between human assessment and deterministic metrics (token overlap, ROUGE-L, and BLEU) adopted in the study. This finding aligns with Szymanski et al. (2025), who highlight that especially in open-ended tasks, metrics such as BLEU[8], ROUGE[9], and BERTScore[10] can have a hard time in capturing the correctness and domain-specific nuances of the generated outputs.

In context of this study, it is important to note that we are dealing with *expert* feedback specifically. Even though the use of LLMs has gained significant popularity also in assessment stage ("*LLM-as-a-Judge"*), their suitability in evaluating domain-specific tasks is not as straight-forward (Szymanski et al., 2025). We believe the expertise of the technical writers, who are the potential end users of the model, to be very valuable also at the evaluation stage. We also believe that given the nature of their writing

[8] N-gram co-occurance-based metric developed by Papineni et al. (2002). Originally intended for measuring quality of machine translations.
[9] Umbrella term by Lin (2004), which stands for different metrics intended for measuring quality of summaries.
[10] Evaluation metric of generated text developed by Zhang et al. (2020), focusing on token-based contextual word embedding similarities.

task - being rather strict and tied to specific writing guidelines - we can expect a rather high inter-rater agreement.

Inspired by the paper of Ouyang et al. (2022), we involved three held-out evaluators in the evaluation process along the three writers who participated in the demonstration data collection. These held-out evaluators were not involved in the demonstration data collection. This allowed us to reduce possible bias in the evaluation stage, since it is possible for the model to overfit the annotations of the three participants.

3.5.2 Evaluation Setup

We carried out the expert evaluations in-person with each evaluator. The evaluations were performed on paper, in order to create a more straight-forward evaluation experience as well as allow the evaluators to easily underline words/sentences from the instances.

Before the evaluation, we asked the evaluators to fill in a pre-study questionnaire (Appendix A). Although not central to our study interests, we asked the participants to answer a few multiple-choice questions regarding their experience of and attitudes towards using AI tools as well as in their experience in technical writing.

Example evaluation formulas are available in Appendix B. The examples in the formulas do not contain any real data from our study due to confidentiality.

In order to familiarize the evaluators with the task, in the first two instances of the evaluation formula we simply asked the evaluators to rank the set of answers/outputs from best to worst. Subsequently, we specified that the ranking of the answers/outputs should be done regarding specific aspects. Table 12 lists the aspects that were specified along with the guiding questions.

Table 12. Evaluation Criteria

Aspect / Criterion	**Guiding question**
General accuracy	Is the answer precise? Is the answer free from false information?
Helpfulness	Does the answer correspond to the prompt? Does the answer provide information that is useful? Does the answer "solve" the prompt?
Formatting/structure	Does the formatting of the answer (paragraph, lists, line breaks etc.) correspond to your preference?
Simplified Technical English (STE)	Does the answer comply with STE guidelines? Does it satisfy (your own and) Saab's standards?

As seen in Table 12, we were interested in four different aspects. Due to our dataset containing various different types of instruction prompts, it was important for us to specify the assessment criteria in order to collect detailed insights into the performance of the model.

The first four instances the participants evaluated were a shared subset across all of the participants. With this subset, our goal is to perform small-scale investigation on inter-evaluator agreement by calculating mean Kendall's W. The following seven instances in the evaluation formula were unique to each evaluator, meaning that we collected 42 unique instances in total.

We performed the evaluation as blinded, meaning that the evaluators were not informed which outputs were produced with which model. Furthermore, also gold outputs were included in the selection. The

order of the outputs on the evaluation sheet was always shuffled, and we also ensured that each evaluator never came across an instance they had annotated themselves.

The evaluators were asked to note down the reasoning for their choices. They were instructed that they could leave their comments on the assigned lines or in a free format, for instance by underlining words or sentences from the outputs and writing comments next to them.

Additionally, we conducted a pilot study with a Saab employee before the evaluations with the technical writers. This employee does not work as a technical writer but has work experience with technical documents in general. This pilot study allowed us to not only monitor the time it would take to evaluate the instances but also add needed details in the instructions.

3.6 Methodological Differences Compared to Bengtsson and Habbe (2024)

The main differences between the methodology of our study and the one adopted in Bengtsson and Habbe (2024) can thus be summarized in the following points:

- We carefully curated the dataset used in model fine-tuning (SIFT) and chose not to use any synthetic or machine translated data in the training and testing. The model is also fine-tuned with longer inputs.
- We involved a sample of the targeted end users of the LLM (the technical writers) in the data collection as well as in evaluation.
- A more qualitative approach is adopted for the evaluation of the results.

4 Results

In this chapter, we will discuss the results of our evaluation.

We will start by presenting the overall results regarding the ranking process done by the evaluators. Subsequently, we will present the results on the four evaluation aspects that we had specified in the evaluation formula.

In each subsection, we will discuss and analyze the comments left by the evaluators. In all the tables and discussion of this section, ***base model*** **refers to the Mistral-7b-Instruct-v.02 model, and** ***fine-tuned model*** **refers to the base model weights of Mistral-7b-Instruct-v.02 that we have merged with our trained adapters.**

4.1 Model Preference

Each rater evaluated 11 instances (4 shared and 7 unique), meaning we collected in total 66 evaluations.

Table 13 illustrates the overall results of the ranking of the outputs and the average rating given for the best output.

Table 13. Overall Model Preferences of the Evaluators

Output from:	**Ranked best in % of all instances (66):**	**Average rating of output quality (when ranked as best)[11] and sample standard deviation:**
Fine-tuned model	45 % (30)	4,15±0,96
Base model	17 % (11)	3,70±0,50
Gold (AMP) / Expert	38 % (25)	4,07±1,07

We see that the fine-tuned model outputs were preferred in 30 of the 66 instances, gold/expert outputs in 25, and base model's outputs in 11. The rating given for the output quality shows only a small difference between the fine-tuned model output and gold/expert output. Sample standard deviations prove moderate variability in the given ratings.

Additionally, with 45 of the total 66 instances, that is **68%** of the time, the evaluators preferred the fine-tuned model's outputs to the base model.

4.1.1 General Accuracy

Each rater evaluated three instances where the specific evaluation aspect was general accuracy. In total, we collected 18 of these instances.

With this aspect, we wanted the evaluators to focus on the accuracy on the content without a specific emphasis on STE. However, we noted that these aspects are understandably intertwined, and many evaluators seemed to have considered STE as a "subcategory" of accuracy.

[11] This was done on a Likert scale from 1 to 5, where 1 is very poor, and 5 is very good.

Here are examples of the prompts that were used in this evaluation aspect:

- *"Transform/rewrite this Swedish text into Simplified Technical English (STE): [*input*]."*
- *"Translate/transform/rewrite this text into STE: [*input*]."*
- *"What is this paragraph about: [*input*]."*
- *"Does this text contain any sentence that has the word [*non-STE word*]? List those sentences to me."*

Table 14 illustrates the results of this aspect.

Table 14. Model Preferences with Focus on General Accuracy

Output from:	**Ranked best in % of all instances (18):**	**Average rating of output quality (when ranked as best) and sample standard deviation:**
Fine-tuned model	33% (6)	4±0,70
Base model	28% (5)	3,6±0,57
Gold (AMP) / Expert	39% (7)	3,7±0,95

We can see that in terms of this evaluation aspect, the two models' outputs and the gold output got ranked as the best almost equally as often. The average quality rating of the fine-tuned model output is slightly higher, but it is important that we also note the impact of the low number of instances on the percentages and average scores. Standard deviation values are moderate to low, ratings given to gold/expert outputs varying slightly more.

Additionally, in 10 out of the 18 instances (**56%**), the raters preferred the fine-tuned model's outputs to the base model.

By analyzing the comments and justifications left in the evaluation formulas, we were able to identify the following main reasons for **when our fine-tuned model's output was preferred to the base model's output**:

- Base model output contained words (especially verbs) that were either non-STE, inconsistent, or hard to understand. Fine-tuned model output retained the content from the prompt better and did not introduce non-typical wording that was seen in base model outputs such as "perform" (instead of "do"), "take out" (instead of "remove"), "initiate" (instead of "start"), "terminate" (instead of "stop"), or "as stated in" (instead of "refer to").

- Base model output contained mistranslations (from Swedish to STE). However, some of these mistranslations were present also in fine-tuned model output. Base model output contained misinformation (several extra items to install that were not present in the prompt and an added heading that was not appropriate).

- Base model output had at times missing parentheses for item positions, and letters were used in a position indication, which is incorrect.

And when the **base model's outputs were preferred to the fine-tuned model's outputs** (44% of the time), the main comments can be summarized in these bullet points:

- Base model output was accurate and the details were good, but the wording was worse in terms of STE. The STE was better with the fine-tuned model output, but it had too much extra information. Fine-tuned model output missed a specific image reference from the input and a section about recommended torque values.

- Base model output had the best structure - the instructions were structured in steps for each action.

- Base model output was more detailed than fine-tuned model's output. Fine-tuned model output had a peculiar sentence added to the end.

- Fine-tuned model contained a sentence that had misinformation regarding the aircraft item.

4.1.2 Helpfulness

Each rater evaluated two instances in terms of their helpfulness, resulting in 12 instances in total.

The instances in subcategory did not contain any prompts asking the model to translate to transform input text into STE. Instead, here are examples of the prompts that were used in this evaluation aspect:

- *"Clarify this paragraph: [*input*]."*
- *"What is this text about /help me understand what this text is about: [*input*]."*
- *"What is happening in this text: [*text*]."*
- *"Explain the main information of this paragraph: [*input*]."*
- *"What does [*technical term/phrase*] mean/define [*technical term/phrase*] in this context: [*input*]."*

Table 15. Model Preference with Focus on Helpfulness

Output from:	**Ranked best in % of all instances (12):**	**Average rating of output quality (when ranked as best) and sample standard deviation:**
Fine-tuned model	50% (6)	4,8±0,44
Base model	25% (3)	3,6±0,57
Expert	25% (3)	5±0

From Table 15, we can observe an almost identical rating given to the quality of expert and fine-tuned model outputs, with expert output reaching a perfect average score. However, the fine-tuned model output got ranked as the best more often. Regarding the results of this aspect, we believe that having generated the base outputs for the expert annotation phase, and the experts having used those as a "starting point" for the post-editing may have influenced the results. The base model and expert outputs were at times not very dissimilar. The average quality rating given to the base model is worse, reaching only 3,6. The sample standard deviations are low across all given ratings.

When evaluating the outputs in terms of helpfulness, the evaluators preferred the fine-tuned model's outputs to the base model's **58%** of the time (7 out of the total 12 instances).

By analyzing the comments and justifications left in the evaluation formulas, we were able to identify the following main reasons for **when our fine-tuned model's output was preferred to the base model's output:**

- The base model output was too long – a lot of text for not much content. The fine-tuned model output was short, easy to read, concise, and correct. However, it did at times also have some unnecessary information at the end of the output.

- Base model output contained misinformation, and it had given the wrong meaning to an aircraft acronym.

And when the **base model's outputs were preferred to the fine-tuned model's outputs** (42% of the time), the main comments are summarized in the following bullet points:

- Base model output was well explained, and it was long enough with suitable level of difficulty. It also contained a word choice that suited the prompt well. Fine-tuned model output made sense, but it contained a long list that was a little bit unnecessary.

- The formatting used in the base model output was better.

4.1.3 Format

Each rater only evaluated one instance regarding its format/structure. This means we only collected 6 instances with focus on format. Our analysis of the sampled generations during the fine-tuning process motivated us to include also this aspect in the evaluation.

Here are examples of the prompts that were used in this evaluation aspect:

- *"Rewrite/translate/transform this text using Simplified Technical English (STE): [*input*]."*
- *"If this text contains any sentence that has a list of items ([*examples*]) – format those in a vertical list using either dashes, bullet points, letters or numbers: [*input*]."*

Table 16 illustrates the results.

Table 16. Model Preference with Focus on Format

Output from:	**Ranked best in % of all instances (6):**	**Average rating of output quality (when ranked as best) and sample standard deviation:**
Fine-tuned model	50% (3)	4,3±1,15
Base model	33% (2)	4±0
Gold (AMP) / Expert	17% (1)	4±0

The results prove to be quite even, the fine-tuned model's outputs preferred 3 out of 6 times. This is interesting due to the fine-tuned model's outputs being always formatted simply as a paragraph of text without line breaks or list formatting, whereas these were present in the base model's outputs. This most likely indicates that errors regarding STE or the content in general might have weighed in more in the ranking process, despite us having specified the evaluation aspect. This can also be seen in the relatively high sample standard deviation value of the fine-tuned model outputs.

It is important to note that the gold outputs are concatenated subtasks, and the XML-tags that allow hierarchical structures and other formatting types of text are not present. When transformed into the

final pdf-format, the XML tags do allow the creation of various types of lists. This most likely explains why they did not get ranked as the best as often.

By analyzing the comments and justifications left in the evaluation formulas, we were able to identify the following main reasons for **when our fine-tuned model's output was preferred to the base model's output** (50% of the time, 3 out of total 6 instances):

- Base output contained many non-STE-words, used not allowed verb tenses, missed definite articles, and listed multiple steps in a single sentence (which is not allowed in STE).

And when the **base model's outputs were preferred to the fine-tuned model's outputs** (50% of the time, 3 out of 6 total instances), the main comments are summarized in the following bullet points:

- In terms of formatting, breaking up text into distinct steps is good/useful. A wall of text is not preferred. This is why the fine-tuned model's output was hard to read.

- Fine-tuned model did not format the output according to the prompt instruction. The base model's output was better, although not very good either, but a good starting point.

4.1.4 Simplified Technical English (STE)

Our raters evaluated three instances each where the focus was on Simplified Technical English (STE) standard. This results in 18 instances in total. With this evaluation criteria, we intended the raters to pay attention particularly to the level of adherence to STE of the outputs, opposed to their general accuracy.

Here are examples of the prompts that were used in this evaluation aspect:

- "*Rewrite/transform/translate this text using Simplified Technical English (STE): [*input*]."*
- *"Rewrite this sentence / these sentences using active voice: [*input*]."*
- *"What words could I use to replace [*non-STE word*] in this context: [*input*]."*
- *"Transform this Swedish text into STE: [*input*]."*
- *"How could I rewrite this sentence without using the word [*non-STE word*]: [*input*]."*

Table 17 illustrates the results of this aspect.

Table 17. Model Preference with Focus on STE

Output from:	**Ranked best in % of all instances (18):**	**Average rating of output quality (when ranked as best) and sample standard deviation:**
Fine-tuned model	50% (9)	3,3±1,25
Base model	6 % (1)	3±0
Gold (AMP) / Expert	44 % (8)	3,8±1,32

In this subcategory, the difference in how often the base and fine-tuned models' outputs were ranked as the best is the most drastic, as the base model's output was ranked as the best only once. We can see from Table 17 that the fine-tuned model's outputs and the gold outputs were ranked as the best almost as often. Additionally, the quality ratings given to these two are quite similar – the gold outputs did receive a slightly higher average rating. The standard deviations illustrate a moderate variation in the ratings.

Our model seems to have performed the best on this criterion, as the participants preferred our fine-tuned model to the base model in **72%** of the instances. This meant that the fine-tuned model was preferred in 13 out of the total 18 instances.

By analyzing the comments and justifications left in the evaluation formulas, we were able to identify the following main reasons for **when our fine-tuned model's output was preferred to the base model's output**:

- The fine-tuned model output looked very satisfactory in terms of STE, as it consisted of short, straightforward, directive sentences with STE-approved words. Base model output contained unusual verb choices like "turn on", "allow the…", "consult", which are not good in terms of STE. Base model output contained characters that are not STE-approved.

- Base model output was divided into different steps, one for each action, which is correct according to STE. However, there were quite a lot of words that were not STE (judged as more severe mistake).

- Base model output offered word choices that were not context relevant.

- Fine-tuned model broke up sentences, referred to AMP, and used words in a way that is STE. It lost some meaning from the prompt though, but the STE was good. "Make sure", "Refer to", "Put"…is typical STE.

And when the **base model's outputs were preferred to the fine-tuned model's outputs** (28% of the time), the main comments are summarized in the following bullet points:

- Base model output was organized according to aircraft model-specific information.

- Fine-tuned model output did not mention a security step mentioned at the beginning of the prompt. (SE-STE instance)

4.1.5 Inter-rater Agreement

As for the shared subset that all the evaluators completed (4 shared instances, 4*6=24), despite the small number of shared instances, we calculated the mean Kendall's W. Proposed in Kendall and Smith (1939), Kendall's W is a statistic for measuring interrater agreement, and it is particularly suitable for ordinal data. In our study, the mean Kendall's W across the three answer options was **0.722**, indicating a moderate inter-rater agreement among the evaluators.

$$W = \frac{12S}{m^2(n^3 - n)}.$$

Figure 8. Formula of Kendall's W where S is the sum of the squared deviations from the mean rank sum, m is the number of raters, and n is the number of options in each instance.

4.2 Feedback on Gold Outputs (AMPs)

Having included also gold outputs from the AMPs into the evaluation formulas, we were able to collect comments about them, too. From instances where the gold outputs were **not** ranked as the best answer, we have listed the three most common remarks left by the evaluators.

Gold output uses non-standard or inconsistent phrasing (in terms of STE)

- "Do a leak check." vs. "Do a check for leaks."
- "As shown in the AMP" vs. "Refer to the AMP"
- "No leaks are permitted." vs. "Leaks are not permitted." vs. "Repair all leaks."

Gold output contains spelling errors or incorrect capitalization

- "o-ring s" instead of "o-rings"
- Some aircraft parts were capitalized in the gold outputs, even though they are not supposed to be. According to the writing rules, aircraft parts (not referring to tools, spares, or chemicals) should only be capitalized when they are followed by an acronym.

Gold output contains another type of deviation from STE

- Two working steps formatted in a single sentence. According to STE, each procedural sentence should consist of only a single working step (unless the steps need to be performed simultaneously).

Despite these types of mistakes not being very common in the AMPs, these results still indicate that the final AMPs might contain some inconsistencies or minor deviations from STE.

4.3 Pre-study Questionnaire Results

Having analyzed the results of the pre-study questionnaire, we found that the majority of the participants reported little or some experience with using AI in general. The use of AI in their personal lives was slightly more frequent than the use of it for technical writing. Most of the participants reported either a positive or very positive general attitude toward AI. A few participants reported a neutral attitude.

We did not observe any relationship between the participants' answers to these background questions and the ranking of the outputs in the shared subset.

5 Discussion

It seems that our fine-tuned model performed well, outperforming the base model in 68% of the evaluated instances. Additionally, the outputs generated with the fine-tuned model received an average quality rating of 4.13 out of 5. We saw that our model was able to produce helpful and accurate answers, containing fewer hallucinations and inaccuracies compared to the base model. Additionally, the inter-rater agreement was relatively high, as the calculated mean Kendall's W came out to 0.72. Our findings thus align with the positive correlation between human-post editing of LLM outputs and qualitative human evaluation presented in Occhipinti et al. (2024).

The comments accompanying the ratings allowed us to learn more about the model performance. We noted that our model performed particularly well in terms of the STE evaluation criterion. The model seemed to have learnt to generalize STE-aspects to some extent – performing necessary vocabulary changes when needed while not forcing word replacements on already approved verb forms. It also formatted working steps in short sentences and seems to have learnt the standardized phrases that are often repeated in the AMPs.

In terms of content-related questions, our model seems to produce more concise answers. Interestingly, we noted that where some of the evaluators seemed to prefer longer, more elaborate model outputs to prompts such as "*What is this text about /help me understand what this text is about: [*input*]*." or "*What does [*technical term/phrase*] mean/define [*technical term/phrase*] in this context: [*input*]*.", some evaluators preferred shorter, more concise outputs.

However, in regard to some aspects, our model did not outperform (or was even outperformed by) the base model. The base model usually reformatted the source documents according to the working steps usually very well and was formatting them using numbers or bullet points. Our model was mainly producing paragraphs of text and lists that were not formatted vertically. In this regard, we were expecting the raters to unanimously prefer the base model outputs to the fine-tuned model outputs when ranking the instances in this aspect. This was because our own observation was for the base model outputs to be more pleasant to read – they were more often formatted using line breaks, lists, and shorter paragraphs. However, noting that 50% of the raters still preferred the fine-tuned model's outputs, we concluded that some of the raters might have valued the factual content over the format of the answers, even when the latter was specified as the evaluation criteria.

Our model also made occasional mistakes when translating from Swedish to STE, left out item positions, or had included unnecessary information when prompted with instructions asking about specific technical tools or content. However, all in all our model contained significantly fewer hallucinations. For instance, our model did not include any hallucinations regarding the aircraft model, whereas the base model was at times coming up with its own views of the aircraft model in question (such as *A-10* and *Thunderbolt*).

Furthermore, we believe that one of the biggest reasons for the raters not having always preferred the gold output is that during the data collection, the expert annotators were offered the base model output as a starting point for the post-editing or rewriting. This might have introduced some bias toward the base model outputs. Another reason for this is most likely the fact that having stripped the target text out of its XML-format (S1000D) might have caused some information loss in the format and even content (headings or model-specific information).

Having included the gold outputs from the AMPs in the evaluation formula allowed us to witness that even final publications might at times contain minor deviations from the STE writing standard. This illustrates that an AI-based tool could be used as a complement to expert knowledge in the technical writing process.

Lastly, we would like to mention some important limitations regarding this type of technical documents, especially when prompting the models to "translate" technical source documents into (company) writing standards. As we were not implementing any information retrieval methods (such as Retrieval Augmented Generation (RAG)), the LLM relies purely on the input document and the learnt weights from the training. It does not have inherent knowledge about technical procedures or about safety precautions and issues. Furthermore, the transformation of document references from their source format into target requires additional detail, as the document references do not look the same in the source and target documents. Lastly, we were enlightened by a technical writer that for instance, position numbers for items in different procedures do not always have an identical correspondence between the source documents written by the maintenance engineers and the final Aircraft Maintenance Publications. This is partly due to the engineers sometimes using different or older versions of drawings, which do not correspond to the position numbers used in the current publications. Below is a fictional example of this.

Source document	**Target document (AMP)**
"Remove the screw (5) from the front wheel."	"Remove the screw (6) from the front wheel."

In conclusion, it can be stated that technical writing process at Saab Aeronautics is an intricate process that extends far beyond simply translating documents into STE. The task requires combining domain knowledge with project-and client-specific requirements while adhering to various writing standards. On top of having to perform additional research, the writers often collaborate with other experts at Saab, depending on the project at hand.

5.1 Delimitations of the Study

Firstly, only three technical writers participated in the annotation process. This meant that the amount of training data we were able to collect during the demonstration data collection was rather limited. Involving human effort in the data cleaning, annotation process and evaluation is a conscious choice to spend more time and resources on the project (Zheng et al., 2023). Furthermore, bias and inter-rater agreement level in human annotations is another aspect to consider, although inter-rater agreement proved to be relatively high in our study. Additionally, the amount of data in the parallel corpus was also relatively low.

Secondly, it is important to note the possibility of the model slightly overfitting, since the testing data is very similar to the training data. Naturally, we made sure there was no data contamination during the data splitting process, although many instances in the dataset contain similar text sequences. For instance, sentences like "*Start/Stop the APU. Refer to the maintenance action XX.*" were very frequent in the dataset, and several instances could have contained the same sentence. This is understandable, as these types of subtasks are needed with specific remove and install procedures. It would thus be very interesting to test the inference of our fine-tuned model with other types of Gripen procedures as well as out-of-distribution examples. It would also be interesting to collect viewpoints from more technical writers on a larger scale, regarding many other types of potential tasks they feel an LLM-powered chatbot could assist them with.

Lastly, it is also worth noting that the group of raters in the evaluation phase consisted of six technical writers. In order to draw more robust conclusions, more evaluators might be needed.

5.2 Ethical Aspects of the Study

Use of LLMs in a high-stake context as the defense industry requires an amplified ethical discussion. First and foremost, LLMs may produce information that is not fully accurate, or is even completely wrong (aka *hallucinations*). Fine-tuning and other post-tuning techniques can help fight against this; however, ultimately the user is the one who is responsible for verifying the factualness and accuracy of the LLM generation. In contexts where precision, safety, and compliance with various standards is crucial this problem gets emphasized, and the question regarding whom ultimate responsibility falls onto is not necessarily simple.

Additionally, despite research into mitigating the issue and LLMs nowadays managing to pass explicit bias tests, the data used in the pre-training process of LLMs may still result in biased outputs (Bai et al., 2025; Ling et al., 2025). Even if this consideration can be seen as to apply to the use of LLMs on a wider scope, it is still an important aspect to consider, especially since the use of inclusive language is also discussed in the STE writing guidelines.

We believe sufficient employee training regarding AI-innovations within the technical writing field is tied to ethical considerations. If an LLM-powered chatbot gets deployed and its use is tested on a wider scope within the technical writing at Saab Aeronautics, we believe that additional training aimed for the users could already help in preventing some of the potential issues. This training could include an introduction to the basics of AI, prompting LLMs, and most importantly the limitations of LLMs, covering the above-mentioned challenges.

6 Conclusion & Future Work

The goal of this study was to explore the potential of LLM's assistance in the technical documentation writing process at Saab. Specifically, on the Generative AI framework of Getto et al., (2025), this study has focused on a specific subcategory of the technical writing tasks that could benefit from AI-enhancement – creation of new Aircraft Maintenance Publications.

We involved the target users, experts, throughout our study. We collected information about the technical writing process, with special interest in aspects with which AI could provide facilitation. Subsequently, we conducted data collection together with the writers in order to compile a high-quality demonstration dataset. We also conducted a thorough investigation of the source and target data. With the goal of creating a high-quality parallel corpus, we combined the use of regular expressions with manual annotation work. These expert demonstration data and parallel corpus were then combined to form the data to be used in the fine-tuning process.

The training of Mistral-7b-Instruct-v.02 was done using parameter-efficient methods on a single 48GB GPU Accelerator. We performed a small-scale hyperparameter search and sampled generations during the training process in order to investigate the optimal settings for both the LoRA and training hyperparameters. Expert evaluation was performed with the three technical writers that were involved in the data collection, and three held-out technical writers who had not been involved in our study beforehand. In a blinded manner, the raters were asked to rank different outputs with various assessment criteria specified.

6.1 Addressing Research Questions

In our work, we were interested in the following questions:

To what extent can fine-tuning of an instruction-tuned LLM with expert-annotated data and a curated parallel corpus produce expert-level outputs?

- Our results showed that the expert raters preferred the outputs generated by the fine-tuned model over the base model outputs 68% of the time. This indicates that the fine-tuning process was able to improve the performance of the model in this domain-specific task.

What kind of qualitative evaluation do the experts provide about the fine-tuned model's performance, and in what ways does their feedback differ from that of the base model?

- In order to answer this question in detail, we specified four different aspects for the raters to focus on during the evaluation. When assessing the model performances with regard to the **general accuracy**, the fine-tuned and base model seemed to achieve similar performances. However, when adherence to STE was considered as a subcategory to accuracy, fine-tuned model outperformed the base model.
- In regard to **helpfulness**, our model's outputs were generally assessed as very helpful and more concise than base model outputs. However, some evaluators preferred longer, more detailed outputs generated by the base model.
- In terms of the **format** of the outputs, multiple raters stated that reading text that had been formatted in steps, bullet points or uses line breaks as more enjoyable to read. The base model was performing better in achieving this.
- Our model performed particularly well in terms of **STE**-compliance: the raters appreciated especially the adherence to STE-approved vocabulary and short, simple sentences. The base

model was often generating not-STE approved verb forms, and when it suggested word replacements, the options were often not context-relevant.

By adopting a more hands-on approach, what type of solution can we offer for the source-target data alignment problem in Bengtsson and Habbe (2024)?

- We were able to investigate the source and target data on a deeper level and find corresponding text units. By using regular expressions and manual annotation we managed to create a parallel corpus consisting of corresponding procedures from the source and target documentation.

In conclusion, the results of this study showed promising results in improving the base model performance by using a high-quality dataset for parameter-efficient instruction-tuning. The small size of our dataset entails obvious caution in interpretating the results, but we believe the time spent on creating the dataset and the efforts put into maintaining its high quality to have compensated for that to some extent. We were also happy to find that the technical writers report a generally positive attitude towards AI-based tools. We strongly believe the expertise and experience of the technical writers is essential in planning the process of incorporating AI-assistance in their work, and we see our findings encouraging for future collaboration. Within the scope of our study, we hope that our work has proved that in domain-specific open-ended generation tasks, expert involvement can bring value in the fine-tuning of an LLM.

6.2 Future Work

We see that there are several paths to explore when it comes to the future extensions of this study.

First and foremost, it would be essential to explore **training the model with more data** consisting of different types of procedures, descriptions, and inputs of different formats. This approach, possibly combined with testing the model with out-of-distribution data, would allow us to better see to which extent the model has learnt to generalize, especially regarding the task of "translating" text to Simplified Technical English. To create more high-quality parallel data needed for supervised training, it could be worthwhile to investigate how, for instance, an LLM-powered AI-Agent[12] could assist in the alignment task of creating the parallel corpus. The restrictions of the development environment need to be considered, however. This could be combined with an investigation into different **prompting techniques**, as suggested by Getto et al., (2025), or into different models available in Saab's development environment.

Once the model has been fine-tuned with more (diverse) data, **exploring different hyperparameters** in order to find an optimal balance between base model and adapter behavior would be interesting. This could allow the fine-tuned model to format the outputs in a more user-friendly way. This could then be extended to **Reinforcement Learning from Human Feedback (RLHF)**, where the goal is to better align the model's performance with human preferences by training a separate reward model with preference data (i.e. the rankings from our study) and then updating the model's policy with the help of the reward model.

Additionally, given that the source and target documents used in this study were full of references to other documents as well as image figures, we think it could be valuable to investigate a way of **encoding their meaning** in the model outputs. This could be initiated by something as simple as creating a mapping between the letter and number combinations and their meaning.

[12] Software components that autonomously make decisions in order to reach set goals, powered by an LLM (Wang et al., 2024).

Lastly, we think that **Retrieval Augmented Generation (RAG)** could be an efficient solution for grounding the model's output in the domain even better. In addition to getting access to more data, this would entail the creation of a vectorized database. It would be interesting to compare the differences between a fine-tuned and a RAG-based model when prompted with questions such as the following examples:

- "What kind of lubricant is usually used for spare X?"
- "When is step X needed during procedure Y for Gripen C?"

We hope that this study has provided insight both into the possibilities and potential of LLM-assistance in technical writing, while taking into consideration the complexity and demands of the technical writing process in general.

References

Acrolinx. (n.d.). *Product*. https://www.acrolinx.com/product/

Aerospace Security and Defence Industries Association of Europe. (2025). *ASD-STE100*. https://www.asd-ste100.org/

Aerospace Security and Defence Industries Association of Europe. (2025). *About STE*. https://www.asd-ste100.org/about_STE.html

Bai, X., Wang, A., Sucholutsky, I., & Griffiths, T. L. (2025). Explicitly unbiased large language models still form biased associations. *Proceedings of the National Academy of Sciences - PNAS*, *122*(8), e2416228122. https://doi.org/10.1073/pnas.2416228122

Bengtsson, H., & Habbe, P. (2024). Direct Preference Optimization for Improved Technical Writing Assistance: A Study of How Language Models Can Support the Writing of Technical Documentation at Saab. (Dissertation). https://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-205071

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, *33*, 1877-1901.

Chen, H., Zhang, Y., Zhang, Q., Yang, H., Hu, X., Ma, X., ... & Zhao, J. (2023). Maybe only 0.5% data is needed: A preliminary exploration of low training data instruction tuning. *arXiv preprint arXiv:2305.09246*.

Common Crawl. (2025). *Overview*. https://commoncrawl.org/overview

Dettmers, T., Pagnoni, A., Holtzman, A., & Zettlemoyer, L. (2023). Qlora: Efficient finetuning of quantized llms. *Advances in neural information processing systems*, *36*, 10088-10115.

Getto, G., Kelley, S., & Vance, B. (2025). How to Write With GenAI: A Framework for Using Generative AI to Automate Writing Tasks in Technical Communication. *Journal of Technical Writing and Communication*. https://doi.org/10.1177/00472816251332208

Honda. (1984). *Official Honda shop manual: CB 550 SC, CB 650 SC Nighthawk (61ME503)*. Internet Archive. https://archive.org/details/official-honda-shop-manual-cb-550-sc-cb-650-sc-nighthawk-61-me-503-27008408-1983-1984-1985/mode/2up

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., ... & Chen, W. (2022). Lora: Low-rank adaptation of large language models. *ICLR*, *1*(2), 3.

Hugging Face. (n.d.). *Bitsandbytes*. https://huggingface.co/docs/transformers/en/quantization/bitsandbytes

Hugging Face. (n.d.). *BitsAndBytesConfig*. https://huggingface.co/docs/transformers/v4.50.0/en/main_classes/quantization#transformers.BitsAndBytesConfig

Hugging Face. (n.d.). *Trainer*. https://huggingface.co/docs/transformers/en/main_classes/trainer

Jurafsky, D., & Martin, J. (2025). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, 3rd edition. Online manuscript released January 12, 2025. https://web.stanford.edu/~jurafsky/slp3/

Kendall, M. G., & Smith, B. B. (1939). The Problem of m Rankings. *The Annals of Mathematical Statistics*, *10*(3), 275–287. https://doi.org/10.1214/aoms/1177732186

Lin, C. Y. (2004). Rouge: A package for automatic evaluation of summaries. *Text summarization branches out*, 74-81.

Ling, L., Rabbi, F., Wang, S., & Yang, J. (2025, April). Bias Unveiled: Investigating Social Bias in LLM-Generated Code. *Proceedings of the AAAI Conference on Artificial Intelligence*, *39*(26), 27491-27499.

Occhipinti, D., Marchi, M., Mondella, I., Lai, H., Dell'Orletta, F., Nissim, M., & Guerini, M. (2024). Fine-tuning with HED-IT: The impact of human post-editing for dialogical language models. *Findings of the Association for Computational Linguistics: ACL 2024*, 11892–11907.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., ... & Lowe, R. (2022). Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, *35*, 27730-27744.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002, July). Bleu: a method for automatic evaluation of machine translation. *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, 311-318.

Qiu, X., Sun, T., Xu, Y., Shao, Y., Dai, N., & Huang, X. (2020). Pre-trained models for natural language processing: A survey. *Science China technological sciences*, *63*(10), 1872-1897.

Saab. (n.d.). *Company in brief*. https://www.saab.com/about/company-in-brief

de Souza, K., Nikolaev, A., & Koponen, M. (2025). Generative AI for Technical Writing: Comparing Human and LLM Assessments of Generated Content. *Proceedings of the Joint 25th Nordic Conference on Computational Linguistics and 11th Baltic Conference on Human Language Technologies (NoDaLiDa/Baltic-HLT 2025)*, 661-679.

Sun, X., Ji, Y., Ma, B., & Li, X. (2023). A comparative study between full-parameter and lora-based fine-tuning on chinese instruction data for instruction following large language model. *arXiv preprint arXiv:2304.08109*.

Szymanski, A., Ziems, N., Eicher-Miller, H. A., Li, T. J. J., Jiang, M., & Metoyer, R. A. (2025). Limitations of the llm-as-a-judge approach for evaluating llm outputs in expert knowledge tasks. *Proceedings of the 30th International Conference on Intelligent User Interfaces*, 952-966.

S1000D. (2025). https://s1000d.org/

Van der Lee, C., Gatt, A., Van Miltenburg, E., Wubben, S., & Krahmer, E. (2019). Best practices for the human evaluation of automatically generated text. *Proceedings of the 12th International Conference on Natural Language Generation*, 355-368.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.

Vieira, I., Allred, W., Lankford, S., Sheila Castilho Monteiro De Sousa, & Way, A. (2024). How Much Data is Enough Data? Fine-Tuning Large Language Models for In-House Translation: Performance Evaluation Across Multiple Dataset Sizes. *Proceedings of the 16th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, 236-249.

Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., ... & Wen, J. (2024). A survey on large language model based autonomous agents. *Frontiers of Computer Science*, *18*(6), 186345. https://doi.org/10.1007/s11704-024-40231-1

Welch, C., Gu, C., Kummerfeld, J. K., Perez-Rosas, V., & Mihalcea, R. (2022). Leveraging similar users for personalized language modeling with limited data. *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers),* 1742-1752. https://doi.org/10.18653/v1/2022.acl-long.122

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). BERTscore: Evaluating text generation with BERT. *8th International Conference on Learning Representations, ICLR 2020.*

Zheng, L., Chiang, W. L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., ... & Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, *36*, 46595-46623.

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., ... & Levy, O. (2023). Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, *36*, 55006-55021.

Appendices

Appendix A: Pre-study Questionnaire

Pre-study questionnaire

- How many years of experience do you have in technical writing? _____
- How would you rate your familiarity / experience with using AI? * Please circle your answer.
    A. No experience whatsoever
    B. Little experience
    C. Familiar with using AI
    D. A lot of experience

- How often do you use AI in your personal life?
    A. Never
    B. Sometimes
    C. From time to time (at least once a week)
    D. Daily or multiple times a day

- Have you used any AI tools* to assist in your technical writing tasks/work in general? Either Saab's internal tools or external. Please circle your answer.
    A. Never
    B. Sometimes
    C. Frequently

- What is your general opinion/attitude towards AI?
    A. Very negative / skeptical
    B. Neutral
    C. Positive
    D. Very positive

**For instance, chatbots and generative AI like ChatGPT.*

Appendix B: Example Evaluation Formulas

INSTANCE: ACC-1

Rank the answers in terms of OVERALL ACCURACY.

- Is the answer precise? Free from false information?

PROMPT (to the AI system)
Transform this text into Simplified Technical English: Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in erat enim. Aenean venenatis nisi ex. Aliquam at condimentum urna. Vivamus pharetra metus tempor, ultrices est sed, scelerisque ante.

A	**B**	**C**
Quisque vel sollicitudin metus. Fusce magna velit, eleifend in ipsum vitae, ullamcorper fermentum nibh. Nulla molestie ac dui at viverra.	In vitae semper mi. Aliquam a tortor interdum, congue ligula nec, placerat eros. Aenean quis luctus urna, sit amet ultricies sapien. Donec id tempus sapien.	Nunc risus dui, egestas eget ultrices quis, imperdiet pretium risus. Aliquam in viverra purus, et sagittis erat. Fusce tellus nisi, dapibus id est quis, finibus vestibulum erat. Sed interdum congue commodo.

RANK FROM BEST TO WORST:

1. ___ and rate the quality of this answer on a scale of 1 to 5 ____ (1= very poor, 5=very good).
2. ___
3. ___

REASONING / COMMENTS (can also be written next to the ratings):

_____

_____

_____

INSTANCE: HEL-1

Rank the answers in terms of HELPFULNESS.

- Does the answer correspond to the prompt? Does the answer provide information that is useful? Does the answer "solve" the prompt?

PROMPT (to the AI system)
Define the term X in the context of this text: Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in erat enim. Aenean venenatis nisi ex. Aliquam at condimentum urna. Vivamus pharetra metus tempor, ultrices est sed, scelerisque ante.

A	**B**	**C**
Quisque vel sollicitudin metus. Fusce magna velit, eleifend in ipsum vitae, ullamcorper fermentum nibh. Nulla molestie ac dui at viverra.	In vitae semper mi. Aliquam a tortor interdum, congue ligula nec, placerat eros. Aenean quis luctus urna, sit amet ultricies sapien. Donec id tempus sapien.	Nunc risus dui, egestas eget ultrices quis, imperdiet pretium risus. Aliquam in viverra purus, et sagittis erat. Fusce tellus nisi, dapibus id est quis, finibus vestibulum erat. Sed interdum congue commodo.

RANK FROM BEST TO WORST:

1. ____ and rate the quality of this answer on a scale of 1 to 5 ____ (1= very poor, 5=very good).
2. ____
3. ____

REASONING / COMMENTS (can also be written next to the ratings):

_____

_____

_____

INSTANCE: FOR-1

Rank the answers in terms of FORMATTING/STRUCTURE.

- Does the formatting of the answer (paragraph, lists, line breaks etc.) correspond to your preference?

PROMPT (to the AI system)
Translate this text into STE: Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in erat enim. Aenean venenatis nisi ex. Aliquam at condimentum urna. Vivamus pharetra metus tempor, ultrices est sed, scelerisque ante.

A	**B**	**C**
Quisque vel sollicitudin metus. Fusce magna velit, eleifend in ipsum vitae, ullamcorper fermentum nibh. Nulla molestie ac dui at viverra.	In vitae semper mi. Aliquam a tortor interdum, congue ligula nec, placerat eros. Aenean quis luctus urna, sit amet ultricies sapien. Donec id tempus sapien.	Nunc risus dui, egestas eget ultrices quis, imperdiet pretium risus. Aliquam in viverra purus, et sagittis erat. Fusce tellus nisi, dapibus id est quis, finibus vestibulum erat. Sed interdum congue commodo.

RANK FROM BEST TO WORST:

1. ___ and rate the quality of this answer on a scale of 1 to 5 ____ (1= very poor, 5=very good).
2. ___
3. ___

REASONING / COMMENTS (can also be written next to the ratings):

_____

_____

_____

INSTANCE: STE-1

Rank the answers in terms of STE.

- Does the answer comply with STE guidelines? Does it satisfy (your own and) Saab's standards?

PROMPT (to the AI system)
Rewrite these sentences using active voice: Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in erat enim. Aenean venenatis nisi ex. Aliquam at condimentum urna. Vivamus pharetra metus tempor, ultrices est sed, scelerisque ante.

A	**B**	**C**
Quisque vel sollicitudin metus. Fusce magna velit, eleifend in ipsum vitae, ullamcorper fermentum nibh. Nulla molestie ac dui at viverra.	In vitae semper mi. Aliquam a tortor interdum, congue ligula nec, placerat eros. Aenean quis luctus urna, sit amet ultricies sapien. Donec id tempus sapien.	Nunc risus dui, egestas eget ultrices quis, imperdiet pretium risus. Aliquam in viverra purus, et sagittis erat. Fusce tellus nisi, dapibus id est quis, finibus vestibulum erat. Sed interdum congue commodo.

RANK FROM BEST TO WORST:

1. ___ and rate the quality of this answer on a scale of 1 to 5 ____ (1= very poor, 5=very good).
2. ___
3. ___

REASONING / COMMENTS (can also be written next to the ratings):

_____

_____

_____