Universität Zürich[UZH]

Master Thesis

for obtaining the academic degree

Master of Informatics

from the Faculty of Business, Economics, and Informatics

Using Large Language Models (LLMs) to Expand Condensed Coordinated German and English Expressions into Explicit Paraphrases

Author: Kartikey Sharma

Matriculation-Nr: 20-744-595

Referentin/Referent: Prof. Dr. Martin Volk

Supervisor: Dr. Simon Clematide

Institut für Informatik

Submission Date: (6. August 2023)

Abstract

This master's thesis explores fine-tuning Large Language Models (LLMs) to reformulate condensed coordinated expressions found in job postings. This kind of condensed coordinated expression is frequently used in job postings, which is our target text genre for this work. Four gold-standard datasets were created for two tasks in English and German.

The first task focuses on truncated word completion, where elided text like "*Haus- und Gartenarbeit*" (house and garden work) needs to be completed to "*Hausarbeit und Gartenarbeit*". The German GS dataset consists of 510 samples, while the English GS contains 402 samples. The primary goal is to assess the LLMs' performance in this task and identify promising models for the second, more complex task.

The second task involves expanding condensed coordinated soft-skill requirements like *"Sie arbeiten sehr selbständig, ziel- und kundenorientiert"* into explicit self-contained paraphrases such as *"Sie arbeiten sehr selbständig, arbeiten zielorientiert und arbeiten kundenorientiert"*. To achieve a proper mapping of soft-skill requirements to a detailed domain ontology, it is crucial to provide self-contained text spans that refer to a single concept. For creating the German GS, we utilized In-Context Learning with ChatGPT, providing 5 examples in the prompt to generate additional samples. Subsequently, these samples were used to fine-tune GPT-3 and later manually verified to form a GS dataset comprising 1968 samples.

In the first task, T5-large, and FLAN-T5-large, and GPT models showed similar levels of accuracy. However, in the second task, T5-large and FLAN-T5-large performed poorly. To improve results, we applied PEFT-based techniques, LORA, to fine-tune BLOOM, T5-Large, FLAN T5-XXL, and mT5-XL on a single GPU. Among these, GPT-3 demonstrated superior performance, closely followed by mT5-XL in overall evaluations. For evaluation, we measured how incomplete soft skill text spans were completed, assessed both completed and incomplete soft skills, and evaluated overall sentence similarity. Error metrics such as Rouge-L, average Levenshtein distance, % of matched skills, and Cosine Similarity were used to evaluate soft skill changes and overall text similarity. In conclusion, Large Language Models (LLMs) effectively expanded condensed coordinated expressions into simpler formulations, including completing hyphenated words in German, without relying on traditional methods sensitive to grammatical and spelling errors.

Zusammenfassung

Diese Arbeit untersucht die Fähigkeit von Large Language Models (LLMs), kondensierte koordinierte Ausdrücke in Stellenanzeigen expliziter zu reformulieren. Diese Art Ausdrücke wird häufig in Stellenausschreibungen verwendet, die zugrundeliegende Textgattung für diese Arbeit sind. Vier Gold-Standard-Datensätze wurden für je zwei Aufgaben in Englisch und Deutsch erstellt.

Die erste Aufgabe konzentriert sich auf die Vervollständigung von verkürzten Wörtern, zum Beispiel soll "*Haus- und Gartenarbeit*" zu "*Hausarbeit und Gartenarbeit*" ergänzt werden. Der deutsche Datensatz besteht aus 510 unterschiedlichen Beispielen, der englische aus 402. Damit soll die Leistung der LLMs in dieser Aufgabe erhoben und geeignete Modelle für die zweite, komplexere Aufgabe identifiziert werden.

In der zweiten Aufgabe werden kondensierte koordinierte Soft-Skill-Anforderungen wie *Sie arbeiten sehr selbständig, ziel- und kundenorientiert* in explizite, in sich geschlossene Paraphrasen wie *Sie arbeiten sehr selbständig, arbeiten zielorientiert und arbeiten kundenorientiert* erweitert. Um eine korrekte Abbildung von Soft-Skill-Anforderungen auf eine detaillierte Domänenontologie zu erreichen, ist es entscheidend, inhaltlich Textabschnitte bereitzustellen, die sich auf ein einzelnes Konzept beziehen. Für die Erstellung des deutschen GS haben wir In-Context Learning mit ChatGPT verwendet, mit jeweils 5 Beispielen in der Eingabe. Danach wurde die manuell korrigierte Ausgabe iterativ für das Optimieren von GPT-3-basierten Modellen verwendet und letztlich ein Datensatz mit 1'968 Beispielen erstellt.

Die erste Aufgabe lösten die Modelle T5-large und FLAN-T5-large sowie GPT mit hoher Genauigkeit. Bei der zweiten Aufgabe jedoch schnitten T5-large und FLAN-T5-large schlecht ab. Bessere Resultate erhielten wir mit PEFT-basierten Feinabstimmungstechniken von BLOOM, T5-Large, FLAN T5-XXL und mT5-XL an. GPT-3 zeigte die beste Leistung, dicht gefolgt von mT5-XL. Für die Bewertung haben wir folgendes gemessen: die Ergänzung von unvollständigen Soft-Skill-Segmenten, die Ähnlichkeit aller (auch vollständigen) Segmente, sowie die Ähnlichkeit des ganzen Satzes. Metriken wie Rouge-L, die Levenshtein-Distanz, % der übereinstimmenden Fertigkeiten und die Cosinus-Ähnlichkeit wurden zur Bewertung der Soft-Skill-Änderungen und der Gesamttextähnlichkeit verwendet. Zusammenfassend lässt sich sagen, dass LLMs kondensierte koordinierte Ausdrücke effektiv in einfachere Formulierungen umwandeln können, einschliesslich der Vervollständigung von Wörtern mit Auslassungsstrichen im Deutschen, ohne auf herkömmliche morphologische und korpusstatistische Methoden zurückgreifen zu müssen.

Acknowledgement

Completing this master's thesis would not have been possible without the support, encouragement, and guidance of several individuals.

First and foremost, I would like to express my sincere appreciation to my thesis advisor, Dr. Simon Clematide. His invaluable guidance and constant support played a critical role in shaping the research process and the final outcome of this thesis. Dr. Clematide's deep understanding of the field, insightful ideas, and meticulous feedback significantly contributed to the success of this work. I am sincerely grateful to him for his mentorship.

I would also like to thank the Department of Computational Linguistics at the University of Zurich for providing me with the resources and facilities necessary to conduct my research.

Lastly, I would like to thank my parents, my sister, my brother-in-law, my girlfriend, and my friends for their unwavering love and encouragement which have been my source of strength and motivation. Their belief in my abilities inspired me to strive for excellence.

Completing this thesis would not have been possible without the collective support of these wonderful individuals. This thesis is dedicated to them with love and gratitude.

Contents

List of Figures

List of Tables

List of Acronyms

SJMM	Swiss Job Market Monitor
MLM	Masked Language Modelling
ESCO	European Skills, Competences, Qualifications and Occupations
NLP	Natural Language Processing
GPT	Generative Pre-trained Transformer
LLM	Large Language Models
ICL	In-Context Learning
RNN	Recurrent Neural Networks
BERT	Bidirectional Encoder Representations from Transformers
RoBERTa	Robustly Optimized BERT Approach
CLM	Causal Language Modeling
MLM	Masked Language Modeling
BLOOM	BigScience Large Open-science Open-access Multilingual Language Model
LoRA	Low-Rank Adaptation of Large Language Models
PEFT	Parameter-Efficient Fine-Tuning
T5	Text-To-Text Transfer Transformer
mT5	Multilingual Text-To-Text Transfer Transformer
BART	Bidirectional and Auto-Regressive Transformer
RegEx	Regular Expression
NER	Named entity recognition
API	Application programming interface
DIY	Do It Yourself
SMOR	Stuttgart Morphological Analyzer
W&B	Weights & Biases
GS	Gold Standard
SS	Silver Standard
C4	Colossal Clean Crawled Corpus
OWA	One Word Addition

HOWA	Hyphenated with One Word Addition
MWA	Multiple Word Addition
MC	Multiple _C Skills
HMC	Hyphenated with Multiple _C Skills
MDC	Multiple and Different _C Skills
HMDC	Hyphenated with Multiple and Different _C Skills

1 Introduction

The Swiss Job Market Monitor(SJMM) [1] plays a crucial role in monitoring and analyzing trends within the Swiss job market. By extracting information from job advertisements, this monitoring system captures a comprehensive dataset comprising print and online job ads in German, French, English, and Italian, spanning several decades from 1950 to the present day. One of the primary objectives of this monitoring initiative is to extract essential details regarding workers' skill requirements and task descriptions from the textual content of job ads and align them with corresponding classes in the multilingual European Ontology ESCO [2](European Skills, Competences, Qualifications, and Occupations).

However, an inherent challenge lies in the complex and dense formulations used to describe the skills of workers in these advertisements, often employing coordinated structures that combine multiple elements of information. For example, phrases like *"plan, organize, and maintain data storage solutions"* pose a difficulty when mapping them onto the ESCO ontology, as it typically represents each task as a separate concept, such as "*plan data storage solutions*," "*organize data storage solutions*," and "*maintain data storage solutions*." Consequently, there arises a need to develop a text simplification framework that can effectively decompose intricately coordinated statements into a series of simpler coordination-free statements, facilitating direct mapping onto the ESCO ontology.

The thesis utilizes *de_soski_ner_model*, a domain-specific spaCy [3]-based soft skill requirements text span recognizer trained within the SJMM project. This model identifies and analyzes specific soft skills that employers seek in job candidates, including communication, teamwork, adaptability, problem-solving, and leadership, among others.

The model distinguishes two categories:

- **Softskill**: A text span within a job posting that contains a complete soft skill.

[1]SJMM:`https://www.stellenmarktmonitor.uzh.ch/en.html`

[2]ESCO: `https://esco.ec.europa.eu/en/about-esco/what-esco`

[3]spaCy: `https://spacy.io/`

- **Softskill_C (Component)**: An incomplete soft skill text span that requires additional information from a preceding or following soft skill span to be fully understood.

This model is capable of identifying the text span which contains the Softskill with the F-score of 88.33 and the Softskill_C with the F-score of 88.89. For example, the below-annotated examples represent two components of the soft skills: one with an *S*, which encloses the text span which contains the complete soft skill; the second with a *C*, which represents the incomplete soft skill

1. ein [[***überzeugendes***]]C und [[vertrauenswürdiges Auftreten]]S
2. Sie sind eine [[***starke***]]C und [[zielorientierte Persönlichkeit]]S.
3. Sie haben eine [[***offence***]]C, [[***aufbauende***]]C und [[wertschätzende Haltung]]S.

The aim of this thesis is to develop methods for a text simplification framework that can effectively decompose complex coordinated statements into simpler, non-coordinated statements such as

1. ein ***überzeugendes Auftreten*** und vertrauenswürdiges Auftreten
2. Sie sind eine ***starke Persönlichkeit*** und zielorientierte Persönlichkeit
3. Sie haben eine ***offence Haltung***, ***aufbauende Haltung*** und wertschätzende Haltung

This research focuses on breaking down complex coordinated statements into simpler, non-coordinated ones, enabling the mapping of complete soft skills onto the skills and competencies defined in the ESCO ontology, commonly used in job applications.

By identifying frequently mentioned soft skills in job advertisements for different job types or industries, this study provides valuable insights into the changing demands of the labor market and the soft skills most valued by employers. This contributes to a deeper understanding of the evolving job market and specific skills required in various sectors.

The developed advanced text simplification framework enhances the efficiency and accuracy of mapping worker skill requirements and task descriptions onto the ESCO ontology. By leveraging the capabilities of LLM (Large Language Models)[4], this framework becomes robust to minor errors in the input text and eliminates the need for morphological knowledge, enabling more effective analysis and monitoring of the

[4]Large Language Models: `https://en.wikipedia.org/wiki/Large_language_model`

Swiss job market to match the right people with the right jobs. Additionally, it aids in identifying job specialization and obsolete jobs, aligning education curricula with job market skills, and improving employability.

In addition, apart from its application in job advertisements and the ESCO ontology, the idea of simplifying complex coordinated statements holds broader significance in natural language processing tasks, including machine translation, summarization, and text generation. The simplification of intricate sentences can result in more precise and efficient automated processing, benefiting content generation in educational and legal contexts, among others. This has the potential to enhance readability and improve comprehension for a wider audience. As a result, this text simplification framework can make valuable contributions to various NLP applications, promoting improved communication, information dissemination, and language comprehension across diverse domains.

1.1 Research Questions

1. **RQ1:** Is there a measurable performance difference among modern state-of-the-art transformer architectures (BART (Lewis et al. [2020]), T5 (Raffel et al. [2020]), GPT3 (Brown et al. [2020])) for learning to solve text generation problems that reformulate coordinated phrases with elided material into completed, but slightly redundant formulations?

2. **RQ2:** What are the differences in the accuracy between few shots classification using GPT3 (Brown et al. [2020]) vs fine-tuning the GPT3 (Brown et al. [2020]) model for reformulating complex coordinated phrases into simpler formulations?

3. **RQ3:** Can Large Language Models (LLM) be taught how to generate redundant and elaborate simpler phrases from the original coordinated expressions text without semantically changing the input sentence

4. **RQ4:** Can LLMs (Large Language Models) be taught how to solve the problem of ellipsis completion? Ex. Haus- und Gartenarbeit —> Hausarbeit und Gartenarbeit

We will explore and experiment with various transformer-based methods for text reformulation and simplification to achieve this. Specifically, we will investigate the efficacy of the GPT family of models by openAI[5], BART (Lewis et al. [2020]), T5-

[5]openAI: `https://openai.com/`

based text-to-text models and other open-source LLM through various fine-tuning approaches.

Efficient creation and validation of training material will be essential for the success of this project. To this end, we propose several strategies. Firstly, we can identify complex structures by analyzing the dependency parse output of the job ad texts. Additionally, we can consider data augmentation by using certain LLMs like chatGPT[6] and then manually correcting them to create a Gold Standard dataset for the training.

To facilitate the exploration and evaluation of the proposed text simplification methods, we outline a two-part approach and divide the tasks into two tasks.

- The First Task will be the "*Noun completion task*" which involves training the model to expand the truncated words. For example, expansion of **"*Ski- und Velohelmen*"** into **"*Skihelmen und Velohelmen*"**
- The Second Task would be a "*Phrase expansion task*" which would involve training the model to decompose complex coordinated statements into simpler, non-coordinated statements. For example, **"Sie arbeiten sehr selbständig, *ziel-* und *kundenorientiert*"** should be completed as **"Sie arbeiten sehr selbständig, *arbeiten zielorientiert* und *arbeiten kundenorientiert*"**. In cases where the original text is already presented in its simplest form, no changes will be made. For example, no decomposition or reformulation is required in "Du lernst *Kunden zu begeistern* und mit *Freude zu verkaufen*"

In this thesis, we will use the terms 'Noun completion task' and 'First Task' interchangeably to refer to the same task. Similarly, we will use the terms '*Phrase Expansion Task*' and 'Second Task' interchangeably. This clarification is intended to avoid any confusion and ensure consistency in the terminology used throughout the document.

For the *Noun Completion Task*, we will extract these truncated words from the job ad dataset and manually annotate approximately 500 instances, simplifying them into their constituent words. This annotated dataset will serve as our gold standard, which we will further split into training and test sets. We will explore the use of few-shot classification techniques or fine-tune the GPT3 (Brown et al. [2020]) on this dataset to learn the decomposition patterns of such hyphenated words. Alternatively, we will evaluate the performance of (BART (Lewis et al. [2020]) or T5 (Raffel et al. [2020]) architectures to compare the results. To extend

[6]chatGPT: `https://openai.com/chatgpt`

the applicability of our approach, we will also translate the gold standard dataset into English and learn to obtain simpler formulations for the English language as well.

For the *Phrase Expansion Task*, a dataset of approximately 2000 samples will be prepared, consisting of spans of text that represent skill requirement descriptions or task descriptions. The dataset will have a column labeled "prompt" containing the original text and a column labeled "completion" containing the simplified structure. The dataset creation process involves manual annotation, followed by the creation of synthetic examples using chatGPT by In-Context Learning[7]. These examples are then used iteratively to fine-tune the GPT3 (Brown et al. [2020]) in order to create more training datasets. The final results are subsequently verified manually, resulting in the creation of the gold standard (GS) dataset, which is then leveraged to fine-tune various models.

1.2 Thesis Structure

The thesis would follow the following structure:

- **Chapter 2** outlines previous work in a similar field, discussing the various techniques that have been attempted for splitting compound words and reformulating complex coordinated phrases.
- **Chapter 3** gives an overview of the data and materials used, detailing the steps taken in preparing and pre-processing the data.
- **Chapter 4** delves into the methods employed in the experiments, explaining the theories behind the models and the architectures used.
- **Chapter 5** describes the experiments conducted and the outcomes obtained. The focus is split into two parts: the first part focuses on the splitting of German compound words into simpler words, while the second part focuses on extracting skills and task descriptions from job ads in both English and German.
- **Chapter 6** examines the results and engages in discussion. Each result from the previous chapter is covered and discussed in depth.
- **Chapter 7** offers a conclusion to the research and outlines future work.

[7]In-Context Learning: `https://en.wikipedia.org/wiki/Prompt_engineering`

2 Related Work

In this section, we explore the related work that directly addresses the research questions posed in this master thesis. The sections below delve into specific aspects of the literature, each corresponding to one of the four key research questions we aim to investigate.

2.1 RQ1: Measurable Performance Difference Among Transformer Architectures for Text Generation

In the context of text generation tasks, it is essential to explore the performance differences among modern state-of-the-art transformer architectures and pre-trained models. The paper by Luo et al. [2022] provides valuable insights into the comparison of transformer-based language models for question-answering tasks. The study explores nine transformer architectures, including T5(Raffel et al. [2020]), BART (Lewis et al. [2020]), RoBERTa (Liu et al. [2019]) to draw generalizable and grounded conclusions.

Inspired by this approach, our research aims to assess the performance of transformer architectures, specifically BART (Lewis et al. [2020]), T5 (Raffel et al. [2020]), GPT3 (Brown et al. [2020]), and other LLMs in solving the text generation problem of expanding condensed coordinated phrases with elided material into completed, slightly redundant formulations. By systematically comparing these architectures, we seek to identify any measurable performance differences, enabling us to make informed choices in selecting the most suitable model for our text paraphrasing framework. The findings from this comparison will enhance our understanding of the effectiveness of different transformer architectures for text generation, thereby improving the accuracy and efficiency of our proposed framework.

2.1.1 Generating texts before Transformers

The previous generations of language models predominantly relied on Recurrent Neural Networks [1] (RNNs) as their architecture of choice. However, the effectiveness of RNNs was hindered by their inability to handle long-range dependencies of tokens within the text.

When using RNNs for text generation tasks, the model often faced limitations in predicting the next word given a sequence of words. Even with an increase in the number of preceding words, the model still struggled to achieve accurate predictions due to the presence of intricate language structures that were not effectively captured by this approach. One prominent challenge lies in the syntactic ambiguity that arises from the potential multiple interpretations of words within a sentence structure.

Examples of different kinds of ambiguities in the context of job ads are mentioned below:

- **Structural ambiguity** *"The company needs someone to manage their employees who speak multiple languages."*
  This sentence exhibits structural ambiguity due to the placement of the relative clause "who speak multiple languages." It can be interpreted either as the employees speaking multiple languages or as the person managing employees who speak multiple languages.
- **Scope ambiguity** *"Applicants must have a minimum of five years of experience in sales or marketing."*
  The phrase "in sales or marketing" introduces scope ambiguity, as it can be understood as requiring five years of experience specifically in sales or in either sales or marketing.

However, the landscape of large language models underwent a significant transformation with the groundbreaking publication of "Attention is all you need" (Vaswani et al. [2017]) by Google and the University of Toronto. The transformer architecture, among other significant factors, played a pivotal role in revolutionizing Generative AI[2] by enabling efficient scaling of models through the utilization of multi-core GPUs, parallel processing of input data, and extensive training datasets.

[1] Recurrent Neural Network: `https://en.wikipedia.org/wiki/Recurrent_neural_network`
[2] Generative AI: `https://en.wikipedia.org/wiki/Generative_artificial_intelligence`

2.1.2 Transformers

The effectiveness of the transformer architecture, as shown in Figure 1, can be attributed to its ability to process individual words in the context of a sentence. Unlike traditional approaches that focus solely on neighboring words, the transformer model establishes connections with all words in the sentence and attends to each of them. By assigning attention weights to these connections, the model gains an understanding of the interdependencies between words. These attention weights are learned during LLM training. However, it's important to note that while the models can effectively learn patterns from language data, they can still be challenged by certain complex linguistic tasks.

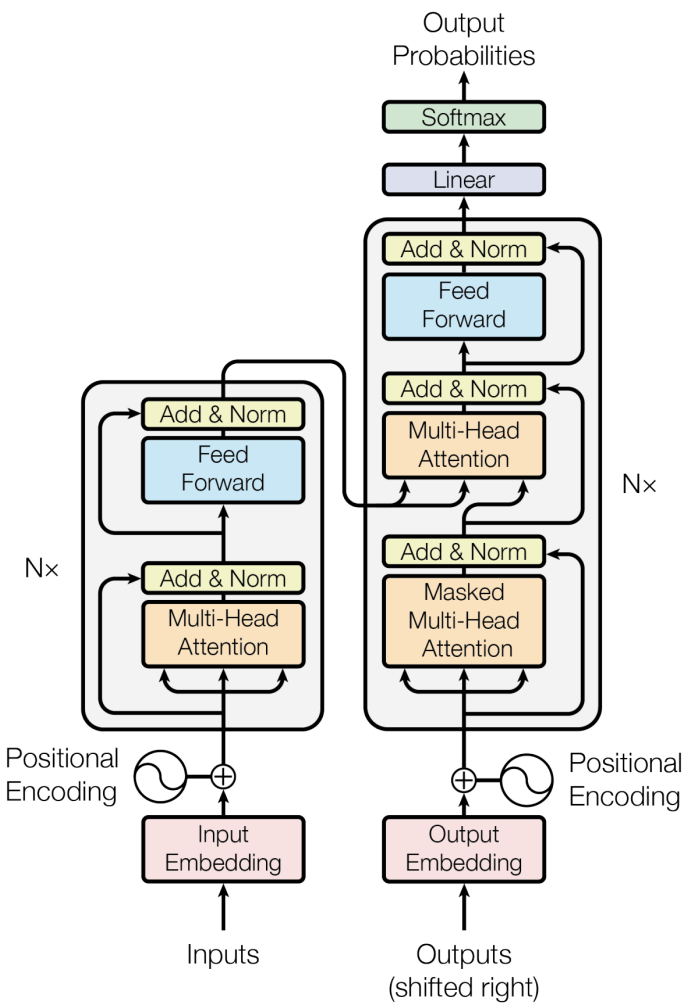

Figure 1: The Transformer Encoder-Decoder Architecture. Image Source: Vaswani et al. [2017]

2.1.2.1 Architecture

The transformer architecture, as shown in figure 1 is comprised of two main components: the encoder and the decoder. These components work together and share similarities in their functionality. To process texts using the model, the text first needs to be sub-tokenized, which means that they should be represented numerically. Once the input is encoded as numerical tokens, they are passed through the embedding layer. This layer represents each token as a vector in a high-dimensional space, allowing for the encoding of meaning and contextual information. Additionally, positional encoding is also added to represent word order and maintain the relevance of word positions within the text.

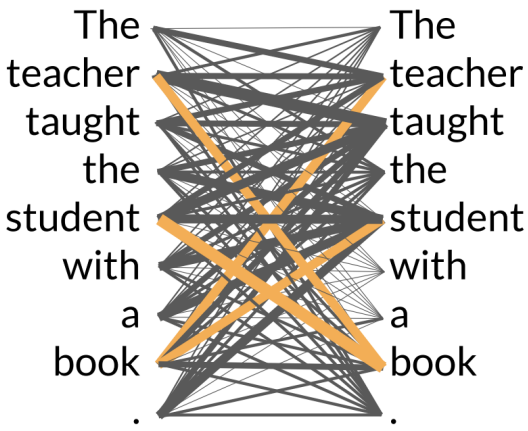

Figure 2: Self-attention depiction. Image Source: `http://deeplearning.ai/`

After adding the token vectors and positional encodings, the resulting vectors are fed into the self-attention layer. As shown in Figure 2, the model examines the relationships between tokens in the input sequence, enabling a better understanding of contextual dependencies. This process happens several times and has multi-headed self-attention. Multi-headed self-attention is a key feature of the transformer architecture, where multiple sets of self-attention weights are learned independently and in parallel. Each attention head learns different aspects of language. Once the attention weights are applied to the input data, the output is processed through a fully-connected feed-forward network.

2.1.2.2 Sequence-to-Sequence Models

These models use both an encoder and decoder from the transformer architecture and use a method known as Span Correction for pre-training the encoder part. In this process, random sequences of input tokens are masked and replaced with a unique Sentinel token (represented as **<MASK>** in Figure 3). Sentinel tokens are special tokens included in the vocabulary but do not represent any specific word from the original text. The decoder's objective is to reconstruct the masked token sequences in an auto-regressive manner. The resulting output consists of the Sentinel token followed by the predicted tokens. This method is illustrated in Figure 3. They are best suited for sequence-to-sequence tasks like machine translation, text summarisation, and Question answering. Example: T5(Raffel et al. [2020]) and BART (Lewis et al. [2020])

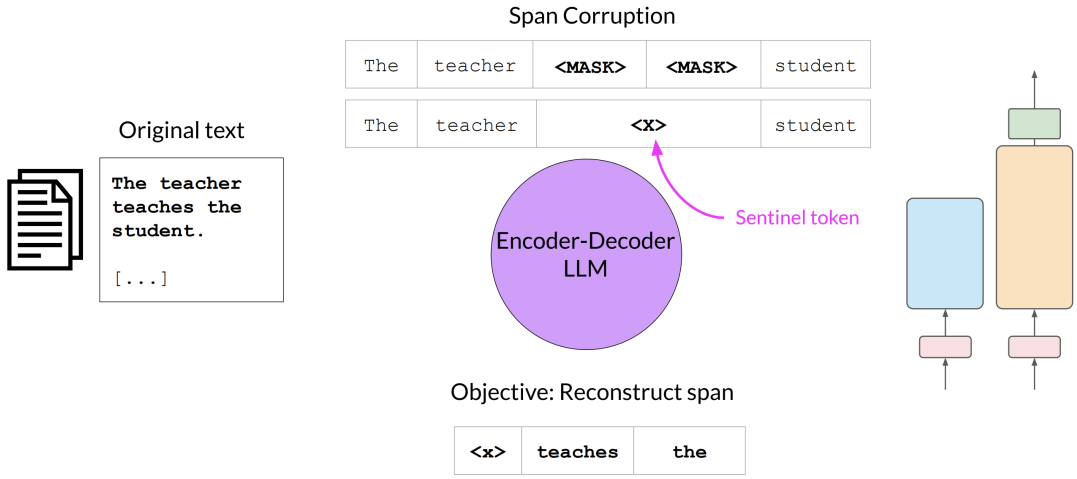

Figure 3: Sequence-to-Sequence Models. Image Source: `https://www.deeplearning.ai/`

2.1.2.3 Autoregressive models

These are the most common types of models. These types of models are pre-trained using Causal Language Modeling (CLM). Here, the objective function is to predict the next token using only the previous sequence of tokens. The attention layers can only access the words appearing before them in the sentence and do not have any knowledge about the future tokens. Hence, unlike the autoencoder models, the context here is unidirectional. They rely on the decoder component of the original architecture and are trained to predict the next token based on extensive examples, which helps them to develop a deep understanding of language. They are best suited

for the task of text generation since their pre-training involves predicting the next word in the sentence. This method is illustrated in Figure 4. Examples of such categories of models are GPT3 (Brown et al. [2020]), BLOOM (Scao et al. [2022]), etc

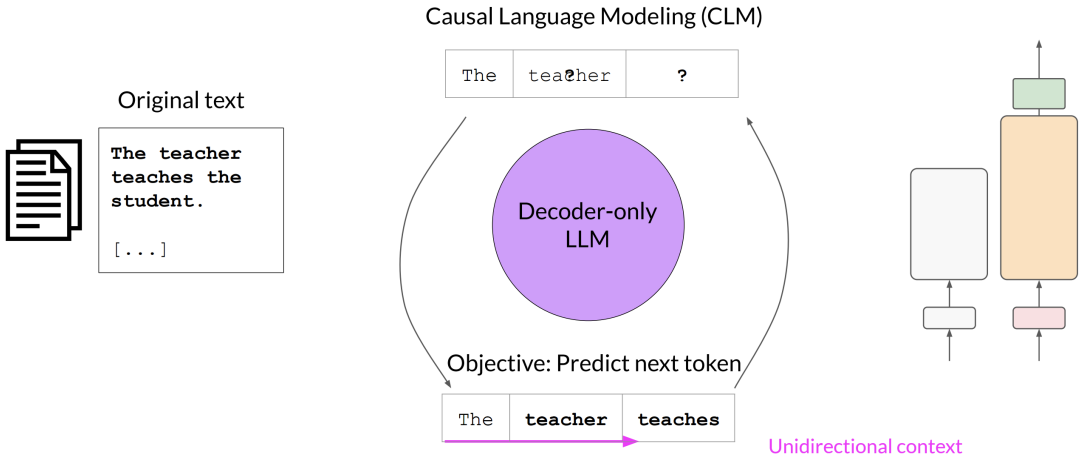

Figure 4: Autoregressive models. Image Source: `https://www.deeplearning.ai/`

We will experiment with various models from both encoder-decoder models as well as decoder-only models.

2.2 RQ2: Accuracy Differences in Few-Shot Classification Using GPT3 and Fine-Tuning for Reformulation

The literature on few-shot learning with language models presents contrasting findings that are relevant to our research question. On one hand, the paper Moradi et al. [2021] highlights that GPT-3 (Brown et al. [2020]), a powerful transformer language model, underperforms when compared to a language model fine-tuned on the full training data in the biomedical domain. Despite achieving near state-of-the-art results in few-shot knowledge transfer on open-domain NLP tasks, GPT-3's (Brown et al. [2020]) performance diminishes significantly when faced with domain-specific biomedical NLP tasks. This suggests that in-domain fine-tuning using job ad data might be essential for optimally solving our problem.

On the other hand, the paper Brown et al. [2020] demonstrates the tremendous few-shot learning capabilities of large language models, exemplified by GPT-3 (Brown

et al. [2020]), with its 175 billion parameters. In a wide array of NLP tasks, GPT-3 exhibits strong performance without any gradient updates or fine-tuning, surpassing prior state-of-the-art fine-tuning approaches. Despite some datasets where GPT-3's (Brown et al. [2020]) few-shot learning still faces challenges and methodological issues related to training on large web corpora, its ability to handle diverse tasks, including translation, question-answering, and cloze tasks, demonstrates the significant potential of few-shot learning with LLMs.

Considering these findings, our research seeks to investigate the accuracy differences in few-shot classification using GPT-3 (Brown et al. [2020]) and fine-tuned specifically for the task of reformulating complex coordinated phrases into simpler formulations in job advertisements. By examining GPT-3's performance in this domain, we aim to discern whether its few-shot learning approach can effectively handle the complexity of transforming coordinated expressions in job ads. Additionally, we will explore if GPT-3 (Brown et al. [2020]) can outperform or match the performance of a language model fine-tuned on the full training data for this specific task. This comparison will provide insights into the suitability of GPT-3's (Brown et al. [2020]) few-shot learning for reformulation in the job market domain.

2.3 RQ3: Leveraging Large Language Models for Redundant and Elaborate Phrase Generation

In recent work, Ding et al. [2023] explores the potential of GPT-3 (Brown et al. [2020]) as a data annotator for various natural language processing (NLP) tasks through three main approaches. Experimental results indicate that GPT-3 (Brown et al. [2020]) can effectively annotate data for different tasks at a relatively lower cost, making it particularly advantageous for individuals or organizations with limited budgets. Despite the cost advantage, the quality of data annotated by GPT-3 (Brown et al. [2020]) still requires improvement when compared to human-annotated data. However, even with the limitations, models trained on GPT-3 (Brown et al. [2020]) annotated data demonstrate performance comparable to or even better than those trained on human-annotated data, given the constraints of the budget.

Considering the insights from this study Ding et al. [2023], our research aims to investigate the capabilities of large language models, like GPT-3 (Brown et al. [2020]), in generating redundant and elaborate phrases from original coordinated expressions. We seek to explore whether GPT-3 (Brown et al. [2020]) can be taught to produce high-quality elaborations and redundant information while ensuring that the gener-

ated phrases maintain the intended meaning of the input sentences. By leveraging GPT-3's (Brown et al. [2020]) language generation abilities, we aim to develop a text simplification framework that can effectively decompose complex coordinated statements into a series of simpler, non-coordinated statements. The findings from this investigation can contribute to advancing the field of automatic phrase generation using large language models and shed light on potential methods to enhance the quality of generated data.

2.3.1 Prompt Tuning

In the paper by Lester et al. [2021], the authors propose a novel approach called *"soft prompts"* for enhancing model performance in text generation tasks without modifying the model weights. This mechanism aims to strike a balance between prompt engineering, where users modify the prompt text, and full fine-tuning, which involves adjusting all model parameters. Unlike prompt engineering, which only involves changing the prompt text, prompt tuning adds additional trainable tokens to the prompt and allows the supervised learning process to optimize its values. This set of trainable tokens is referred to as a *"soft prompt"* and is prepended to the embedding vectors representing the input text. The key advantage of prompt tuning is its parameter efficiency, as it requires training only a few parameters.

The prompt tuning does not exhibit the same level of performance as full fine-tuning, particularly for smaller Language Models (LLMs). However, as the model size increases, the effectiveness of prompt tuning also improves. Once the models reach a scale of approximately 10 billion parameters, prompt tuning becomes comparably effective to full fine-tuning and provides a substantial performance boost compared to prompt engineering alone.

The concept of soft prompts is highly relevant to our research as it offers a potential mechanism which can be useful to improve the performance of text generation tasks.

2.4 RQ4: Solving the Problem of Ellipsis Completion Using Large Language Models

A recent study conducted by Testa et al. [2023] explores the use of large language models (LLMs), specifically openAI's GPT-2 and BERT (Devlin et al. [2019]), for the task of retrieving elided verbs in English sentences. The goal is to determine the effectiveness of these models in reconstructing the correct event when presented

with sentences where the verb has been omitted.

In the example mentioned by Testa et al. [2023], ***The engineer completed the project*, but the student didn't**, the second part of the sentence in itself is not complete and the verb has been omitted. The expanded part could be written as ***The engineer completed the project*, but *the student didn't complete the project.***

Experimental results suggest that both GPT-2 and BERT (Devlin et al. [2019]) face challenges in accurately reconstructing the correct event in the retrieval task. As part of their future work, Testa et al. [2023] propose to leverage larger language models like GPT-3 (Brown et al. [2020]) to improve the performance of the retrieval task.

In the context of our research, these findings provide valuable insights into the challenges associated with elided text and demonstrate the potential of using GPT-3 (Brown et al. [2020]) as a candidate model for addressing this issue. Our investigation will explore whether GPT-3 (Brown et al. [2020]) can be effectively taught to generate redundant and elaborate phrases from original coordinated expressions in a way that preserves the intended meaning of the input sentences. By leveraging the power of GPT-3 (Brown et al. [2020]), we aim to enhance the efficiency and accuracy of mapping worker skill requirements to the ESCO ontology.

2.4.1 Traditional Appproaches

A study by Aepli and Volk [2013] focuses on ellipsis completion, where they aimed to reconstruct complete lemmas of truncated compounds by breaking down the full compound into its constituent elements. Specifically, they utilized Gertwol[3], a comprehensive morphology system for German, to de-construct words where all segments were known to the system.

In cases where the input word was not recognized by Gertwol, Aepli and Volk [2013] attempted to segment it using words from their corpus. They divided the compound into various possible ways, ensuring at least three characters remained on both the left and right sides. To determine the most probable split, they employed the frequencies of these words within their corpus. Subsequently, the truncated word was combined with each potential right part, and the most frequently occurring word in their corpus was adopted as the solution.

[3]Gertwol: `https://www.lingsoft.fi/`

Their study contributes significant insights into addressing the ellipsis completion challenge, employing a combination of a morphological analyzer and corpus frequencies to complete previously unseen compounds accurately.

In another study conducted by Hätty et al. [2019], the authors present a comparative evaluation of various tools for the task of splitting German compounds in the context of the German language. The main objective is to assess the performance and effectiveness of different approaches in handling compound words.

Three different tools are compared in the study, and the one that demonstrates the best performance across various types of problems is CompoST[4] (Cap et al. [2014]). The described method utilizes the geometric mean of subword frequencies to disambiguate potential splits. CompoST (Cap et al. [2014]) relies on lexicons and corpus frequencies to analyze and split compounds. It utilizes SMOR[5] (Schmid et al. [2004]), a rule-based morphological analyzer specifically designed for German, to analyze candidate items. By leveraging SMOR's (Schmid et al. [2004]) analysis capabilities, CompoST aims to identify and separate morphemes within compounds.

However, the study also highlights certain limitations associated with CompoST. One notable drawback is that the tool is unable to split words that are unknown to SMOR (Schmid et al. [2004]), as it heavily relies on SMOR's (Schmid et al. [2004]) analysis capabilities. Additionally, the disambiguation of potential splits is solely based on frequency counts obtained from the corpus. This approach may introduce inconsistencies, especially when dealing with a non-lemmatized word list.

There have been many more similar researches published on this topic, including Escartín [2014] where the researchers compare different compound splitting tools based on traditional approaches.

In the context of our research, the findings from these studies provide valuable insights into the existing challenges and limitations in handling German compound splitting. Our work aims to explore the effectiveness of LLMs in addressing similar text-processing tasks. By teaching LLMs to generate redundant and elaborate phrases from original coordinated expressions, we aim to enhance the performance of our text expansion framework for German job advertisements.

[4]CompoST: `http://dx.doi.org/10.18419/opus-3474`

[5]SMOR, a finite-state based morphological analyzer: `https://www.ims.uni-stuttgart.de/en/research/resources/tools/smor/`

3 Data

The data used in this study comes from the Swiss Job Market Monitor platform, covering the period from 1950 to 2021. The data is stored in the JSONL[1] format, with each line representing a single data entry. A sample JSONL line is provided in Table 1

Fields	Values
id	sjmm-11950111030005
year	1950
month	3
channel	1
pipeline_version	0.0pre3
text_valid	-9
length	223
language	de
tokens	{\n", "\n", "\n", "SPACE", "_SP", "sb", 0, 0, "30", 1.0}, ...}

Table 1: Sample of annotated data from SJMM

1. **id**: sjmm-11950111030005 - An identifier for the data entry.
2. **year**: 1950 - The year associated with the data.
3. **month**: 3 - The month associated with the data.
4. **channel**: 1 - The media channel associated with the data (newspaper, company website, online job portals).
5. **pipeline_version**: 0.0pre3 - The version of the pipeline used to process the data.
6. **text_valid**: -9 - A value indicating the validity of the text (-9 suggests invalid and 1 suggests valid).
7. **length**: 223 - The length of the text in characters.

[1]JSONL: `https://jsonlines.org/`

Text	Text w/ Whitespace	Lemma	POS (Broad)	POS (Fine)	Dependency	Token Index	Char Index	Zone	Zone Probability
	\n	\n	SPACE	_SP	sb	0	0	30	1.0
Gesucht	Gesucht	Gesucht	VERB	VVPP	ROOT	1	1	30	1.0
:	:	:	PUNCT	$.	punct	2	8	30	1.0
	\n	\n	SPACE	_SP	ROOT	3	9	30	1.0
Auf	Auf	Auf	ADP	APPR	mo	4	10	30	1.0
15.	15.	15.	ADJ	ADJA	nk	5	14	30	1.0
März	März	März	NOUN	NN	nk	6	18	30	1.0
oder	oder	oder	CCONJ	KON	cd	7	23	30	1.0
April	April	April	NOUN	NN	cj	8	28	30	1.0
williges	williges	williges	ADJ	ADJA	nk	9	34	80	1.0
,	,	,	PUNCT	$,	punct	10	42	80	1.0
fleissiges	fleissiges	fleissiges	ADJ	ADJA	cj	11	44	80	1.0
	\n	\n	SPACE	_SP	sb	12	54	80	1.0
Mädchen	Mädchen	Mädchen	NOUN	NN	ROOT	13	55	80	0.98
	\n	\n	SPACE	_SP	nk	14	62	80	0.98
für	für	für	ADP	APPR	mnr	15	63	60	1.0
Haus-	Haus-	Haus-	X	**TRUNC**	cj	16	67	60	1.0
und	und	und	CCONJ	KON	cd	17	73	60	1.0
Gartenarbeit	Gartenarbeit	Gartenarbeit	NOUN	NN	nk	18	77	60	1.0
.	.	.	PUNCT	$.	punct	19	89	60	1.0
Gute	Gute	Gute	ADJ	ADJA	nk	20	91	50	1.0
Behandlung	Behandlung	Behandlung	NOUN	NN	oa	21	96	50	1.0
und	und	und	CCONJ	KON	cd	22	107	50	...

Table 2: Examples of Tokens

8. **language**: “de" - The language of the text (in this case, German).
9. **tokens**: A list of dictionaries containing information about individual tokens within the text, including their text, lemma, part of speech, dependency, position, and zone.

As can be seen in Table 2, each object within the **“tokens"** array represents a token and provides the following information:

- **text**: The actual text of the token.

- **text_w_ws**: The text of the token along with whitespace characters.
- **lemma**: The lemma or base form of the token.
- **pos_broad**: The broad part-of-speech category of the token.
- **pos_fine**: The fine-grained part-of-speech category of the token.
- **dep**: The dependency relation of the token.
- **token_i**: The index of the token within the text.
- **char_i**: The character index of the token within the text.
- **zone**: The zone associated with the token.
- **zone_prob**: The probability of the token belonging to a specific zone.

This dataset allows for the analysis and exploration of job market trends over time, with detailed information about the text, its linguistic features, and other relevant metadata. The provided JSONL format enables easy processing and extraction of the required information for further research and analysis in the context of this study.

3.1 Data Preparation for *Noun Completion Task*

3.1.1 Data preparation for German language

The data preparation involved processing the JSONL files from 1950 to the present. Each JSONL file represented a job advertisement, and the data was extracted in JSON format. The "tokens" field, generated using SpaCy tokenizer, contained attributes such as "pos_broad" and "pos_fine"

For the first part of our experiments, we aimed to create a dataset specifically focused on truncated words with a hyphen at the end, found in German job ads, following the pattern *"NOUN-HYPHEN-AND-NOUN"* (e.g., "Haus- und Gartenarbeit"). The objective of this task was to provide a proof of concept and shortlist different types of Large Language Models (LLMs) for our more complex task of text simplification. We wanted to determine whether LLMs could effectively process these noun groups with ellipsis, expanding them into individual words without explicit morphological splitting. (e.g., "Hausarbeit und Gartenarbeit").

To identify job ads with truncated words, we specifically focused on the "pos_fine"

tag from the **"tokens"** field. We considered only those job ads where at least one token had a "pos_fine" tag of **"TRUNC"**, indicating the presence of a hyphen. These job ads qualified as potentially having the elliptical expression we were interested in.

Once we obtained the job ads with elliptical formations, we further processed each ad using Regular expressions[2] to specifically search for the pattern **"word-und word"**

Here's the pattern that we used and its breakdown:

- **Pattern**:

  `[A-Za-zäöüÄÖÜß]\w*\s?-\sund\s[A-Za-zäöüÄÖÜß]\w*`

- **Explanation**:
  - `[A-Za-zäöüÄÖÜß]`: Matches any alphabetical character in German (both uppercase and lowercase) and the specific German characters ä, ö, ü, Ä, Ö, Ü, and ß.
  - `\w*`: Matches zero or more word characters (letters, digits, and underscores).
  - `\s?-\s : Matches an optional space followed by a hyphen (-) and another optional space.`
  - `und`: Matches the word "und" verbatim
  - `[A-Za-zäöüÄÖÜß]`: Matches any alphabetical character in German (including accented characters).
  - `\w*`: Matches zero or more word characters.

This allowed us to extract the complete set of words exhibiting the elliptical formation. Additionally, we recorded the spans of these words within the job ad for further analysis. We extracted a total of 54736 examples in this format which were later sampled for their uniqueness and diversity.

3.1.2 Data preparation for the English language

For the purpose of creating an English dataset for the same task, we used the API from DeepL and translated the sampled German dataset into English.

[2]Regular expressions: `https://en.wikipedia.org/wiki/Regular_expression`

3.1.3 Sampling the examples

1. Data Extraction: Initially, several thousand data samples containing noun phrases with ellipsis were collected from March 1950 to March 2019.
2. Nilsimsa[3] Sampling: The Nilsimsa sampler was used to select unique cases from the extracted data for dataset creation.
3. Total of 4000 items were sampled out
4. Nilsimsa Hash Calculation: The Nilsimsa hash was computed for the noun group generating a unique signature for each noun group.
5. Similarity Check: The Nilsimsa sampler assessed if the Nilsimsa hash of a current sample matched any previously seen samples.
6. Exclusion of Similar Samples: If similarity with a previous sample exceeded the threshold, the candidate sample was excluded from the final dataset.
7. Distinct Data Samples: The Nilsimsa sampler effectively obtained distinct data samples, eliminating redundant soft skill representations based on Nilsimsa hash signatures.
8. Gold Standard Dataset: The resulting 4000 samples were considered distinct and used for preparing the manually annotated dataset.
9. English Dataset Creation: To create the English dataset, the sampled German datasets were translated, creating a parallel dataset for both tasks.
10. The Nilsimsa sampler ensured the dataset comprised diverse and unique examples of complete soft skill representations, offering value for further analysis and evaluation.

3.1.4 Gold Standard dataset creation

Once we sampled the dataset in the required format, we proceeded towards creating a Gold Standard dataset for noun groups with ellipsis for both English and German. Each of the samples was manually completed and verified. Table 3 shows the annotation examples of the German dataset. The column named "Context" was shown to the human annotator in order to understand the meaning of the elliptical construction and Table 4 shows the annotation examples of the English dataset which were translated and manually verified.

[3]Nilsimsa: `https://pypi.org/project/nilsimsa/`

The *"Ellipsis"* part of the GS was extracted using the Python code explained in the section 3.1.1. The *"Expanded"* part of the German dataset was manually curated by a native German speaker.

For the English dataset, the *"Ellipsis"* and *"Expanded"* part of the GS was obtained by translating the German dataset using the deepL[4] API and then manually verified by a native English speaker.

Context	Ellipsis	Expanded
Verkehrsmittel Attraktive Weiterbildungs- und Entwicklungsmöglichkeiten ROCKEN Partner stehen ein für interne	Weiterbildungs- und Entwicklungsmöglichkeiten	Weiterbildungsmöglichkeiten und Entwicklungsmöglichkeiten
zu den führenden Anbietern in der Stanz- und Umformtechnologie . Zur Ergänzung unseres Teams in der	Stanz- und Umformtechnologie	Stanztechnologie und Umformtechnologie
Tigers . Beteiligung beim Kauf von Ski- und Velohelmen und Vergünstigung in diversen	Ski- und Velohelmen	Skihelmen und Velohelmen
dabei Planen die Baukosten Abklären Vor- und Nachteile verschiedener Bauverfahren Erarbeiten	Vor- und Nachteile	Vorteile und Nachteile
motivierte Jugendliche mit hoher Lern- und Leistungsbereitschaft und legen grossen Wert auf	Lern- und Leistungsbereitschaft	Lernbereitschaft und Leistungsbereitschaft
Weltmarktführer für die Aufladung von Diesel- und Gasmotoren im Leistungsbereich oberhalb 500 kW	Diesel- und Gasmotoren	Dieselmotoren und Gasmotoren

Table 3: German GS dataset for *Noun Completion Task*

[4]deepL: `https://www.deepl.com/translator`

Ellipsis	Expanded
Further training and development opportunities	Further training opportunities and Further development opportunities
Stamping and forming technology	Stamping technology and forming technology
Ski and bicycle helmets	Ski helmets and bicycle helmets
Advantages and disadvantages	Advantages and disadvantages
Willingness to learn and perform	Willingness to learn and Willingness to perform
Diesel and gas engines	Diesel engines and gas engines

Table 4: English GS dataset or *Noun Completion Task*

3.2 Data Preparation for *Phrase Expansion Task*

3.2.1 Data Preparation for German

The data used in this study is also sourced from the Swiss Job Market Monitor platform. The dataset that we are using here is a quarterly dataset covering the period from 2014 September until March 2023. The data is stored in JSONL format, with each line representing a single job ad. A sample job ad extracted from the JSON line is provided in Table 5

For this task, we use the data coming from a domain-specific spaCy-based soft skill requirements text span recognizer (trained within the SJMM project using spaCy's NER[5] annotation model allowing flat non-overlapping segments). This NER model processed the *adve_text_copy* from Table 5 to recognize the soft skills as shown in Table 6. The spans can have the following labels: 'SoftSkill', and 'SoftSkill_C' depending on whether the soft skill provides self-contained text spans that refer to a single concept.

The below excerpts are from the python script *ss_sentence_extractor.py* from the GitLab repository [6]

```
nlp = de_soski_ner_model.load()
```

[5]NER: `https://en.wikipedia.org/wiki/Named-entity_recognition`

[6]Code link: `https://gitlab.uzh.ch/stellenmonitor-nfp77/student-theses/ma-kartikey-sharma`

Here, `de_soski_ner_model` refers to the domain-specific spaCy-based NER model for soft skill detection in German. This line of code loads the NER model and assigns it to the `nlp` object.

```
senter = nlp.create_pipe("sentencizer")

nlp.add_pipe("sentencizer")
```

Field	Value
occu_titl_copy	Praktikum 100%
adve_iden_adve	sjmm_sjmm_qua-2-02-2021-06-02853-1-000018488
adve_time_year	2021
adve_time_quar	2021-06-01
adve_chan_gene	2
length	794
pipeline_version	f5db770
text_valid	1
adve_text_copy	Für unsere Abteilung Handwerk & Kunst, an unserem Standort Baar-Inwil, suchen wir per 1. August 2021 für 6 Monate zwei engagierte Personen für ein Praktikum 100%. Untere Rainstrasse 31, 6340 Baar. Aufgaben: • Unterstützen und Begleiten einer Gruppe von erwachsenen Menschen mit einer psychischen, geistigen und/oder körperlichen Beeinträchtigung • Mithelfen beim Herstellen von marktfähigen Produkten mit verschiedensten Materialien wie Papier, Holz, Ton, Textil • Mithelfen beim Bearbeiten von Kundenaufträgen mit und durch die Klientinnen und Klienten Profil: • Abgeschlossene Berufsausbildung oder Matura • Kreatives und handwerkliches Geschick • Flair im Umgang mit Menschen mit einer Beeinträchtigung • Gute PC-Anwenderkenntnisse (MS Office) Kontakt: Céline Cudré, Sachbearbeiterin HR, 041 781 64 99
language	de

Table 5: Quarterly Raw Job ads data

This line of code creates a pipeline component called "sentencizer[7]" using the `create_pipe`

[7]sentencizer: `https://spacy.io/api/sentencizer`

method of the `nlp` object. The "sentencizer" component is responsible for sentence boundary detection and ensures that the text will be segmented into sentences based on the detected sentence boundaries.

```
nlp.add_pipe('paragrapher', after="transformer")
```

This line of code adds a custom pipeline component called "paragrapher" to the pipeline of the `nlp` object. Normal sentence segmentation does not work well because of the frequent use of bullet lists that lack punctuation symbols which normal sentence segmenters rely on. The paragrapher inserts sentence boundaries after `"\n\n"` tokens.

Table 7 shows the sample data once the final processing of the data has been completed. Here, *actual_sentence* extracts only the span which contains the soft skills. *entity_sent* contains the processed sentence where the special tags *<SoftSkill_C>, </SoftSkill_C>, <SoftSkill> and </SoftSkill>* are used to enclose the soft skills

3.2.2 Data Preparation for the English Language

As with the previous English Gold Standard dataset, we employed DeepL[8] using the `create_pipe` API to translate the sampled German dataset. The translated dataset underwent a manual verification process by a native English speaker, ensuring the quality of the Gold Standard dataset.

[8]DeepL: `https://www.deepl.com/en/translator`

Field	Value
adve_text_copy	Für unsere Abteilung Handwerk & Kunst, an unserem Standort Baar-Inwil, suchen wir per 1. August 2021 für 6 Monate zwei engagierte Personen für ein Praktikum 100%. Untere Rainstrasse 31, 6340 Baar. Aufgaben: • Unterstützen und Begleiten einer Gruppe von erwachsenen Menschen mit einer psychischen, geistigen und/oder körperlichen Beeinträchtigung • Mithelfen beim Herstellen von marktfähigen Produkten mit verschiedensten Materialien wie Papier, Holz, Ton, Textil • Mithelfen beim Bearbeiten von Kundenaufträgen mit und durch die Klientinnen und Klienten Profil: • Abgeschlossene Berufsausbildung oder Matura • Kreatives und handwerkliches Geschick • Flair im Umgang mit Menschen mit einer Beeinträchtigung • Gute PC-Anwenderkenntnisse (MS Office)
Entities	[(engagierte, SoftSkill), (Kreatives, SoftSkill_C), (handwerkliches Geschick, SoftSkill), (Flair im Umgang mit Menschen mit einer Beeinträchtigung, SoftSkill)]

Table 6: Quarterly Raw Job ads data - Processed using domain-specific spaCy-based soft skill recognizer

Field	Value
job_ids_list	sjmm_qua-2-02-2021-06-02853-1-000018488
year_list	2021
quarter_list	2021-06-01
adve_text_copy	Für unsere Abteilung Handwerk & Kunst, an unserem Standort Baar-Inwil, suchen wir per 1. August 2021 für 6 Monate zwei engagierte Personen für ein Praktikum 100%. Untere Rainstrasse 31, 6340 Baar. Aufgaben: • Unterstützen und Begleiten einer Gruppe von erwachsenen Menschen mit einer psychischen, geistigen und/oder körperlichen Beeinträchtigung • Mithelfen beim Herstellen von marktfähigen Produkten mit verschiedensten Materialien wie Papier, Holz, Ton, Textil • Mithelfen beim Bearbeiten von Kundenaufträgen mit und durch die Klientinnen und Klienten Profil: • Abgeschlossene Berufsausbildung oder Matura • Kreatives und handwerkliches Geschick • Flair im Umgang mit Menschen mit einer Beeinträchtigung • Gute PC-Anwenderkenntnisse (MS Office)
actual_sentence	Abgeschlossene Berufsausbildung oder Matura Kreatives und handwerkliches Geschick Flair im Umgang mit Menschen mit einer Beeinträchtigung Gute PC-Anwenderkenntnisse (MS Office)
Entities	[(Kreatives, SoftSkill_C), (handwerkliches Geschick, SoftSkill), (Flair im Umgang mit Menschen mit einer Beeinträchtigung, SoftSkill)]
entity_sent	Abgeschlossene Berufsausbildung oder Matura <SoftSkill_C>Kreatives</SoftSkill_C> und <SoftSkill>handwerkliches Geschick</SoftSkill> <SoftSkill>Flair im Umgang mit Menschen mit einer Beeinträchtigung</SoftSkill> Gute PC-Anwenderkenntnisse (MS Office)

Table 7: Post-Processed dataset for *Phrase Expansion Task*

3.2.3 Sampling the examples for *Phrase Expansion Task*

1. Data Extraction: Initially, several thousand data samples were extracted from the years September 2014 to March 2023.

2. Nilsimsa Sampler: The Nilsimsa sampler was employed to select unique cases from the extracted data for dataset creation.

3. Focus on *Complete Soft Skill Expressions*: The sampler focused on the text enclosed between the tags *<SoftSkill>(.*?)</SoftSkill>*, as this text is expected to contain a complete expression of a soft skill.

4. Criteria for Inclusion: To be included in the dataset, the sampled text had to meet several criteria:

   - The dataset should contain tokens with fewer than 25 words.
   - The language of the dataset should be German.
   - The maximum count of samples should be limited to 2000.

5. Nilsimsa Hash Calculation: The Nilsimsa hash of the text extracted from *<SoftSkill>(.*?)</SoftSkill>* was computed. This process assigned a unique Nilsimsa hash signature to each text.

6. Similarity Check: The Nilsimsa sampler then checked if the Nilsimsa hash of the current sample was similar to any of the previously seen samples.

7. Exclusion of Similar Samples: If the similarity between the candidate sample and any previously seen sample exceeded the given threshold, it indicated that the candidate sample was similar to a previously seen sample. In such cases, the candidate sample was excluded from the final sampling.

8. Distinct Data Samples: The Nilsimsa sampler effectively sampled the input data, excluding similar or redundant representations of complete soft skills based on their Nilsimsa hash signatures.

9. Gold Standard Dataset: The 2000 data samples obtained through this process were considered distinct and suitable for preparing the Gold Standard dataset.

10. By following this step-by-step approach, the Nilsimsa sampler ensured that the dataset contained unique and diverse examples of complete soft skill representations, making it valuable for further analysis and evaluation.

11. English Dataset Creation: For the English dataset, the sampled German datasets were translated to ensure a parallel dataset for both tasks.

3.2.4 Gold Standard dataset creation

Table 8 shows the subset of the final dataset for German and Table 9 shows the subset of the final dataset for English which would be further used for fine-tuning the LLM. In Table 8 there is a third column named *Problem Type*.

We decided to divide all the samples into 8 problem types as shown in Table 8.

1. **NONE** - These are the kind of problems which do not require any changes to be made in the text between *<SoftSkill_C> and </SoftSkill_C>* because the soft skill enclosed inside these tags is already complete. We decided to keep these "special" tags because, in the application phase, it will also see this kind of erroneous input and should treat it conservatively.

2. **OWA(One Word Addition)** - These types of problems only have one incomplete set of *<SoftSkill_C> and </SoftSkill_C>* tags and they can be completed by adding only one word from the text between nearby *<SoftSkill> and </SoftSkill>* tags

3. **HOWA(Hyphenated with One Word Addition)** - These types of problems are similar to OWA but they involve the completion of hyphenated text.

4. **MWA(Multiple Word Addition)** - These types of problems only have one incomplete set of *<SoftSkill_C> and </SoftSkill_C>* tags and they can be completed by adding only multiple words from the text between corresponding *<SoftSkill> and </SoftSkill>* tags

5. **MC(Multiple _C Skills)** - These types of problems have more than one incomplete set of *<SoftSkill_C> and </SoftSkill_C>* tags and they can be completed by adding one or more words from the text between corresponding *<SoftSkill> and </SoftSkill* tags. But all the incomplete soft skills can be completed by adding the same set of words. For example, in Table 8, the soft skills are completed by adding *Auftreten* to both incomplete soft skills (*freundliches* and *sauberes*)

6. **HMC(Hyphenated with Multiple _C Skills)** - These types of problems are similar to the MC types but additionally HMC problems also include hyphenated words in the incomplete soft skills

7. **MDC(Multiple and Different _C Skills)** - These types of problems have more than one incomplete set of *<SoftSkill_C> and </SoftSkill_C>* tags and they can be completed by adding one or more words from the text between corresponding *<SoftSkill> and </SoftSkill* tags. But the only difference from MC

types of problems is that these problems require different sets of words to complete the soft skills. In the mentioned example Skill_C> ausdauernde </SoftSkill_C> and <SoftSkill_C> sicherheitsbewusste </SoftSkill_C> were completed by adding “Persönlichkeit” and “Arbeitsweise” respectively.

8. **HMDC(Hyphenated with Multiple and Different __C Skills)** - Exactly similar to MDC but incomplete soft skills contain hyphenated words too.

Prompt	Completion	Problem Type
Du lernst<SoftSkill>Kunden zu begeistern</SoftSkill> und **<SoftSkill_C>mit Freude zu verkaufen**</SoftSkill_C>.	Du lernst <SoftSkill>Kunden zu begeistern</SoftSkill> und **<SoftSkill>mit Freude zu verkaufen</SoftSkill>**.	NONE
Sie verfügen über <SoftSkill>unternehmerisches Denken</SoftSkill> sowie **<SoftSkill_C>Handeln</SoftSkill_C>**	Sie verfügen über <SoftSkill>unternehmerisches Denken</SoftSkill> sowie **<SoftSkill>unternehmerisches Handeln**</SoftSkill>	OWA
<SoftSkill_C>Kommunikations-</SoftSkill_C> und <SoftSkill>Teamfähigkeit</SoftSkill>	**<SoftSkill>Kommunikationsfähigkeit</SoftSkill>** und <SoftSkill>Teamfähigkeit</SoftSkill>	HOWA
<SoftSkill>Organisationstalent</SoftSkill>, <SoftSkill>Verantwortungsbewusst</SoftSkill> Mit der **<SoftSkill_C>Fähigkeit</SoftSkill_C>** und <SoftSkill>Freude im Team zu arbeiten</SoftSkill> und **<SoftSkill_C>etwas zu bewegen</SoftSkill_C>**!	<SoftSkill>Organisationstalent</SoftSkill>, <SoftSkill>Verantwortungsbewusst</SoftSkill> Mit der **<SoftSkill>Fähigkeit im Team zu arbeiten</SoftSkill>** und <SoftSkill>Freude im Team zu arbeiten</SoftSkill> und **<SoftSkill>etwas zu bewegen</SoftSkill>**!	MWA
<SoftSkill_C>freundliches</SoftSkill_C>, **<SoftSkill_C>sauberes</SoftSkill_C>** und <SoftSkill>motiviertes Auftreten</SoftSkill>	**<SoftSkill>Freundliches Auftreten</SoftSkill>**, **<SoftSkill>sauberes Auftreten</SoftSkill>** und <SoftSkill>motiviertes Auftreten</SoftSkill>	MC
Sie <SoftSkill>arbeiten sehr selbständig</SoftSkill>, **<SoftSkill_C>ziel-</SoftSkill_C>** und **<SoftSkill_C>kundenorientiert</SoftSkill_C>**	Sie <SoftSkill>arbeiten sehr selbständig</SoftSkill>, **<SoftSkill>arbeiten zielorientiert</SoftSkill>** und **<SoftSkill>arbeiten kundenorientiert</SoftSkill>**	HMC
Als **<SoftSkill_C>ausdauernde</SoftSkill_C>** und <SoftSkill>proaktiv handelnde Persönlichkeit</SoftSkill> legen Sie grossen Wert auf eine **<SoftSkill_C>sicherheitsbewusste</SoftSkill_C>** und <SoftSkill>effiziente Arbeitsweise</SoftSkill>.	Als **<SoftSkill>ausdauernde Persönlichkeit</SoftSkill>** und <SoftSkill>proaktiv handelnde Persönlichkeit</SoftSkill> legen Sie grossen Wert auf eine **<SoftSkill>sicherheitsbewusste Arbeitsweise</SoftSkill>** und <SoftSkill>effiziente Arbeitsweise</SoftSkill>.	MDC
Dipl. Pflegefachfrau/-mann HF/FH, Berufserfahrung im Akutspital **<SoftSkill_C>Innovative</SoftSkill_C>**, **<SoftSkill_C>kooperative</SoftSkill_C>** und <SoftSkill>dynamische Persönlichkeit</SoftSkill> <SoftSkill>Freude an selbstständiger Arbeitsweise</SoftSkill> **<SoftSkill_C>Hohe Organisations-</SoftSkill_C>** und <SoftSkill>Kommunikationsfähigkeit</SoftSkill>	Dipl. Pflegefachfrau/-mann HF/FH, Berufserfahrung im Akutspital **<SoftSkill>Innovative Persönlichkeit</SoftSkill>**, **<SoftSkill>kooperative Persönlichkeit</SoftSkill>** und <SoftSkill>dynamische Persönlichkeit</SoftSkill> <SoftSkill>Freude an selbstständiger Arbeitsweise</SoftSkill> **<SoftSkill>Hohe Organisationsfähigkeit</SoftSkill>** und <SoftSkill>Kommunikationsfähigkeit</SoftSkill>	HMDC

Table 8: German GS for *Phrase Expansion Task*

Prompt	Completion
<SoftSkill_C>High communication</SoftSkill_C> and <SoftSkill>negotiation skills</SoftSkill>	<SoftSkill>High communication skills</SoftSkill> and <SoftSkill>Negotiation skills</SoftSkill>.
Are you a <SoftSkill_C>versatile</SoftSkill_C>, <SoftSkill_C>motivated</SoftSkill_C> and <SoftSkill>learning-oriented person</SoftSkill> who <SoftSkill>enjoys actively designing and developing sophisticated solutions, especially in the field of digitalisation</SoftSkill>?	Are you a <SoftSkill>versatile person</SoftSkill>, <SoftSkill>motivated person</SoftSkill> and <SoftSkill>learning-oriented person</SoftSkill> who <SoftSkill>enjoys actively designing and developing sophisticated solutions, especially in the field of digitalisation</SoftSkill>?
- <SoftSkill>Convincing appearance</SoftSkill>; <SoftSkill_C>dynamic</SoftSkill_C> and <SoftSkill>resilient personality</SoftSkill>, who aligns his or her objective to above-average performance results.	- <SoftSkill>Convincing appearance</SoftSkill>; <SoftSkill>Dynamic personality</SoftSkill> and <SoftSkill>Resilient personality</SoftSkill>, who aligns his or her objective to above-average performance results.
<SoftSkill>Enjoyment in teaching</SoftSkill> and in the <SoftSkill_C>personality development of young people</SoftSkill_C	<SoftSkill>Enjoyment in teaching</SoftSkill> and <SoftSkill>Enjoyment in personality development of young people</SoftSkill>.
- <SoftSkill>Interest in technical work</SoftSkill> and <SoftSkill_C>Procedures</SoftSkill_C> and <SoftSkill>Ability to work on and solve problems</SoftSkill	- <SoftSkill>Interest in technical work</SoftSkill> and <SoftSkill>Interest in procedures</SoftSkill> and <SoftSkill>Ability to work on and solve problems</SoftSkill>.
- <SoftSkill_C>independent</SoftSkill_C>, <SoftSkill_C>exact</SoftSkill_C> and <SoftSkill>reliable work style</SoftSkill>; <SoftSkill>commitment</SoftSkill> and <SoftSkill>resilience</SoftSkill	- <SoftSkill>independent work style</SoftSkill>, <SoftSkill>exact work style</SoftSkill> and <SoftSkill>reliable work style</SoftSkill>; <SoftSkill>commitment</SoftSkill> and <SoftSkill>resilience</SoftSkill>.
You are <SoftSkill>able to manage projects prudently</SoftSkill> and <SoftSkill_C>goal-oriented manner</SoftSkill_C>, with <SoftSkill_C>clear</SoftSkill_C> and <SoftSkill>level-appropriate communication</SoftSkill> and <SoftSkill>social competence</SoftSkill>.	You are <SoftSkill>able to manage projects prudently</SoftSkill> and <SoftSkill>able to manage projects in a goal-oriented manner</SoftSkill>, with <SoftSkill>clear communication</SoftSkill> and <SoftSkill>level-appropriate communication</SoftSkill> and <SoftSkill>social competence</SoftSkill>.
Your <SoftSkill_C>abstract</SoftSkill_C>, <SoftSkill_C>analytical</SoftSkill_C> and <SoftSkill>systematic way of thinking</SoftSkill>, sound experience and <SoftSkill>innovative ideas</SoftSkill> <SoftSkill> you actively bring into the projects</SoftSkill>.	Your <SoftSkill>abstract way of thinking</SoftSkill>, <SoftSkill>analytical way of thinking</SoftSkill> and <SoftSkill>systematic way of thinking</SoftSkill>, sound experience and <SoftSkill>innovative ideas</SoftSkill> <SoftSkill> you actively bring into the projects</SoftSkill>.

Table 9: English GS for *Phrase Expansion Task*

3.2.4.1 Dataset Bootstrapping

- We employed the Dataset Bootstrapping strategy shown in Figure 5 to create the Gold Standard (GS) for the German dataset. The process involved extracting the *"Prompt"* part of the dataset using a Python script, as explained in section 3.2. The *"Completion"* part was manually curated based on these prompts.

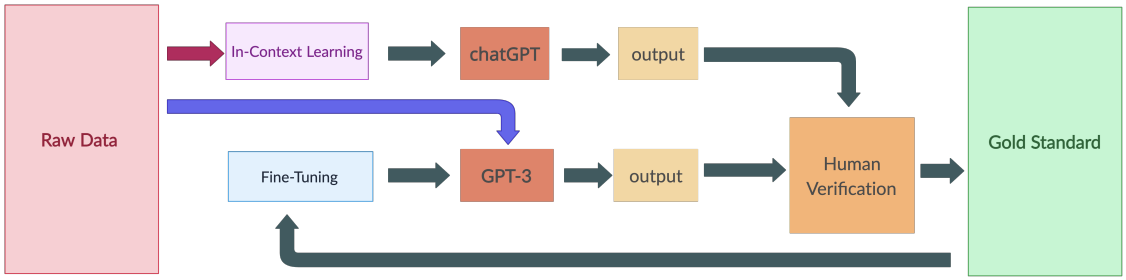

Figure 5: Dataset Bootstrapping

- The manual curation process began with the creation of 10 samples. These samples were used as examples in In-Context learning with chatGPT[9](GPT-3.5 Feb 13 version), which helped engineer[10] suitable prompts for generating more *"Completion"* samples. We used a prompt that explained the task and provided several examples.
- The prompt that we ultimately selected is shown in Figure 6.
- Using this approach, around 22 more samples were generated. We then tried to fine-tune[11] the GPT-3 (Brown et al. [2020]) model with a total of 32 samples. However, the results from GPT-3 (Brown et al. [2020]) were not as meaningful as expected. Therefore, we continued the In-Context Learning approach with chatGPT to generate a larger dataset containing 177 GS samples.
- With 177 GS samples, GPT-3 (Brown et al. [2020]) was fine-tuned again, and inference was run on 150 examples, which showed significant improvement in performance.
- After receiving results from chatGPT or GPT-3 (Brown et al. [2020]), two human annotators verified the samples. One annotator was a native English speaker with some knowledge of German, and the other was a native German

[9]chatGPT: `https://openai.com/chatgpt`

[10]Prompt engineering: `https://en.wikipedia.org/wiki/Prompt_engineering`

[11]Fine-tuning: `https://en.wikipedia.org/wiki/Fine-tuning_(deep_learning)`

speaker, providing two pairs of eyes for the verification process to classify the samples for the Gold Standard dataset.

- This iterative process was repeated, leading to the training of four variants of GPT-3 (Brown et al. [2020]) while building the GS dataset. The four variants of GPT-3 were fine-tuned on 32, 177, 315 and 768 samples. This can be viewed in Table 10

Samples for Fine-Tuning	Samples Generated	GS created
32	150	145
177	150	138
315	500	485
768	1200	1200 (Silver-Standard)
		1968 (GS+SS)

Table 10: Creation of GS by fine-tuning GPT-3 (Brown et al. [2020]) iteratively for soft skill paraphrasing

- In the end, we obtained a total of **1968 samples** with *"Prompt"* and *"Completion"* pairs. However, out of these, only 768 samples were manually corrected, while the remaining 1200 samples were designated as silver standard. The silver standard samples were generated using the GPT-3 (Brown et al. [2020]) model fine-tuned on the **768 Gold Standard** samples. The final set of **1200 Silver Standard** samples was created at the last moment, and due to time constraints, it was not possible to manually verify all of them. Nonetheless, the overall accuracy of GPT-3 was sufficiently good, so we decided to utilize them in our training.
- Table 11, shows that a total of four gold standard (GS) datasets were created. Specifically, two GS datasets were developed for the initial task of "Completing incomplete representations," one in English and the other in German. Additionally, two GS datasets were prepared for the more intricate second task, "Expanding condensed coordinated expressions into explicit paraphrases."

Dataset	Language	Samples	Train	Test
Noun Completion Task - de	German	510	400	110
Noun Completion Task - en	English	402	354	48
Phrase Expansion Task - de	German	1968	1773	195
Phrase Expansion Task - en	English	20	-	-

Table 11: Overview of datasets

- One important thing to note is that although the overall training samples contain more than 50% silver standard data, the final evaluation which was done on 195 data samples was human-verified Gold Standard.

Task:
Given a sentence or a piece of text containing 4 types of special tags: "<SoftSkill>", "</SoftSkill>", "</SoftSkill_C>" and "<SoftSkill_C>". The task is to complete the incomplete text between "<SoftSkill_C>" and "</SoftSkill_C>" by appending the necessary text from the text between nearby "<SoftSkill>" and "</SoftSkill>". If you change anything between "<SoftSkill_C>" and "</SoftSkill_C>", then replace "<SoftSkill_C>" with "<SoftSkill>" and replace "</SoftSkill_C>" with "</SoftSkill>". Do not change anything else in the sentence. If there is no change between "<SoftSkill_C>" and "</SoftSkill_C>" then do not change anything in the input text. The number of special tags in the Input text should be equal to the number of special tags in the Completed text.

Here are a few examples:

Example 1:

Input text: "<SoftSkill>Unternehmerisches Denken </SoftSkill> und <SoftSkillC>Handeln </SoftSkill_C>"
Completed text: "<SoftSkill>Unternehmerisches Denken </SoftSkill> und <SoftSkill>unternehmerisches Handeln </SoftSkill>"

Example 2:

Input text:"Eine <SoftSkill_C>belastbare</SoftSkill_C>, <SoftSkill_C>lösungsorientierte</SoftSkill_C> und <SoftSkill>teamfähige Persönlichkeit</SoftSkill>"
Completed text:"Eine <SoftSkill>belastbare Persönlichkeit</SoftSkill>, <SoftSkill>lösungsorientierte Persönlichkeit</SoftSkill> und <SoftSkill>teamfähige Persönlichkeit</SoftSkill>"

Example 3:

Input text:"der <SoftSkill>Freude an interessanten statistischen Arbeiten</SoftSkill> und an der <SoftSkill_C>Ausarbeitung von Analysen </SoftSkill_C> hat"
Completed text:"der <SoftSkill>Freude an interessanten statistischen Arbeiten</SoftSkill> und <SoftSkill>Freude an der Ausarbeitung von Analysen </SoftSkill> hat"

Example 4:

Input text:"Für die Region Zug/Innerschwyz suchen wir am Standort Schwyz eine <SoftSkill_C>organisations-</SoftSkill_C> und <SoftSkill>führungsgewandte Persönlichkeit</SoftSkill> als
Leiter/-in Regionalsekretariat 100%"
Completed text:"Für die Region Zug/Innerschwyz suchen wir am Standort Schwyz eine <SoftSkill>organisationsgewandte Persönlichkeit</SoftSkill> und <SoftSkill>führungsgewandte Persönlichkeit</SoftSkill> als
Leiter/-in Regionalsekretariat 100%"

Example 5:

Input text:"Als <SoftSkill_C>motivierende</SoftSkill_C> und <SoftSkill>begeisternde Persönlichkeit</SoftSkill> <SoftSkill>handeln Sie eigenverantwortlich</SoftSkill> und <SoftSkill_C>ergebnisorientiert</SoftSkill_C>."
Completed text:"Als <SoftSkill>motivierende Persönlichkeit</SoftSkill> und <SoftSkill>begeisternde Persönlichkeit</SoftSkill> <SoftSkill>handeln Sie eigenverantwortlich</SoftSkill> und <SoftSkill>handeln Sie ergebnisorientiert</SoftSkill>."

Now complete the following sentence

Input text: "<SoftSkill>begeisterungsfähig</SoftSkill> und <SoftSkill>flexibel</SoftSkill>
<SoftSkill>ehrgeizig</SoftSkill> und <SoftSkill>leistungsorientiert</SoftSkill> eine <SoftSkill_C>gewinnende</SoftSkill_C> und <SoftSkill>gepflegte Persönlichkeit</SoftSkill>"

Figure 6: Prompt for In-Context Learning

4 Methods and Architecture

4.1 Generative AI

Generative AI falls within the area of conventional machine learning and the machine learning models that drive generative AI have acquired their capabilities by finding statistical patterns in extensive datasets usually consisting of trillions of words that were initially composed by humans. Large Language Models (LLMs) have been trained on this massive amount of data which requires lots of computation power and several months of training. These models are often known as foundational models and have billions of parameters. Figure 7 shows the comparison of the size of various LLMs that have been released until March 2023 They usually possess properties that extend beyond linguistic capabilities. The more parameters a model has, the more sophisticated and complex tasks it can perform with higher accuracy. There have been several types of research going around this field wherein researchers are exploring the potential of LLMs to deconstruct intricate tasks, engage in reasoning and facilitate problem-solving.

The primary focus of this thesis is to explore the applications of LLMs (Large Language Models) in natural language generation. While Generative AI has the potential to handle diverse multi-modal tasks involving videos, images, audio, and speech, our research will specifically delve into the use of LLMs for decomposing complex coordinated sentences extracted from job ads data.

To accomplish this, we will investigate various LLMs and assess their suitability for our specific context. This evaluation will encompass both the original form of the LLMs and the application of fine-tuning techniques, which will be customized to cater to the requirements of our particular use case.

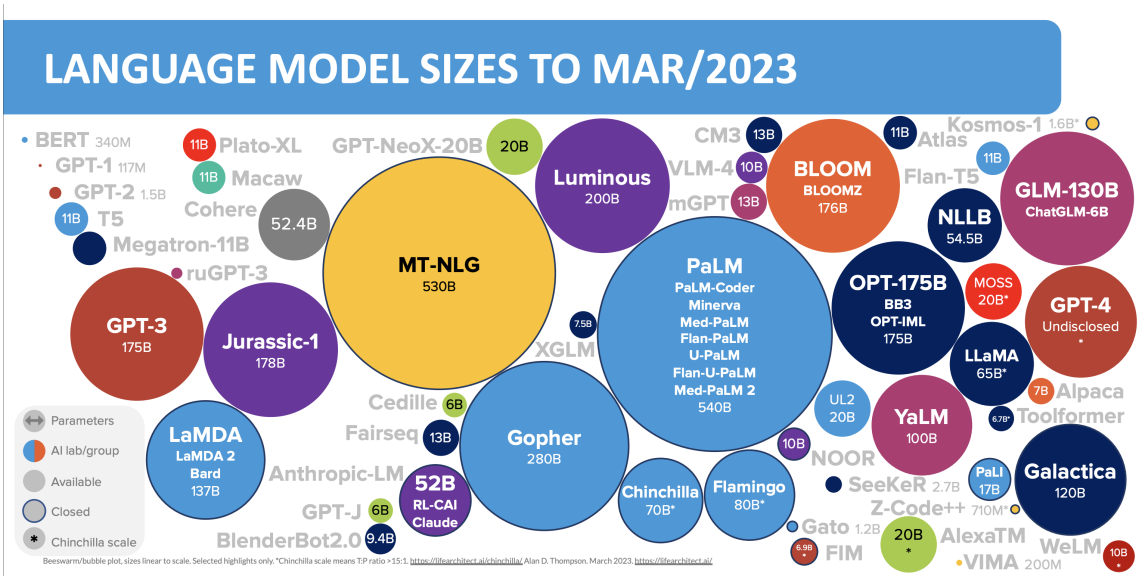

Figure 7: Size of the Language Models. Image Source: `https://lifearchitect.ai/models/#model-bubbles`

4.2 Prompting and Prompt Engineering

The process to input the text to a model for performing some kind of task is known as *'Prompting'*. While the act of generation of the text is called the *'Inference'*. The resulting text generated by the model is known as *'Completion'*. The available text or the memory that is available to use for providing the inference using the prompt is known as the *'Context window'*

It is very common that even after using several different combinations of prompts, the model does not demonstrate accurate results. In such situations, the language used in the prompt or the structure of the information provided in the prompt is revised repeatedly to get the desired results from the model. This iterative process of refining the prompt is known as *'prompt engineering'*

4.2.1 In-Context Learning (ICL)

There are various techniques used in Prompt Engineering to improve the model performance. One such technique is to provide more context and some examples of the specific task the model is intended to perform within the prompt itself. Hence, by providing such task-specific information within the context window, model performance can be improved and better results can be generated. This technique is

known as *"In-context learning"*.

For the examples in the below inferences, we have used GPT 3.5 model

4.2.1.1 Zero-Shot Inference

This is the type of inference where you provide specific information to the model on what needs to be done to successfully complete the task but no examples are provided in this case. Usually, the LLM with a large number of parameters does a good job for a variety of trivial tasks like sentiment analysis, translation, classification, etc. But in our case, as seen in Table 12, the model does not provide the expected result. This can be because of the German language and also because the task is not so trivial.

4.2.1.2 One-Shot Inference

In this type of inference, in addition to the Zero-Shot inference, one set of examples is also provided to the model. This helps the model to provide better results. In our case, as seen from Table 12, the model generates the exact result that we expected it to generate. However, for certain difficult tasks, providing just one example is not sufficient.

ICL	Prompt	Completion
Zero-Shot Inference	"Split the following German phrase containing elided words into simpler German words Haus- und Gartenarbeit"	The phrase "Haus- und Gartenarbeit" can be split into simpler German words as follows: Haus = house und = and Gartenarbeit = gardening work So the simplified words are Haus, und, Gartenarbeit."
One Shot Inference	"Split the following German phrase containing elided words into simpler German words Input : Vor- und Nachteile Output : Vorteile und Nachteile Input : Haus- und Gartenarbeit Output : "	Hausarbeit und Gartenarbeit

Table 12: Examples of Zero-Shot Inference and One-Shot Inference

4.2.1.3 Few-Shot Inference

The idea remains the same in this type of inference. But in addition to providing only one example, several examples are provided which helps the model to adapt to the learning and provide expected results. Here, in Table 13, we have used few-shot inference for a rather difficult task. But since we are using GPT 3.5 with 175 billion parameters, we see that the model generates the perfect result.

But later in our experiments, we will be using another technique called fine-tuning which involves training certain parameters of the model by providing several hundred input-output pairs. Fine-tuning is required because we will experiment with several different models which are not as large as GPT 3.5

ICL	Prompt	Completion
Few Shot Inference	Task: Given a sentence or a piece of text containing 4 types of special tags : "<SoftSkill>", "</SoftSkill>", "</SoftSkill_C>" and "<SoftSkill_C>". The task is to complete the incomplete text between "<SoftSkill_C>" and "</SoftSkill_C>" by appending the necessary text from the text between nearby "<SoftSkill>" and "</SoftSkill>". If you change anything between "<SoftSkill_C>" and "</SoftSkill_C>", then replace "<SoftSkill_C>" with "<SoftSkill>" and replace "</SoftSkill_C>" with "</SoftSkill>". Do not change anything else in the sentence. If there is no change between "<SoftSkill_C>" and "</SoftSkill_C>" then do not change anything in the input text. The number of special tags in the Input text should be equal to the number of special tags in the Completed text. Here are a few examples: Example 1: **Input text**: "<SoftSkill>Unternehmerisches Denken </SoftSkill> und <SoftSkill_C>Handeln </SoftSkill_C>" **Completed text**: "<SoftSkill>Unternehmerisches Denken </SoftSkill> und <SoftSkill>unternehmerisches Handeln </SoftSkill>" Example 2: **Input text**: "Sie <SoftSkill>packen gerne mit an</SoftSkill>, <SoftSkill>arbeiten zuverlässig</SoftSkill> und <SoftSkill_C>selbständig</SoftSkill_C>." **Completed text**: "Sie <SoftSkill>packen gerne mit an</SoftSkill>, <SoftSkill>arbeiten zuverlässig</SoftSkill> und <SoftSkill>arbeiten selbständig</SoftSkill>." Example 3: **Input text**: "Sie bezeichnen sich als <SoftSkill_C>flexible</SoftSkill_C> und <SoftSkill>teamfähige Person</SoftSkill>." **Completed text**: "Sie bezeichnen sich als <SoftSkill>flexible Person</SoftSkill> und <SoftSkill>teamfähige Person</SoftSkill>." Now complete the following sentence **Input text**: "Sie <SoftSkill>arbeiten speditiv</SoftSkill>, <SoftSkill_C>exakt</SoftSkill_C> und <SoftSkill_C>strukturiert</SoftSkill_C>." **Completed text**:	"Sie <SoftSkill>arbeiten speditiv</SoftSkill>, <SoftSkill>arbeiten exakt</SoftSkill> und <SoftSkill>arbeiten strukturiert</SoftSkill>."

Table 13: Examples of Few-Shot Inference

4.2.2 Limitations of In-Context Learning (ICL)

As demonstrated by the aforementioned cases, the use of ICL proves sufficient for achieving favorable outcomes in specific tasks. However, the effectiveness of this approach also relies on the size of the models employed. In the case of larger models, incorporating one or more examples in the prompt can often yield the desired results. Nonetheless, this technique does possess a few limitations:

1. When utilizing smaller models, even with the provision of 5-6 examples, this technique falls short of generating satisfactory outcomes.
2. Due to the inherent restriction on the size of the context window, there is a limit to the number of examples that can be added to attain the desired accuracy.

To address these challenges, an alternative technique known as "***Fine-Tuning***" is employed. In contrast to pre-training, where the majority of training occurs in a self-supervised manner, fine-tuning involves updating the weights in a supervised fashion. This approach utilizes labelled pairs of 'prompt' and 'completion' to enhance the model's decision-making capabilities for specific tasks.

A particular strategy called "***Instruction fine-tuning***" proves particularly beneficial in improving model accuracy across a variety of tasks. This method involves training a model on specific examples that demonstrate particular instructions, which the model should adhere to when generating results. For instance, in the case of text translation, a prefix such as "***Translate this sentence :***" is added. These examples enable the model to generate results that align with the given instructions.

4.2.3 Computation Challenges of Fine-Tuning LLMs

The fine-tuning technique mentioned above proves effective for models that possess a considerable number of parameters. However, the feasibility of fine-tuning larger models with an extensive parameter count on consumer hardware is hindered by memory limitations. Furthermore, if there are multiple models dedicated to distinct tasks, the storage of various variants becomes prohibitively expensive due to the fine-tuned models being comparable in size to the original models. To address these challenges, an approach known as "*Parameter-Efficient Fine-Tuning (PEFT)*" is employed, which aims to tackle both issues simultaneously.

4.3 Parameter Efficient Fine-Tuning (PEFT)

Contrary to full model fine-tuning, where all model weights are updated during supervised learning, Parameter-Efficient Fine-Tuning (PEFT) approaches involve fine-tuning only a small subset of model parameters, such as specific layers and components, while keeping the majority of trained parameters frozen. This approach significantly reduces the memory requirements for fine-tuning the model for downstream tasks and compresses the model, effectively addressing storage issues. As a result, PEFT techniques make memory requirements more manageable, enabling the training of very large models on single GPUs. Moreover, since the original Language Model (LLM) is minimally modified, there is no compromise in the quality of results, and the performance remains comparable to full fine-tuning.

There are three main classes of PEFT methods[1]:

1. **Selective**: These methods selectively fine-tune a subset of the original model parameters. Various approaches are employed to identify the parameters that should be updated. There are options to train specific layers, certain types of parameters, or particular components of the models. However, the performance of these methods elicits mixed responses, and hence, we will not be covering them in the experiments.
2. **Reparameterization**: These methods dramatically reduce the number of trainable parameters by creating new low-rank transformations of the original network weights. One commonly used technique in this class is LoRA by Hu et al. [2021], which will be discussed in the next section.
3. **Additive**: These methods keep all model weights frozen but introduce new trainable components.
    - Adapter methods: This approach adds new trainable layers to the model architecture, typically within the encoder or decoder components after the attention layers.
    - Soft Prompt methods: This approach maintains the frozen model architecture but focuses on modifying the input to improve results. One of the techniques is known as *"Prompt Tuning"* and is explored in great detail in the paper by Lester et al. [2021]

[1]Classes of PEFT methods: `https://www.deeplearning.ai/`

4.3.1 LoRA (Hu et al. [2021]): Low-Rank Adaptation of Large Language Models

In this paper by Hu et al. [2021], the authors present an approach to fine-tuning by finding an efficient way to update the weights of the model without having to train every single parameter again.

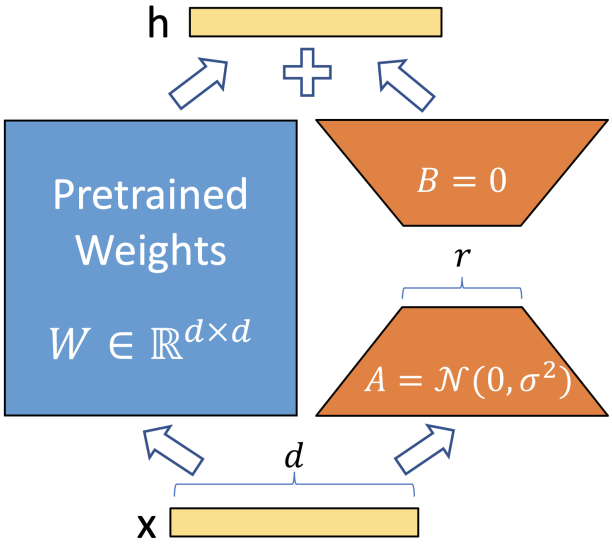

Figure 8: Reparameterisation in LoRA. Image Source: Hu et al. [2021]

It achieves this by initially freezing all of the original model parameters and then introducing a pair of rank decomposition matrices (A and B in Figure 8) alongside the original weights. These smaller matrices are carefully dimensioned so that their product results in a matrix with the same dimensions as the weights they are modifying.

During the fine-tuning process, the original weights of the Language Model (LLM) remain frozen, while the smaller matrices(A and B in Figure 8) are trained using the same supervised learning approach discussed earlier. In the inference stage, these two low-rank matrices are multiplied together to generate a matrix with dimensions equivalent to the frozen weights. This resulting matrix is then added to the original weights, effectively replacing them in the model with the updated values.

By utilizing the LoRA (Hu et al. [2021]) fine-tuning technique, a model can be adapted for a specific task, allowing it to perform the desired functionality while significantly reducing the number of parameters that need to be trained.

4.4 Generative Configuration

Let's explore the various configuration parameters that can affect the model's generation of the final output in terms of next-word prediction. In our upcoming experiments, we will be specifically investigating these configurations to attain the desired outcomes. It's important to note that these configurations are distinct from the parameters learned during training. Instead, they solely impact the output during the inference phase, allowing us to control factors such as the maximum length of output tokens and the level of creativity in the generated text.

4.4.1 Max New Tokens

This configuration parameter determines the maximum number of new tokens allowed in the generated output. By setting an appropriate value, we can limit the length of the generated text and prevent excessively long or verbose outputs.

4.4.2 Greedy Decoding VS Random Sampling

When the model is given a set of words and the decoder predicts the next word in the sequence, a probability distribution is created over the entire vocabulary of words in the output layer, and the softmax function is applied to calculate the probabilities. By default, many Language Models (LLMs) use *"Greedy Decoding"*, where the word with the highest probability is chosen as the output. This method is simple and often yields good results for generating short sequences of text.

However, using Greedy Decoding can result in the generation of repetitive text. To introduce some randomness and avoid repetition, an alternative approach called *"Random (-weighted) Sampling"* can be used. Instead of selecting the word with the highest probability, the output word is randomly chosen based on the probability distribution.

Both Greedy Decoding and Random Sampling have their drawbacks. Greedy Decoding can lead to repetitive sequences, while Random Sampling may generate completely random and incoherent sequences. To address these issues, the following configurations are used to control this setting which would ensure that the output is more sensible.

4.4.3 Sample Top K

The Top K sampling approach involves selecting the K most probable tokens at each generation step. By adjusting the value of K, we can control the uniqueness and diversity of the generated text. A smaller K limits the options to a few highly probable tokens, leading to repetitive outputs, while a larger K allows for more varied and diverse results.

4.4.4 Sample Top P

It is similar to Sample Top K. It involves sampling from the top tokens until the cumulative probability exceeds a certain threshold P. By increasing the value of P, we include a larger portion of the probability mass, resulting in more creative and diverse outputs.

4.4.5 Temperature

The temperature parameter also affects the randomness or diversity of the generated text. The higher value of temperature makes the probability distribution curve more flat and hence it will include more words in the distribution which would allow the model to sample the words from the larger distribution thereby making more creative predictions. On the other hand, the lower value of temperature will allow the distribution to peak around the center which would consider only a small set of words to sample from, thereby making more deterministic and focussed predictions

4.5 Error Metrics

To test the performance of our fine-tuned Language Models (LLMs), we have adopted various error metrics and comparisons. The evaluation process is divided into three distinct categories:

1. **C Skills Evaluation**: In this category, we extract soft skills that were originally incomplete and identified between *<SoftSkill_C>* and </SoftSkill_C> tags. The objective is to evaluate how effectively the incomplete skills were completed while disregarding other modifications in the text.

2. **All Skills Evaluation**: This category involves extracting all soft skills, whether they were originally complete (inside <SoftSkill> tags) or incomplete (inside

<SoftSkill_C> tags). The goal is to evaluate all skills without considering anything outside the <SoftSkill> and <SoftSkill_C> tags.

3. **Complete Sentences Evaluation**: In this category, all special tags such as <SoftSkill_C>, </SoftSkill_C>, <SoftSkill>, and </SoftSkill> are removed, and the entire sentences of the generated text are compared with the gold standard text.

For both the *C Skills* and *All Skills* evaluations, the same error metrics are used since both categories focus on evaluating the quality of individual soft skills. The following four error metrics will be employed:

For calculating all 4 metrics, there is a common data preparation step required. We obtain all the necessary soft skills (either All or only incomplete ones) from the sentence, and we store them in a list called the generated list. We also have another list called the reference list which contains the same set of soft skills from the Gold Standard dataset.

1. **Rouge[2]-L Score** - Once we have both the generated and reference lists, we do the one-on-one comparison of each soft skill and find the Rouge-L scores for each pair and then take the average of Rouge-L scores across all the soft skills for one sentence. Rouge-L measures the longest common subsequence between the generated text and the reference text, which provides an indication of how well the generated soft skills align with the reference soft skills in terms of common words and phrases. A higher Rouge-L score indicates a better similarity and alignment between the generated text and the reference text, suggesting better performance of the language model in generating relevant and coherent soft skills. The maximum value of 1 is achieved if two texts are identical.

2. **Levenshtein Distance[3]** - To assess the quality of the generated soft skills, we calculate the Levenshtein distance for each pair of soft skills in the generated list and the reference list. The Levenshtein distance quantifies the minimum number of single-character edits (insertions, deletions, or substitutions) required to transform one string into another. In our context, it measures the dissimilarity between the generated soft skills and the corresponding reference soft skills in terms of the minimum number of edit operations needed to align the two sequences. By summing over all the pairs, we obtain the final Levenshtein distance value, where a lower value indicates a higher degree of

[2]ROUGE: `https://en.wikipedia.org/wiki/ROUGE_(metric)`

[3]Levenshtein Distance: `https://en.wikipedia.org/wiki/Levenshtein_distance`

similarity and alignment between the generated soft skills and the reference soft skills, signifying better performance of the language model in accurately producing the desired soft skills.

3. **% Skills** - The % Skills metric evaluates the Percentage of soft skills that were an exact match between the generated list of soft skills and the reference list of soft skills. For each pair, it checks if there is an exact match between the soft skills in the generated list and the reference list. The metric then computes the number of matched soft skills and the total number of soft skills present in the reference list. Across all testing samples, it calculates the sum of the number of matched soft skills and divides it by the sum of the total number of soft skills present in the reference lists. This provides a percentage value indicating the proportion of soft skills that were accurately matched by the language model out of the total soft skills present in the reference lists, thereby quantifying the model's performance in generating correct soft skills.

4. **Cosine Similarity**[4] - Cosine similarity measures the similarity between the vectors representing the generated list of soft skills and the reference list of soft skills. We calculate cosine similarity for each pair of soft skills in the generated and reference lists and then take the average score. This metric quantifies how closely the generated soft skills align with the reference soft skills, providing an overall evaluation of the model's effectiveness in reformulating soft skills.

Lastly, for the *Complete Sentences Evaluation*, the error metrics, including **Rouge-L score**, **Levenshtein distance**, and **Cosine Similarity**, remain the same as explained before. The only difference is in the calculation method. In this case, we directly compare entire sentences for each metric and then take the average of these metrics across all samples in the test set to obtain the final evaluation.

[4]Cosine Similarity: `https://en.wikipedia.org/wiki/Cosine_similarity`

5 Experiment Pipeline

5.1 Data Extraction, Sampling, and Storage

The generation of four Gold Standard datasets involved data extraction for two tasks, one in German and one in English for both the *Noun Completion Task* and *Phrase Expansion Task*.

For the Noun Completion Task in German, data extraction utilized an automated data pipeline that employed *Regex* patterns to specifically search for the pattern *"NOUN-HYPHEN-AND-NOUN"* (e.g., "Haus- und Gartenarbeit"), as explained in Section 3.1.1. For the English dataset, the German dataset was translated using deepL, as detailed in Section 3.1.2.

For the *Phrase Expansion Task* in German, data extraction was accomplished through a spaCy pipeline comprising a domain-specific spaCy-based soft skill requirements text span recognizer and a paragrapher inserting sentence boundaries after `"\n\n"` tokens. This pipeline processed the original job ads into the desired format like "*<SoftSkill_C>Kreatives</SoftSkill_C> und <SoftSkill> handwerkliches Geschick </SoftSkill>*" necessary for fine-tuning. Similar to the *Noun Completion Task*, data extraction for the *Phrase Expansion Task* in English was carried out using deepL, as discussed in Section 3.2.1.

Both the *Noun Completion Task* and *Phrase Expansion Task* datasets in German underwent a final sampling step using the *Nilsimsa sampler* to select unique and diverse cases from the extracted data, as elaborated in Sections 3.1.3 and 3.2.4 for the *Noun Completion Task* and *Phrase Expansion Task*, respectively.

The automated pipelines were configured to extract datasets in German for the two tasks. The Python script allowed users to select the language and number of samples to extract from each JSONL data file. Only German cases from all available files across all years were extracted and saved as CSV[1] files. These CSV files served as input sources for data sampling, and the resulting sampled data was also stored

[1]CSV: `https://en.wikipedia.org/wiki/Comma-separated_values`

as CSV files. The CSV files were then manually verified and used as sources for fine-tuning the models.

5.2 Programming Methodology for Fine-tuning Scripts

Python[2] serves as the primary programming language for implementing all fine-tuning pipelines. We have developed two separate Python scripts: one for standard fine-tuning of models without utilizing any Parameter Efficient Fine-Tuning (PEFT) techniques, and the other pipeline incorporates a PEFT technique called LoRA. Each of these scripts is designed as a modular and configurable system, accepting various arguments such as model checkpoint, learning rate, dataset path, weight decay rate, batch sizes, and others.

To streamline experimentation and manage multiple fine-tuning runs with different hyperparameter combinations, we employ a Bash[3] file. This bash file allows us to input different configurations and execute the Python script iteratively. By doing so, we can easily track and manage various experiments that run simultaneously, thus facilitating better code maintenance and organization.

5.3 NLP Ecosystem and Experiment Tracking

For fine-tuning, we leveraged models available in the Hugging Face model hub, importing various checkpoints from there. These pre-trained models played a crucial role in our experiments.

In addition to the Hugging Face[4] resources, GPT-3 from OpenAI[5] proved to be a valuable asset during dataset preparation in our dataset bootstrapping pipeline.

To effectively track and log various metrics during training, we adopted Weight&Biases[6] (W&B) as our logging strategy. W&B allowed us to record and monitor evaluation metrics, training progress, and accuracy curves. These logged data were instrumental in analyzing the performance of different models in later stages.

As a final step, the best-performing models were shared with the community by

[2]Python: `https://www.python.org/`

[3]Bash: `https://www.gnu.org/software/bash/`

[4]Hugging Face: `https://huggingface.co/`

[5]OpenAI: `https://openai.com/`

[6]Weight&Biases: `https://wandb.ai/site`

exporting them to our Hugging Face hub account, under the username *Kartikey95*[7]. By doing so, other users can conveniently access and utilize these models by simply importing them from Hugging Face, making their integration into new projects seamless and accessible to a wider audience.

5.4 Inference on Fine-Tuned Model

After obtaining the fine-tuned models, we establish an additional pipeline that imports these models along with their corresponding tokenizers to perform inferences. The inference step for the *Phrase Expansion Task* involves running the models on new data to generate predictions. Once the inferences are completed, we execute a post-processing step to extract all the soft skills from the results.

The extracted soft skills are then compared against the gold standard dataset to perform a final evaluation. This evaluation allows us to assess the performance and effectiveness of the fine-tuned models in expanding soft skills from the input text.

In conclusion, the described pipeline for fine-tuning is instrumental in achieving the goal of extracting expanded soft skills from text data.

[7]Hugging Face Account: `https://huggingface.co/Kartikey95`

6 Results and Discussion

In Chapter 4, two distinct tasks were identified, each requiring the preparation of gold-standard (GS) datasets and subsequent model training. As shown in Table 14, a total of four gold standard (GS) datasets were created. Specifically, two GS datasets were developed for the first task of the "*Noun Completion Task*," one in English and the other in German. Additionally, two GS datasets were prepared for the more intricate second task, "*Phrase Expansion Task*"

Due to time restrictions, we could not add more samples to the English dataset for "*Phrase Expansion Task*". Hence, this dataset is not used for fine-tuning the pre-trained LLMs

Dataset	Language	Samples	Train	Test
Noun Completion Task - de	German	510	400	110
Noun Completion Task - en	English	402	354	48
Phrase Expansion Task - de	German	1968	1773	195
Phrase Expansion Task - en	English	20	-	-

Table 14: Overview of datasets

6.1 Evaluation of Models for Noun Completion Task

The primary objective of this task is to determine the most suitable pretrained models for performing the *Phrase Expansion Task*

To accomplish this, various models based on "*Sequence-to-Sequence Models*" and "*Autoregressive*" architectures were fine-tuned and tailored to address our specific tasks.

6.1.1 German dataset for *Noun Completion Task*

The German dataset containing Noun Groups with truncated words was fine-tuned using a total of seven models, two of which are not publicly and freely available: *Ada*[1] and *Davinci*[2], both being variants of the GPT-3 (Brown et al. [2020]).

Figure 10 plot the Evaluation Loss during training, with log scales utilized for both the x-axis and y-axis to enhance the graph's interpretability.

Upon analyzing Figure 10, it is evident that the ***mt5-base***[3] model exhibits a worse loss curve compared to all other models, a conclusion further supported by the results in Table 15. Conversely, the ***Flan-T5 large***[4] model achieves the lowest loss on the evaluation set, which aligns with its position as the best-performing model according to Table 15. For assessing the models' performance during training, RougeL scores were calculated and can be viewed in Figure 9. These scores corroborate our earlier findings, with ***Flan-T5 large*** attaining the maximum RougeL score of 66.0 at a specific training step, while simultaneously reaching a minimum score of only 50.5.

It should be noted that the Evaluation Loss progression displayed in Figure 10 does not include information on the GPT-3 model variants, as their fine-tuning is conducted through OpenAI's API

Upon reviewing Table 15, it becomes apparent that the open-source models from the T5 family outperform the GPT-3 variants by a notable margin. Based on these results, the open-source variants appear to be more promising for this particular task than the GPT-3 models.

[1] Ada: `https://platform.openai.com/docs/models/gpt-3`

[2] Davinci: `https://platform.openai.com/docs/models/gpt-3`

[3] mt5-base: `https://huggingface.co/google/mt5-base`

[4] Flan-T5 large: `https://huggingface.co/google/flan-t5-large`

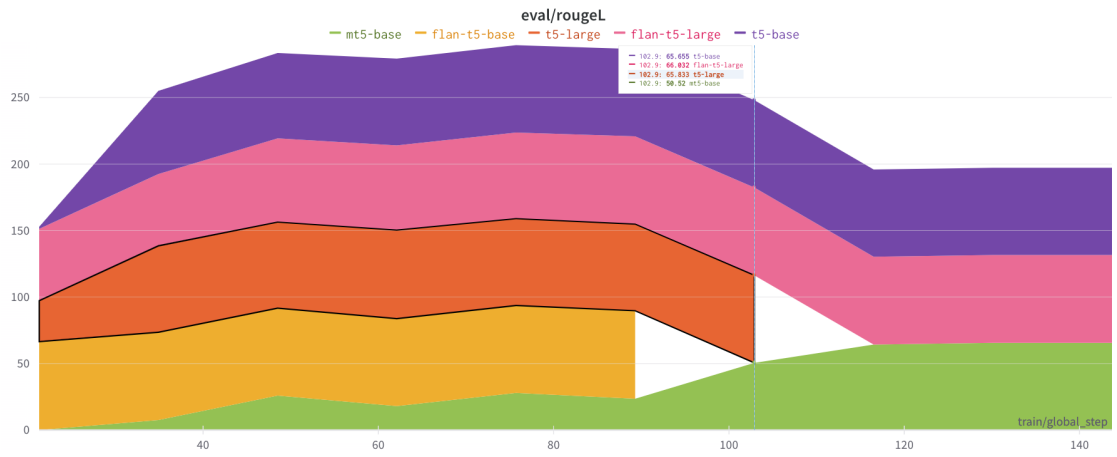

Figure 9: Rouge Scores for various models for *Noun Completion Task* on German dataset. It is the representation of the area plot where at any given step, the Rouge score is the vertical length in each colour space representing each model

Model with LR	Near Match	Exact Match	Average Levenshtein
Flan-T5 large[5]	0.95	0.91	0.37
T5 large[6]	0.93	0.91	0.44
t5-base[7]	0.93	0.89	0.68
GPT-3 DaVinci	0.92	0.87	0.70
Flan-T5 base[8]	0.87	0.85	0.76
mt5-base[9]	0.85	0.78	1.23
GPT-3 ada	0.77	0.72	1.52

Table 15: **Model results for *Noun Completion Task* for German dataset** Exact Match refers to the % of total cases where the generated text was identical to the expected text. Near Match refers to the % of total cases where the Levenshtein distance between the generated text and expected text is less than 3. The results presented in the table have been arranged in ascending order based on the *Average Levenshtein* scores.

6.1.2 English dataset for *Noun Completion Task*

Similar to the task described in subsection 6.1.1, this task involves the English dataset. As with the previous task, various models were fine-tuned for this specific

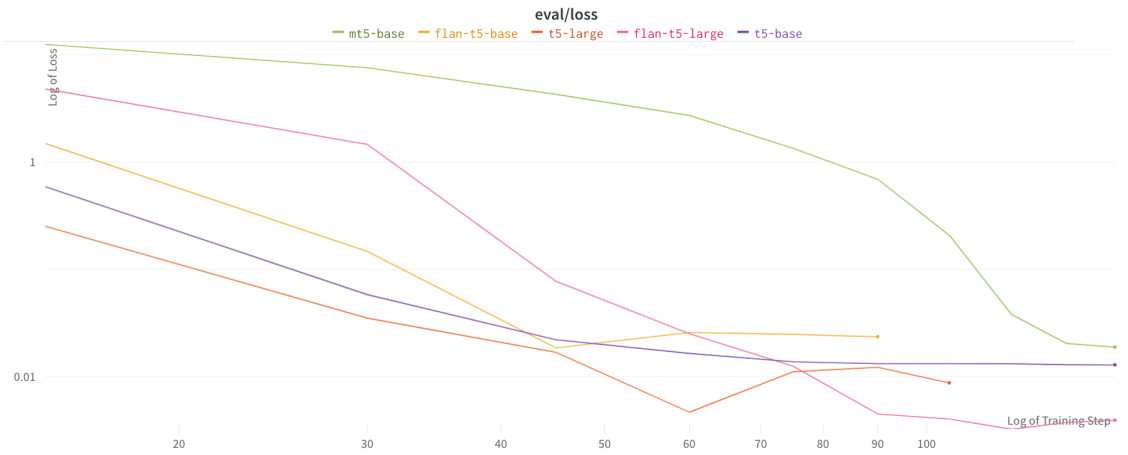

Figure 10: Log of Evaluation Loss progression for various models for *Noun Completion Task* on German dataset

objective, but due to the superior performance of open-sourced models over GPT-3 variants, we excluded the fine-tuning of GPT-3 variants for this task.

As shown in Table 16, the ***Flan-T5 large*** model achieves 100% accuracy on the *Near Match* metric with very low *Average Levenshtein* score. The results for the English task are notably better compared to the German task, which is consistent with the fact that these models are predominantly pre-trained in the English language. Interestingly, the ***mT5-base***, a multilingual variant of the T5 model, performs substantially worse for the English task compared to the German task, and the results from Table 16 indicate that the model failed to learn effectively.

Figure 12 illustrates the progression of Evaluation Loss during training, utilizing log scales for both the x-axis and y-axis for enhanced interpretation. Since ***mT5-base*** did not demonstrate any improvements during fine-tuning, we excluded its learning curve from the graph. Apart from this, all other models consistently reach very low values of validation loss, as further confirmed by the results in Table 16. Similarly, Figure 11 presents the RougeL scores of all the models during training, and no significant difference can be observed from this figure. This suggests that all these models have comparable accuracies.

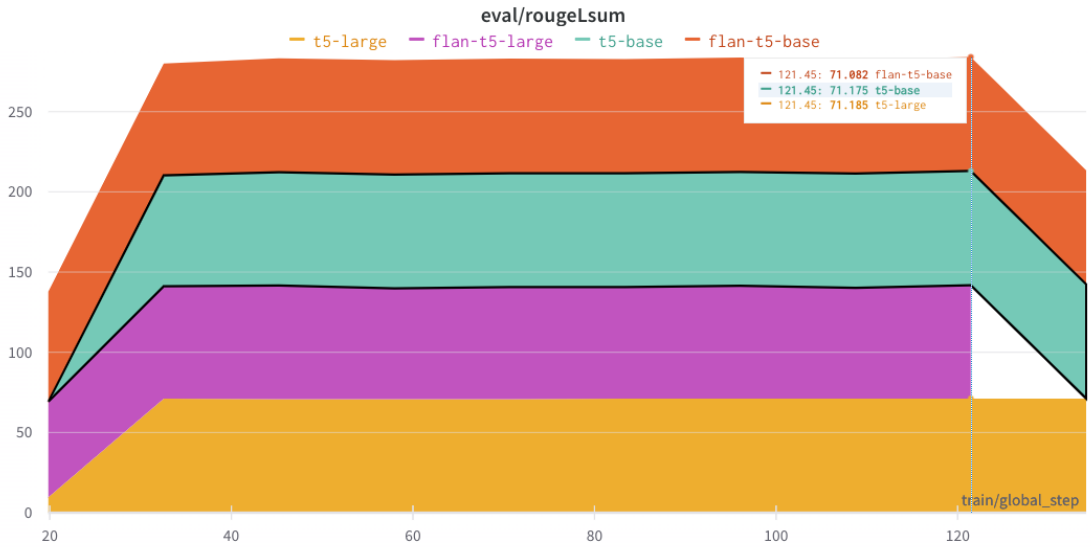

Figure 11: Rouge Scores for various models for *Noun Completion Task* on English dataset. It is the representation of the area plot where at any given step, the Rouge score is the vertical length in each colour space representing each model

Model with LR	Near Match	Exact Match	Average Levenshtein
flan-t5-large	1.0	0.98	0.02
t5-large	0.98	0.96	0.13
flan-t5-base	0.96	0.94	0.29
t5-base	0.94	0.92	0.46
mt5-base	0.00	0.00	38.18

Table 16: **Model results for *Noun Completion Task* for English dataset** Exact Match refers to the % of total cases where the generated text was identical to the expected text. Near Match refers to the % of total cases where the Levenshtein distance between the generated text and expected text is less than 3. The results presented in the table have been arranged in ascending order based on the *Average Levenshtein* scores.

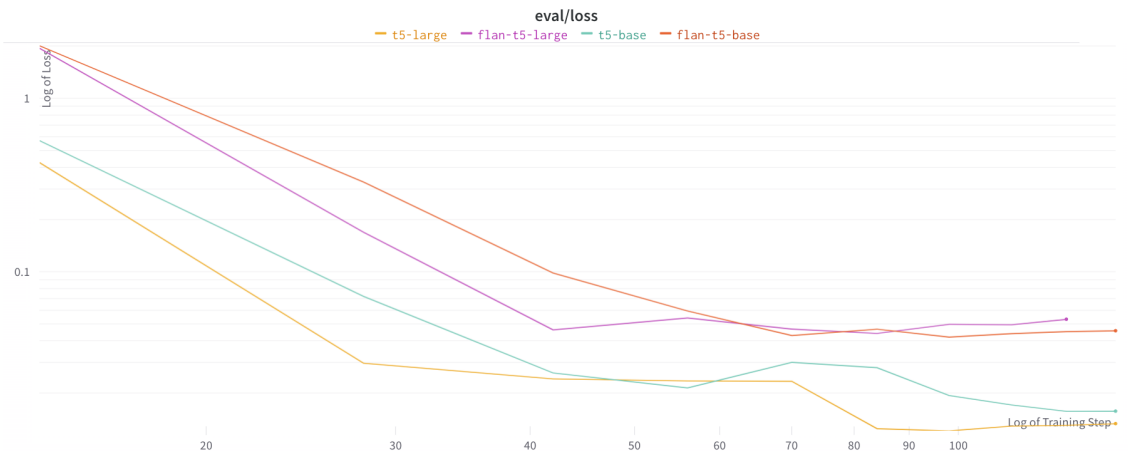

Figure 12: Log of Evaluation Loss progression for various models for *Noun Completion Task* on English dataset

6.2 Evaluation of Models for *Phrase Expansion Task*

6.2.1 In-Context Learning

In order to address the *Phrase Expansion Task*, we initially explored the In-Context Learning approach. We experimented with various models, including DaVinci from GPT-3 and different variants of T5[10] and FLAN T5[11]. Unfortunately, these models yielded unsatisfactory results, and thus, we omit their outcomes from this discussion. The complexity of the task, involving eight problem types and numerous special tags within the sentences, posed challenges for these models.

Subsequently, our attention turned to chatGPT as it has demonstrated exceptional performance on instruction-based tasks. Through multiple iterations of prompt engineering, we arrived at a suitable prompt that offers a comprehensive explanation of the task, complemented by examples of various problem types along with their corresponding accepted solutions.

The prompt that we ultimately selected is shown in the Figure 6

The selected prompt produced the expected text generation when applied to 35 samples. The evaluation was conducted across three distinct categories, each assessing different aspects of text generation quality:

[10]T5: `https://huggingface.co/docs/transformers/model_doc/t5`

[11]FLAN-T5: `https://huggingface.co/docs/transformers/model_doc/flan-t5`

1. **ALL Skills**: This category focuses on all skills, whether already complete or incomplete. The results in Table 17 indicate impressive performance, with a *"% Skills"* score of 91.59% which means that 91.59% of all skills mentioned were correctly generated by chatGPT.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
0.98	1.25	0.97	0.99	HOWA	8
0.95	8.60	0.90	0.99	MC	5
1.00	0.00	1.00	1.00	MDC	1
0.81	22.75	0.75	0.96	MWA	4
1.00	0.00	1.00	1.00	NONE	2
0.97	1.20	0.91	1.00	OWA	15
0.95	4.63	0.92	0.99	All Samples	35

Table 17: Results of **chatGPT** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

2. **C Skills**: In this category, we specifically evaluate the completion of skills. Since this evaluation is more constrained, the metrics in Table 18 may be slightly lower than those in Table 17. Nevertheless, chatGPT performed well across all problem types.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HOWA	8
0.91	8.60	0.82	0.98	MC	5
1.00	0.00	1.00	1.00	MDC	1
0.67	17.25	0.60	0.91	MWA	4
1.00	0.00	1.00	1.00	NONE	2
0.84	0.87	0.87	0.99	OWA	15
0.88	3.57	0.86	0.98	All Samples	35

Table 18: Results of **chatGPT** on *"C skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

3. **Complete Sentences**: This category compares complete sentences word by word. The results in Table 19 show an overall *Levenshtein* distance of 4.94, with favourable scores for most problem types, except for MC and MWA.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
1.00	1.25	1.00	HOWA	8
1.00	8.00	1.00	MC	5
1.00	0.00	1.00	MDC	1
1.00	22.75	0.97	MWA	4
1.00	0.00	1.00	NONE	2
0.99	2.13	1.00	OWA	15
1.00	4.94	1.00	All Samples	35

Table 19: Results of **chatGPT** on *"Complete Sentences"* Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

Based on the results from these three categories, the following conclusions can be drawn:

1. The obtained results are decent, providing a strong pre-annotation for building the Gold Standard for this particular task.

2. Language Models (LLMs) demonstrate their capability to perform this task effectively. With a broader variety of cases and samples, open-source LLMs can likely achieve even higher accuracy in the same task.

6.2.2 Fine-Tuning

After generating a sufficient amount of data through In-Context Learning using chatGPT, the focus shifted towards the fine-tuning approach. To achieve decent results using the fine-tuning method with GPT-3 (Brown et al. [2020]), a dataset of at least 100 samples was required. Below this threshold, the accuracy was found to be average and the model performed poorly on certain tasks. To address this, dataset bootstrapping was employed, involving the generation of samples using GPT-3's Davinci and chatGPT models, which were then manually verified to create a Gold Standard (GS) dataset.

The bootstrapping process was carried out iteratively. Initially, GPT-3 (Brown et al. [2020]) was fine-tuned using data from the first iteration, and then the generated samples were manually verified to create the next iteration of the GS dataset. As can be seen from Table 10, this process continued iteratively until a total of 1968 samples were generated. However, it is essential to mention that due to the pre-annotation with GPT-3, there might be a positive bias in the generated samples.

For evaluation purposes, only the final fine-tuned version of GPT-3's Davinci was considered. It is important to note that the training set for GPT-3 and other models was not identical. While the dataset for other models was shuffled before the split, GPT-3 was used for iterative fine-tuning and generating additional samples. As a result, the final testing of the model was conducted on a different, relatively larger test set compared to the test set used for other models.

6.2.2.1 All types of problems

We conducted a detailed evaluation of various models for different types of problems on the German dataset. The following three categories were analyzed:

1. **All Skills** : Table 20 presents the comparison of accuracy metrics for different models. All soft skills were extracted from the generated text and the expected text and then compared. GPT-3 achieved the highest scores in all metrics.

However, BLOOM[12] (Scao et al. [2022]), mT5-XL[13] (Xue et al. [2021]), and FLAN-T5-XXL[14] (Chung et al. [2022]) exhibited comparable accuracy, with all three models correctly extracting approximately 90% of all mentioned skills in the job ad span.

Rouge-L Score	Avg Levenshtein	% Skills	Cosine Similarity	Model Name
0.98	1.42	0.97	1.00	**GPT-3**
0.96	2.18	0.93	1.00	mT5-XL
0.96	3.27	0.91	1.00	FLAN-T5-XXL
0.95	3.01	0.90	0.99	BLOOM
0.88	9.39	0.75	0.98	T5Large

Table 20: Results of all models for all types of problems on *"ALL skills"* Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending order based on the *Avg Levenshtein* distances

2. **C Skills**: The Table 21 focuses on skills that needed to be completed by understanding the context. GPT-3 again performed exceptionally well, topping the chart in all metrics. However, BLOOM (Scao et al. [2022]) and FLAN-T5-XXL (Chung et al. [2022]) did not achieve comparable performances compared to GPT-3 (Brown et al. [2020]), while mT5-XL (Xue et al. [2021]) showed similar metric scores to GPT-3.

[12]BLOOM: `https://huggingface.co/bigscience/bloom-7b1`

[13]mT5-XL: `https://huggingface.co/google/mt5-xl`

[14]FLAN-T5-XXL: `https://huggingface.co/google/flan-t5-xxl`

Rouge-L Score	Avg Levenshtein	% Skills	Cosine Similarity	Model Name
0.94	1.10	0.93	1.00	**GPT-3**
0.88	1.80	0.85	0.99	mT5-XL
0.90	2.75	0.81	0.99	FLAN-T5-XXL
0.92	4.10	0.80	0.99	BLOOM
0.65	8.84	0.54	0.97	T5Large

Table 21: Results of all models for all types of Problems on *"C skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending order based on the *Avg Levenshtein* distances

3. **Complete Sentence**: The Table 22 presents the comparison of various accuracy metrics for different models when evaluating entire sentences. As expected, GPT-3 excelled in all metrics. On the other hand, BLOOM and T5Large performed poorly and were among the worst-performing models for this task. In this specific case, the performance gap between GPT-3 and the next two widened substantially.

Rouge-L Score	Avg Levenshtein	Cosine Similarity	Model Name
0.99	**1.76**	**1.00**	**GPT-3**
0.98	4.42	0.99	FLAN-T5-XXL
0.98	5.13	0.99	mT5-XL
0.96	14.82	0.98	BLOOM
0.96	15.52	0.98	T5Large

Table 22: Results of all models for all types of Problems on Complete Sentences. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending order based on the *Avg Levenshtein* distances

In conclusion, the evaluation of different language models on three categories of problems - *All Skills*, *C Skills*, and *Complete Sentences* - revealed the following:

1. GPT-3 exhibited superior performance compared to the rest of the models and

by a significant margin. It performed exceptionally well across all problem types

2. mT5-XL performed very well overall and came closest to GPT-3 in terms of performance.

3. The performance gap between GPT-3 and the next two models (mT5-XL and FLAN-T5-XXL) continued to widen substantially as we moved from *C skills* to *All Skills* and then to *Complete Sentence*. This suggests that these models have not reformulated the texts between *<SoftSkill_C>* and *</SoftSkill_C>* should be changed, while everything else should remain constant.

4. BLOOM performed worse than GPT-3, mT5-XL, and Flan-T5-XXL, in the category of *Complete Sentences*. This suggests that the model substantially altered the text outside the soft skill tags, affecting overall sentence quality.

5. T5-Large consistently performed poorly compared to other models and was the worst-performing model overall.

6. While the comparison may not be entirely equitable as GPT-3 was evaluated on a distinct test set, it was still included as a benchmark. The reason for not testing GPT-3 on the same dataset as other LLMs was the cost associated with generating inferences. Despite this difference, mT5-XL exhibited remarkable performance, comparable to that of GPT-3.

6.2.2.2 Evaluation on *ALL Skills*

Here we do a more comprehensive comparison of several performance metrics for five fine-tuned LLMs on the German dataset. In this section, we will be looking at how different LLMs solve the *All Skills* category of problems.

1. **GPT-3**: Table 23 showcases the results for GPT-3. The model demonstrated exceptional performance by successfully generating 97% of all the soft skills across all different types of problems correctly in the job ad span, which is quite remarkable. This includes both expanding the incomplete soft skills and rewriting the already complete soft skills. Notably, the model excelled in solving the "OWA" and "HOWA" types of problems, achieving accuracy rates of 98% and 99%. These types involve simple tasks like adding one word or completing a hyphenated word, making them easier for the model.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
0.92	0.00	0.94	0.99	HMC	8
1.00	0.00	1.00	1.00	HMDC	5
1.00	0.17	0.99	1.00	HOWA	30
0.99	0.32	0.98	1.00	MC	50
0.94	3.83	0.92	0.99	MDC	30
0.97	2.98	0.96	1.00	MWA	66
1.00	0.00	1.00	1.00	NONE	3
0.99	0.59	0.98	1.00	OWA	98
0.98	1.42	0.97	1.00	All Samples	292

Table 23: Results of **GPT-3** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

However, the most astonishing result is the model's performance on the "HMDC" types of problems. These are the most challenging problem types, requiring the completion of hyphenated text by intelligently considering the surrounding context and completing the soft skills involving the addition of several single/-multiple words across the span. Despite the complexity, the model achieved 100% accuracy for these types of problems indicating that the text generation was precisely identical to what was expected. However, it's important to note that this highscore may be influenced by the small number of samples on which it was tested, and thus the result may not be perfectly reliable.

One of the examples of the GPT-3 text generation for "HMDC" type of problem is:

Input : Abgeschlossene mechanische Grundausbildung\nMehrjährige Berufserfahrung im mechanischen Bereich, vorzugsweise in einem Produktionsbetrieb \n<SoftSkill>Schichtbereitschaft</SoftSkill> \n<SoftSkill_C>*Durchsetzungs-*</SoftSkill_C> und <SoftSkill>organisationsstarke Persönlichkeit</SoftSkill> \n<SoftSkill_C>*Flexible*</SoftSkill_C>, <SoftSkill_C>*zuverlässige* </SoftSkill_C> und <SoftSkill>selbständige Arbeitsweise</SoftSkill>

Expected output : Abgeschlossene mechanische Grundausbildung\n Mehrjährige Berufserfahrung im mechanischen Bereich, vorzugsweise in einem Produktions-

betrieb\n<SoftSkill>Schichtbereitschaft</SoftSkill>\n<SoftSkill> *Durchsetzungsstarke Persönlichkeit*</SoftSkill> und <SoftSkill>organisationsstarke Persönlichkeit </SoftSkill> \n<SoftSkill>*Flexible Arbeitsweise* </SoftSkill>, <SoftSkill>*zuverlässige Arbeitsweise*</SoftSkill> und <SoftSkill>selbständige Arbeitsweise</SoftSkill>

GPT-3 generated text : Abgeschlossene mechanische Grundausbildung\n Mehrjährige Berufserfahrung im mechanischen Bereich, vorzugsweise in einem Produktionsbetrieb\n<SoftSkill>Schichtbereitschaft</SoftSkill>\n<SoftSkill> *Durchsetzungsstarke Persönlichkeit*</SoftSkill> und <SoftSkill> organisationsstarke Persönlichkeit </SoftSkill> \n<SoftSkill>*Flexible Arbeitsweise*</SoftSkill>, <SoftSkill>*zuverlässige Arbeitsweise*</SoftSkill> und <SoftSkill>selbständige Arbeitsweise</SoftSkill>

2. **BLOOM**: Table 24 presents the results for the BLOOM model on the *"ALL skills"* category. The findings reveal that it performs best on *OWA* types of problems, which is expected due to their lower complexity. The model's poor performance on *HMDC* types of problems is understandable, as these are the most challenging ones to solve. Notably, the model demonstrates learning capabilities in dealing with *NONE* types of problems by refraining from making unnecessary changes to the text.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
0.78	19.00	0.33	0.97	HMC	1
0.86	7.20	0.75	0.98	HMDC	10
0.91	2.63	0.87	1.00	HOWA	19
0.95	4.03	0.91	0.99	MC	34
0.94	3.64	0.91	0.99	MDC	11
0.96	3.30	0.91	1.00	MWA	53
1.00	0.00	1.00	1.00	NONE	3
0.97	1.49	0.93	1.00	OWA	63
0.95	3.01	0.90	0.99	All Samples	195

Table 24: Results of **BLOOM** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

Moreover, the average Levenshtein score of 3.01 across all problem types for

all the soft skills in the span is remarkably good, considering that the metric is case-sensitive. Additionally, the high Cosine Similarity of 99.58% suggests that the generated skills were highly similar to the expected ones.

3. **Flan T5-XXL**: Table 25 displays the results for the BLOOM model in the *"ALL skills"* category. As anticipated, the model encounters challenges with *MDC* and *HMDC* types of problems, while it performs well on *HOWA*, *OWA*, and *MWA* types of problems. Notably, it achieves 100% accuracy in solving *NONE* types of problems, which is commendable. However, the Levenshtein score of 8.09 indicates a weakness in understanding the context and completing the skills using different logic and adding multiple sets of words for completing each soft skill. Nevertheless, the overall Levenshtein score of 3.27 across all soft skills in all the samples is still considered decent.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.92	4.00	0.86	0.99	HMDC	10
0.96	2.79	0.90	1.00	HOWA	19
0.97	4.38	0.93	1.00	MC	34
0.92	8.09	0.89	0.99	MDC	11
0.97	2.23	0.91	1.00	MWA	53
1.00	0.00	1.00	1.00	NONE	3
0.97	2.92	0.91	1.00	OWA	63
0.96	3.27	0.91	1.00	All Samples	195

Table 25: Results of **Flan-T5-xxl** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

4. **mT5-XL**: Table 26 presents the results for the mT5-XL model in the *"ALL skills"* category. Interestingly, unlike other models, it demonstrates poor performance on *NONE* types of problems, as evidenced by a RougeL score of 87 and a Levenshtein score of 12.33. However, it excels in solving easier problems, specifically *OWA*, *MWA*, and *HOWA* types, while its performance is less satisfactory for more challenging problems such as *HMDC* and *MDC* types. Overall, considering the Levenshtein score of 2.17 across all samples and all the soft skills mentioned in the span, mT5-XL emerges as the best-performing

model after GPT-3.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.91	4.30	0.88	0.99	HMDC	10
0.95	2.89	0.89	1.00	HOWA	19
0.99	2.29	0.96	1.00	MC	34
0.89	5.09	0.87	0.99	MDC	11
0.97	1.79	0.94	1.00	MWA	53
0.87	12.33	0.83	0.98	NONE	3
0.97	0.97	0.93	1.00	OWA	63
0.96	2.18	0.93	1.00	All Samples	195

Table 26: Results of **mT5-xl** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

5. **T5-large**: Table 27 displays the results for the T5-large model in the *"ALL skills"* category. The overall Levenshtein score of 9.38 indicates a lower level of accuracy, and the fact that only 74.9% of all skills were exactly matched is also not satisfactory. It performs best on OWA problems, which is not surprising, but its performance in all other major types of problems and overall is notably poor. Another noteworthy aspect to highlight is the particularly low Levenshtein score of 36.5 and RougeL score of 0.78 for the *MDC* type of problems. These scores indicate the high level of difficulty associated with these problems and the significant challenges they pose for the models.

   One of the examples where the Levenshtein distance = 148 is following:

   Input : Ihre Stärken sind <SoftSkill_C>*strukturiertes*</SoftSkill_C>, <SoftSkill>*zielbewusstes Denke*n</SoftSkill> und <SoftSkill_C>*Arbeiten*</SoftSkill_C> sowie <SoftSkill_C>*Sicherheit*</SoftSkill_C> und <SoftSkill>*Gewandtheit im mündlichen und schriftlichen Ausdruck*</SoftSkill>

   Expected output : Ihre Stärken sind <SoftSkill>*strukturiertes Denken*</SoftSkill>, <SoftSkill>*zielbewusstes Denken*</SoftSkill> und <SoftSkill>*zielbewusstes Arbeiten*</SoftSkill> sowie <SoftSkill>*Sicherheit im mündlichen und schriftlichen Ausdruck*</SoftSkill> und <SoftSkill>*Gewandtheit im mündlichen und schriftlichen*

Ausdruck</SoftSkill>.

T5-large generation: Ihre Stärken sind SoftSkill>*strukturiertes Denken*/SoftSkill> und SoftSkill>*zielbewusstes Denken*/SoftSkill> und SoftSkill>SoftSkill>SoftSkill> SoftSkill> SoftSkill>SoftSkill>SoftSkill> SoftSkill> SoftSkill>SoftSkill>SoftSkill> SoftSkill>Arbeiten/SoftSkill> sowie SoftSkill>Sicherheit/SoftSkill> und SoftSkill>*Gewandtheit im mündlichen und schriftlichen Ausdruck*/SoftSkill>.

In this analysis, the generated text faced difficulties in expanding the word "Arbeiten", while it made no attempt to generate any text for "Sicherheit". This is an improvement over the former case, where random text was generated.

Furthermore, when considering the average and median Levenshtein scores within 2 standard deviations, they amount to 25.4 and 18.5, respectively. These scores underscore the inadequate performance of T5-Large in this context.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.87	11.80	0.75	0.98	HMDC	10
0.89	3.26	0.79	0.99	HOWA	19
0.88	14.35	0.72	0.98	MC	34
0.78	36.55	0.57	0.95	MDC	11
0.82	9.64	0.69	0.97	MWA	53
0.96	3.33	0.83	0.98	NONE	3
0.93	3.81	0.84	0.99	OWA	63
0.88	9.39	0.75	0.98	All Samples	195

Table 27: Results of **T5-large** on *"ALL skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

Based on the results obtained from the evaluation of the five models on the *All Skills* types of problems, the following conclusions can be drawn:

- GPT-3 and mT5-XL demonstrated promising performance, with the metrics for all samples for mT5-XL being comparable to GPT-3.

- All models, except GPT-3, faced challenges in accurately handling *HMDC* and *MDC* types of problems.
- Models performed better on *OWA* and *MWA* types of problems, showcasing good results in these categories.
- mT5 encountered difficulties in correctly solving problems where no changes were required. However, other models like BLOOM and FLAN-T5-XXL managed to learn this aspect effectively.
- T5-large, being one of the smallest models used in the evaluation, exhibited poor performance and could not match the results of other models.

These findings provide valuable insights into the strengths and weaknesses of each model in handling the different types of problems in the *All Skills* category.

6.2.2.3 Evaluation on *C Skills*

Here we do a similar comprehensive comparison of five fine-tuned LLMs on the German dataset. In this section, we will be looking at how different LLMs solve the *C Skills* category of problems

1. **GPT-3**: Table 28 presents the results for GPT-3 in solving this category of problems. The model achieved perfect results for all cases involving hyphenated text. However, similar to its performance in the *All Skills* category, the results for *NONE* types of problems are still not satisfactory. On the other hand, the performance on *OWA* and *MC* types of problems is highly commendable. Overall, the Levenshtein score of 1.10 across all problem types and samples indicates a very impressive performance.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	8
1.00	0.00	1.00	1.00	HMDC	5
1.00	0.00	1.00	1.00	HOWA	30
0.96	0.24	0.97	1.00	MC	50
0.81	3.43	0.79	0.99	MDC	30
0.95	2.29	0.91	0.99	MWA	66
0.86	7.67	0.67	0.99	NONE	3
0.95	0.34	0.96	1.00	OWA	98
0.94	1.10	0.93	1.00	All Samples	292

Table 28: Results of **GPT-3** on *"C skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

2. **BLOOM**: Table 29 displays the results for BLOOM in solving this category of problems. The performance of *HMDC* types of problems is fairly poor. The overall Levenshtein score increased from 3.01 in the *All Skills* category to 4.09 in the *C Skills* category of problems. This suggests that the specific challenge lies in completing the soft skills rather than reproducing the already completed ones. The model also does not perform well on the *MC*, *MDC*, *MWA*, *HMC*, and *HMDC* types of problems. This indicates that the model struggles in completing tasks that involve more than one soft skill.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
0.83	5.00	0.50	0.97	HMC	1
0.79	15.20	0.64	0.98	HMDC	10
0.92	0.95	0.89	1.00	HOWA	19
0.92	5.71	0.84	0.99	MC	34
0.91	5.45	0.87	0.98	MDC	11
0.91	4.17	0.71	0.99	MWA	53
1.00	0.00	1.00	1.00	NONE	3
0.93	2.37	0.84	0.99	OWA	63
0.92	4.10	0.80	0.99	All Samples	195

Table 29: Results of **BLOOM** on *"C skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

3. **Flan-T5-XXL**: Table 30 presents the results for Flan-T5-XXL in solving this category of problems. The model performs well on *OWA* and *MWA* problem types, indicating its effectiveness in completing a single soft skill by adding one or a few words from the nearby context. However, its performance on problem types that involve completing more than one soft skill (e.g., *MC* and *MDC*) is not as good. Nevertheless, the overall Levenshtein score of 2.74 is reasonable.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.69	4.40	0.72	0.98	HMDC	10
0.84	0.95	0.84	0.99	HOWA	19
0.90	4.26	0.86	0.99	MC	34
0.88	6.27	0.83	0.98	MDC	11
0.91	1.98	0.77	0.99	MWA	53
1.00	0.00	1.00	1.00	NONE	3
0.95	2.38	0.81	0.99	OWA	63
0.90	2.75	0.81	0.99	All Samples	195

Table 30: Results of **Flan-T5-xxl** on *"C skills"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

4. **mT5-XL**: Table 31 displays the results for mT5-XL in solving this category of problems. Similar to the *All Skills* category, the model comes the closest to the GPT-3 results for the *C Skills* category as well. For problem types involving completing one skill (*OWA*, *MWA*, and *HOWA*), the model's performance is comparable to GPT-3. Compared to other open-source models in our evaluation, mT5-XL performs well on *MDC* types of problems. This indicates the model's understanding of the task where more than one logic is used to complete multiple soft skills.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.89	4.40	0.84	0.98	HMDC	10
0.79	1.05	0.79	0.99	HOWA	19
0.92	2.29	0.90	1.00	MC	34
0.76	3.27	0.78	0.99	MDC	11
0.89	1.79	0.82	1.00	MWA	53
0.74	12.33	0.67	0.96	NONE	3
0.89	0.65	0.86	1.00	OWA	63
0.88	1.80	0.85	0.99	All Samples	195

Table 31: Results of **mT5-xl** on *"C skills"*. Section 4.5 provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

5. **T5-Large**: Table 32 presents the results for T5-Large, and with an overall Levenshtein score of 8.84, it is the worst-performing model among all the models evaluated. The model's performance in problem types that involve completing more than one skill (*MC*) or require completing multiple skills with different logic (*MDC* and *HMDC*) is particularly poor.

Rouge-L Score	Levenshtein	% Skills	Cosine Similarity	Problem Type	Num
1.00	0.00	1.00	1.00	HMC	1
0.62	11.90	0.56	0.97	HMDC	10
0.68	1.58	0.74	0.99	HOWA	19
0.71	14.06	0.55	0.96	MC	34
0.47	32.45	0.30	0.92	MDC	11
0.43	9.60	0.39	0.95	MWA	53
0.92	3.33	0.67	0.96	NONE	3
0.80	3.51	0.64	0.98	OWA	63
0.65	8.84	0.54	0.97	All Samples	195

Table 32: Results of **T5-large** on *"C skills"*. Section 4.5 provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

From the performances of all models on the *C Skill* category of problems, we can conclude the following.

- Overall, GPT-3 performed best in solving all types of problems except for the ones where the soft skill was already complete.
- mT5-XL came closest to GPT-3 in terms of performance across all samples involving all problem types.
- T5-Large was the worst performing model
- All models performed better at cases which involved the expansion of only one soft skill ie (*OWA* and *MWA*)
- All models struggled at cases that involved the expansion of more than one soft skill and that too involved different logic for expansion ie (*MDC* and *HMDC*)

6.2.2.4 Evaluation on *Complete Sentences*

In this section, we will examine how different Language Model Models (LLMs) tackle the *Complete Sentences* category of problems. Unlike the previous two categories, which focused on either all the soft skills or only partially completed soft skills, this category emphasizes the overall quality of the generated sentence.

1. **GPT-3**: Table 33 displays the results for GPT-3 in the *Complete Sentences* category. The average Levenshtein distance has increased due to the comparison being made on a larger length of text. However, an average Levenshtein distance of 1.75 is still low for the entire text. Interestingly, the model performs exceptionally well for two of the toughest types of problems (*HMDC* and *MDC*), achieving Cosine Similarity scores of 100% and 99.86% respectively. Even for hyphenated cases (HMC, HMDC, and HOWA), the model shows near-perfect accuracy. The only issue can be observed in solving the *NONE* types of problems.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
1.00	1.88	1.00	HMC	8
1.00	0.00	1.00	HMDC	5
1.00	0.17	1.00	HOWA	30
0.99	0.84	1.00	MC	50
0.99	4.10	1.00	MDC	30
0.99	3.70	1.00	MWA	66
1.00	7.67	0.99	NONE	3
1.00	0.62	1.00	OWA	98
1.00	1.76	1.00	All Samples	292

Table 33: Results of **GPT-3** on *"Complete Sentences"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

2. **BLOOM**: Table 34 presents the results for BLOOM in the "Complete Sentences" category. The average Levenshtein distance of 14.81 is high, indicating that the model's performance is not satisfactory overall. Interestingly, the model performs poorly for two of the easiest types of problems (*OWA* and *MWA*). However, it did not perform this poorly in the *All Skills* and *C Skills* categories for the same types of problems. It is worth noting that the model

performed well for *HOWA* types of problems but very poorly for *OWA* types of problems. This discrepancy might be influenced by the effect of outliers, which requires further investigation.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
0.93	19.00	0.99	HMC	1
0.96	21.20	0.99	HMDC	10
0.98	3.26	0.99	HOWA	19
0.97	11.68	0.99	MC	34
0.98	11.45	0.99	MDC	11
0.97	21.36	0.98	MWA	53
1.00	0.33	1.00	NONE	3
0.93	14.92	0.97	OWA	63
0.96	14.82	0.98	All Samples	195

Table 34: Results of **BLOOM** on *"Complete Sentences"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

3. **Flan T5 XXL**: Table 35 displays the results for Flan T5 XXL in the *Complete Sentences* category. In terms of Average Levenshtein distance, this model comes closest to the GPT-3 model but still has a significant gap. As expected, it performs better in *HOWA*, *OWA*, and *MWA* types of problems compared to the tougher *HMDC* and *MDC* types of problems. However, its overall performance is not on par with GPT-3, especially in terms of generating high-quality complete sentences.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
1.00	1.00	1.00	HMC	1
0.97	10.20	0.99	HMDC	10
0.99	3.26	0.99	HOWA	19
0.99	5.68	0.99	MC	34
0.98	7.27	0.98	MDC	11
0.98	3.28	0.99	MWA	53
1.00	0.00	1.00	NONE	3
0.98	3.86	0.99	OWA	63
0.98	4.42	0.99	All Samples	195

Table 35: Results of **Flan-T5-xxl** on *"Complete Sentences"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

4. **mT5-XL**: Table 36 presents the results for mT5-XL in the "Complete Sentences" category. In terms of Rouge-L scores and Cosine Similarity metrics, mT5-XL shows decent performance being placed second after GPT-3. Interestingly, it has a very high Levenshtein score of 7.53 for one of the simplest types of problems, *OWA*. However, on the brighter side, it performed well for problems where more than one skill had to be completed and also with a different logic (*MC* and *MDC*). Also, for *HMDC* types of problems, the average Levenshtein score of 8.5 may seem high, but a very high score of Cosine Similarity suggests that the generated texts are very similar overall.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
1.00	1.00	1.00	HMC	1
0.97	8.50	0.99	HMDC	10
0.98	3.37	0.99	HOWA	19
0.99	3.56	1.00	MC	34
0.98	4.00	1.00	MDC	11
0.98	3.26	0.99	MWA	53
1.00	12.33	0.99	NONE	3
0.97	7.54	0.99	OWA	63
0.98	5.13	0.99	All Samples	195

Table 36: Results of **mT5-xl** on *"Complete Sentences"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

5. **T5-Large**: Table 37 displays the results for T5-Large in the *Complete Sentences* category. For the previous two categories of problems where the performance on the overall soft skills was measured, the performance gap between T5-Large and other models was quite significant. However, when we compare complete sentences, it still is the worst model of all, but its performance is very similar to the BLOOM model and is extremely bad. Interestingly, it does perform better than mT5-XL and BLOOM on OWA types of problems. But apart from this, the overall performance is pretty bad.

Rouge-L Score	Levenshtein	Cosine Similarity	Problem Type	num
1.00	1.00	1.00	HMC	1
0.94	23.80	0.99	HMDC	10
0.97	7.89	0.99	HOWA	19
0.95	20.26	0.98	MC	34
0.93	34.82	0.97	MDC	11
0.94	23.26	0.97	MWA	53
1.00	0.00	1.00	NONE	3
0.98	5.25	0.99	OWA	63
0.96	15.52	0.98	All Samples	195

Table 37: Results of **T5-large** on *"Complete Sentences"*. Section 4.5, provides a comprehensive explanation of all the mentioned error metrics. The results presented in the table have been arranged in ascending alphabetical order based on the *Problem Type*

The following conclusions can be reached:

- GPT-3 exhibits superior performance compared to the rest of the models by a significant margin.
- BLOOM performed substantially worse compared to GPT-3, mT5-XL and Flan-T5-XXL which was not the case when only the individual soft skills were evaluated. This suggests that outside the soft skills part, the model significantly alters the rest of the sentence as well.
- mT5-XL performed very well overall and was closest to the GPT-3. Although the average Levenshtein score for the *Complete Sentences* category was substantially higher than the GPT-3 model, this was not the case when we compared only the soft skills. Hence, one aspect where mT5-XL substantially deviates from GPT-3 is the alteration of the text outside of the soft skill tags.
- Considering the difference in metrics between GPT-3 and other LLMs, it is evident that GPT-3 is the clear winner when evaluating the overall quality of sentences.

6.2.3 Qualititative Evaluation

In this section, we conduct a detailed comparison of the two top-performing models from our previous quantitative evaluation, namely **GPT-3** and **mT5-XL**.

Initially, we focus on specific noteworthy cases where GPT-3 demonstrates exceptional performance for our tasks. The input data used in the datasets are extracted from another NER (Named Entity Recognition) model that detects soft skills in job ads. These skills are then tagged as either *SoftSkill* or *SoftSkill_C*, depending on whether they are complete on their own or represent condensed versions. Consequently, the quality of the input is crucial for our task and for creating the Gold Standard dataset.

Our aim is to retain the entire sentence unchanged, except for the text between *<SoftSkill_C>* and *</SoftSkill_C>*, which needs to be completed by adding certain words from the nearby complete representation of soft skills (*<SoftSkill>* and *</SoftSkill>*).

However, in practice, the training and testing samples may contain errors in the input. There are cases where even the text between *<SoftSkill>* and *</SoftSkill>* is not a complete representation, or the input data is defective. In certain instances, words outside the *<SoftSkill>* and *</SoftSkill>* tags also need to be added to achieve a complete representation of the soft skills.

Remarkably, GPT-3 addresses these challenges and attempts to solve them, even without being explicitly trained for such tasks. Let us examine some examples illustrating these scenarios, which are presented in Table 38:

1. GPT-3: Examples mentioned in the 1st and 3rd points highlight cases where words outside *<SoftSkill>* and *</SoftSkill>* and *<SoftSkill_C>* and *</SoftSkill_C>* tags are also modified to create a comprehensive representation of soft skills.

2. The remaining examples depict instances where the original input was incorrect. For instance, in the 4th example, the input suggested **Neugier** to be a complete soft skill by itself. However, *Neugier* is an ambiguous soft skill without proper context. GPT-3 addresses this issue by altering the text between *<SoftSkill>* and *</SoftSkill>*, resulting in **Neugier am IT-Umfeld**.

Idx	input	Completion
1.	Sie <SoftSkill>arbeiten gerne fachbezogen</SoftSkill> und verstehen es, <SoftSkill>Jugendliche im selbständigen Arbeiten zu coachen</SoftSkill> **und zu** <SoftSkill_C>motivieren</SoftSkill_C>.	Sie <SoftSkill>arbeiten gerne fachbezogen</SoftSkill> und verstehen es, <SoftSkill>Jugendliche im selbständigen Arbeiten zu coachen</SoftSkill> **und** <SoftSkill>Jugendliche im selbständigen Arbeiten **zu** motivieren</SoftSkill>.
2.	Durch Ihr <SoftSkill_C>engagiertes</SoftSkill_C>, <SoftSkill_C>freundliches</SoftSkill_C> und <SoftSkill>kundenorientiertes Auftreten</SoftSkill> <SoftSkill>**tragen Sie als belastbare**</SoftSkill> und <SoftSkill>flexible Persönlichkeit</SoftSkill> einen wesentlichen Teil zur erfolgreichen Teamzusammenarbeit bei.	Durch Ihr <SoftSkill>engagiertes Auftreten</SoftSkill>, <SoftSkill>freundliches Auftreten</SoftSkill> und <SoftSkill>kundenorientiertes Auftreten</SoftSkill> <SoftSkill>**tragen Sie als belastbare Persönlichkeit**</SoftSkill> und <SoftSkill>flexible Persönlichkeit</SoftSkill> einen wesentlichen Teil zur erfolgreichen Teamzusammenarbeit bei.
3.	<SoftSkill>Interesse an der Kommunikation</SoftSkill> **und der** <SoftSkill_C>Teamarbeit</SoftSkill_C>	<SoftSkill>Interesse an der Kommunikation</SoftSkill> **und** <SoftSkill>Interesse an **der** Teamarbeit</SoftSkill>.
4.	<SoftSkill_C>Freude</SoftSkill_C> und <SoftSkill>**Neugier**</SoftSkill> am IT-Umfeld	<SoftSkill>Freude am IT-Umfeld</SoftSkill> und <SoftSkill>**Neugier am IT-Umfeld**</SoftSkill>
5.	Sie <SoftSkill>führen die Aufgaben verantwortungsbewusst</SoftSkill>, <SoftSkill_C>mit Freude</SoftSkill_C> und <SoftSkill>**Engagement**</SoftSkill> aus und gewährleisten ein hohes Dienstleistungsniveau.	Sie <SoftSkill>führen die Aufgaben verantwortungsbewusst</SoftSkill>, <SoftSkill>führen die Aufgaben mit Freude</SoftSkill> und <SoftSkill>**führen die Aufgaben Engagement**</SoftSkill> aus und gewährleisten ein hohes Dienstleistungsniveau.
6.	Du <SoftSkill>arbeitest selbständig</SoftSkill>, <SoftSkill_C>motiviert</SoftSkill_C> und <SoftSkill>**gerne im Team**</SoftSkill>.	Du <SoftSkill>arbeitest selbständig</SoftSkill>, <SoftSkill>arbeitest motiviert</SoftSkill> und <SoftSkill>**arbeitest gerne im Team**</SoftSkill>.
7.	<SoftSkill_C>**Gute** </SoftSkill_C><SoftSkill_C>**konzeptionelle** </SoftSkill_C> und <SoftSkill>**analytische Fähigkeiten**</SoftSkill>	<SoftSkill>**Gute konzeptionelle Fähigkeiten**</SoftSkill> und <SoftSkill>analytische Fähigkeiten</SoftSkill>

Table 38: Interesting cases of reformulation by GPT-3

Despite the numerous positive aspects of reformulating condensed representations, the model also exhibits several weaknesses, which are thoroughly explored and summarized in Table 39.

input	Generation	GS
Dank ihrem breiten und vertieften Fachwissen können Sie <SoftSkill>Kunden situationsgerecht und kompetent beraten</SoftSkill> und <SoftSkill_C>informieren </SoftSkill_C>.	Dank ihrem breiten und vertieften Fachwissen können Sie <SoftSkill>Kunden situationsgerecht und kompetent beraten</SoftSkill> und <SoftSkill>**Kunden informieren**</SoftSkill>.	Dank ihrem breiten und vertieften Fachwissen können Sie <SoftSkill>Kunden situationsgerecht und kompetent beraten</SoftSkill> und <SoftSkill> Kunden situationsgerecht und kompetent informieren </SoftSkill>.
Sie können <SoftSkill>Veränderungen erfolgreich</SoftSkill> und <SoftSkill>motivierend initiieren</SoftSkill>, <SoftSkill_C>umsetzen</SoftSkill_C> und <SoftSkill_C> begleiten </SoftSkill_C>.	Sie können <SoftSkill> Veränderungen erfolgreich initiieren </SoftSkill> und <SoftSkill> motivierend initiieren </SoftSkill>, <SoftSkill> Veränderungen umsetzen </SoftSkill> und <SoftSkill> **motivierend begleiten** </SoftSkill>.	Sie können <SoftSkill>Veränderungen erfolgreich initiieren</SoftSkill> und <SoftSkill>motivierend initiieren</SoftSkill>, <SoftSkill>Veränderungen umsetzen</SoftSkill> und <SoftSkill>Veränderungen begleiten </SoftSkill>.
Sie besitzen ein <SoftSkill>gutes Verständnis für Geodaten</SoftSkill> und <SoftSkill_C> zeitgemässe Technologien </SoftSkill_C>.	Sie besitzen ein <SoftSkill>gutes Verständnis für Geodaten</SoftSkill> und <SoftSkill>**ein zeitgemässe Technologien**</SoftSkill>.	Sie besitzen ein <SoftSkill>gutes Verständnis für Geodaten</SoftSkill> und <SoftSkill> gutes Verständnis für zeitgemässe Technologien </SoftSkill>.
Sie <SoftSkill> arbeiten sehr strukturiert </SoftSkill> und haben eine <SoftSkill_C> ausgeprägte </SoftSkill_C>, <SoftSkill_C> analytische</SoftSkill_C> und <SoftSkill> pragmatische Denkweise </SoftSkill>	Sie <SoftSkill> arbeiten sehr strukturiert </SoftSkill> und haben eine <SoftSkill> **ausgeprägte analytische Denkweise** </SoftSkill>, <SoftSkill> analytische Denkweise </SoftSkill> und <SoftSkill> pragmatische Denkweise </SoftSkill>.	Sie <SoftSkill> arbeiten sehr strukturiert </SoftSkill> und haben eine <SoftSkill> ausgeprägte Denkweise </SoftSkill>, <SoftSkill> analytische Denkweise </SoftSkill> und <SoftSkill> pragmatische Denkweise </SoftSkill>.
Die pädagogische Arbeit planen Sie <SoftSkill_C> zielorientiert </SoftSkill_C> und <SoftSkill>einfühlsam in Partner-schaft mit den Eltern und dem Team </SoftSkill>.	Die pädagogische Arbeit planen Sie <SoftSkill> zielorientiert in Partner-schaft mit den Eltern und dem Team </SoftSkill> und <SoftSkill> **einfühlsam** </SoftSkill>.	Die pädagogische Arbeit planen Sie <SoftSkill> zielorientiert in Partner-schaft mit den Eltern und dem Team </SoftSkill> und <SoftSkill> einfühlsam in Partner-schaft mit den Eltern und dem Team </SoftSkill>.
<SoftSkill>Unternehmerisches Verständnis</SoftSkill> und <SoftSkill_C> Denken </SoftSkill_C>	<SoftSkill>Unternehmerisches Verständnis</SoftSkill> und <SoftSkill> **denken** </SoftSkill>.	<SoftSkill>Unternehmerisches Verständnis</SoftSkill> und <SoftSkill> Unternehmerisches denken </SoftSkill>.

Table 39: Incorrect cases of reformulation by GPT-3

6.2.3.1 GPT-3 vs mT5-XXL

In this section, we conduct an analysis comparing the performance of two models, GPT-3 and mT5-XXL, to determine how each model processes the same text and in which cases one outperforms the other.

For the first part, we look at cases where GPT-3 generated better outcomes than mT5-XXL. These results can be seen in Table 40

1. In the first example (Table 40), there are two sets of *<SoftSkill_C>* tags that need to be completed ("offenen" and "schnell"). However, due to faulty input, there is only one set of *<SoftSkill>* tags mentioned, *<SoftSkill> positiven Art </SoftSkill>*, which allows completing "offenen" to form *<SoftSkill> offenen Art </SoftSkill>*, but there is no *<SoftSkill>* tag to help complete "schnell".

   Remarkably, both models display the ability to search for relevant text in the sentence to complete the soft skills, even beyond the designated *<SoftSkill>* tags. They understand the context as "bedienen Sie unsere Gäste freundlich," which should be used to complete "schnell." However, mT5-XXL changes the natural order of "schnell," while GPT-3 places it correctly, resulting in a more fluent and natural reading experience.

2. The second example showcases GPT-3's impressive ability to correctly complete the dense formulation between *<SoftSkill>* tags. This case involves faulty input because "Überzeugung" alone is not a complete soft skill that can be mapped to the ESCO ontology. However, it would be unfair to fault mT5-XXL for this example since it is not explicitly trained to modify the text between *<SoftSkill>* tags.

3. In the third example, GPT-3 successfully corrects the word "Kassenbedienung," while mT5-XXL leaves it unchanged. However, it is important to note that mT5-XXL is not explicitly trained to modify any text outside the *<SoftSkill_C>* tags, and thus it performs as intended.

4. The final example is also noteworthy, as GPT-3 correctly understands from the context that "qualitätsorientiert" should be expanded using both "denken" and "handeln." On the other hand, the resulting text generated by mT5-XXL is not incorrect but incomplete.

input	mT5-XL	GPT-3
Mit Ihrer <SoftSkill_C> offenen </SoftSkill_C> und <SoftSkill> positiven Art </SoftSkill> bedienen Sie unsere Gäste freundlich,<SoftSkill_C> schnell </SoftSkill_C> und nach STARBUCKS® Standards.	Mit Ihrer <SoftSkill> offenen Art </SoftSkill> und <SoftSkill> positiven Art </SoftSkill> bedienen Sie unsere Gäste freundlich, <SoftSkill> **schnell bedienen Sie unsere Gäste** </SoftSkill> und nach STARBUCKS® Standards	Mit Ihrer <SoftSkill> offenen Art </SoftSkill> und <SoftSkill> positiven Art </SoftSkill> bedienen Sie unsere Gäste freundlich, <SoftSkill> bedienen Sie unsere Gäste schnell </SoftSkill> und nach STARBUCKS® Standards.
Sie sind eine <SoftSkill_C> gestandene </SoftSkill_C>, "<SoftSkill> proaktive" Persönlichkeit </SoftSkill>, die durch <SoftSkill> Überzeugung </SoftSkill> und <SoftSkill> Durchsetzungskraft </SoftSkill> <SoftSkill> Dinge verändern bzw. umsetzen kann </SoftSkill>.	Sie sind eine <SoftSkill> gestandene Persönlichkeit </SoftSkill>, "<SoftSkill> proaktive Persönlichkeit </SoftSkill>, die durch <SoftSkill> **Überzeugung** </SoftSkill> und <SoftSkill>Durchsetzungskraft </SoftSkill> <SoftSkill> Dinge verändern bzw. umsetzen kann </SoftSkill>.	Sie sind eine <SoftSkill> gestandene Persönlichkeit</SoftSkill>, "<SoftSkill> proaktive Persönlichkeit </SoftSkill>, die durch <SoftSkill> Überzeugungskraft </SoftSkill> und <SoftSkill> Durchsetzungskraft </SoftSkill> <SoftSkill>Dinge verändern bzw. umsetzen kann</SoftSkill>.
Eine <SoftSkill_C> freundliche </SoftSkill_C>, <SoftSkill> zuvorkommende Bedienung </SoftSkill> mit der nötigen Effizienz in der Verkaufsabwicklung (Beratung und Kas- senbedienung) und ein <SoftSkill> wertschätzendes Miteinander im Team sind für Sie selbstverständlich </SoftSkill>.	Eine <SoftSkill> freundliche Bedienung </SoftSkill>, <SoftSkill>zuvorkommende Bedienung</SoftSkill> mit der nötigen Effizienz in der Verkaufsabwicklung **(Beratung und Kas- senbedienung)** und ein <SoftSkill> wertschätzendes Miteinander im Team sind für Sie selbstverständlich </SoftSkill>.	Eine <SoftSkill> freundliche Bedienung </SoftSkill>, <SoftSkill>zuvorkommende Bedienung</SoftSkill> mit der nötigen Effizienz in der Verkaufsabwicklung (Beratung und Kassebedienung) und ein <SoftSkill> wertschätzendes Miteinander im Team sind für Sie selbstverständlich </SoftSkill>.
Sie <SoftSkill_C> denken </SoftSkill_C> und <SoftSkill> handeln kunden-</SoftSkill> und <SoftSkill_C> qualitätsorientiert </SoftSkill_C>.	Sie <SoftSkill> denken kundenorientiert </SoftSkill> und <SoftSkill>handeln kundenorientiert </SoftSkill> und <SoftSkill> **handeln qualitätsorientiert** </SoftSkill>.	Sie <SoftSkill> denken kundenorientiert </SoftSkill> und <SoftSkill> handeln kundenorientiert </SoftSkill> und <SoftSkill> denken qualitätsorientiert </SoftSkill> und <SoftSkill> handeln qualitätsorientiert </SoftSkill>.

Table 40: Results where GPT-3 performed better than mT5-XXL

In summary, the evaluation of different language models reveals that GPT-3 demonstrates a higher level of intelligence and robustness when processing input sentences. Even in cases where the input sequences are imperfect, containing spelling errors or misplaced *<SoftSkill>* tags, GPT-3 exhibits remarkable performance and accuracy in completing the tasks at hand. This ability to handle various imperfections in the input text underscores the effectiveness and versatility of GPT-3 as a language model. On the other hand, mT5-XXL excelled in tasks aligned with its specific

training and was adept at completing condensed soft skills representations within the designated format.

For the next part, we look at cases where mT5-XXL generated better outcomes than GPT-3. These results can be seen in Table 41

1. In the first example, GPT-3 struggles to generate a complete representation of the soft skills. While it produces *"freundliches Auftreten"* and *"soziales Auftreten"* as improved versions compared to the original soft skills *"freundliches"* and *"soziales"* in the input, mT5-XXL provides a more comprehensive reformulation.

2. The second example is akin to the first one, wherein GPT-3 expands the condensed expression *"vertrauenswürdig"* into *"agieren vertrauenswürdig"*. However, mT5-XXL accurately reproduces the exact representation by utilizing the complete information *"agieren in schwierigen Situationen"* from the corresponding *<SoftSkill>* tag.

3. The third example demonstrates mT5-XXL's ability to maintain information completeness based on context. Here, the two soft skills *'Umgang'* and *'Betreuung von Menschen'* are incomplete representations and should be completed using the information from *<SoftSkill>Freude am Beraten</SoftSkill>*. While both models correctly complete the condensed formulation to generate *'Freude am Betreuung von Menschen'* for the second soft skill, GPT-3 generates *'Freude am Umgang'* for the first soft skill. However, considering the structure of the second soft skill, *'Freude am Umgang mit Menschen'* appears to be a more consistent reformulation. Notably, mT5 does not merely copy text from the context but rather replaces *von Menschen* with *mit Menschen*.

4. The last example aligns with the previous ones, involving the task of completing *<SoftSkill_C> fachlichen </SoftSkill_C>* by utilizing the information from the corresponding soft skill information *ausgeprägten sozialen Kompetenzen*. Based on contextual information, it seems more consistent to add *'sozialen Kompetenzen'* instead of just *'Kompetenzen'* to complete the condensed expression.

input	mT5-XL	GPT-3
Ein <SoftSkill_C> freundliches </SoftSkill_C>, <SoftSkill_C> soziales </SoftSkill_C> und <SoftSkill> hilfsbereites Auftreten gegenüber Kunden und Kollegen </SoftSkill	Ein <SoftSkill> **freundliches Auftreten gegenüber Kunden und Kollegen** </SoftSkill>, <SoftSkill> **soziales Auftreten gegenüber Kunden und Kollegen** </SoftSkill> und <SoftSkill> hilfsbereites Auftreten gegenüber Kunden und Kollegen </SoftSkill>	Ein <SoftSkill> freundliches Auftreten </SoftSkill>, <SoftSkill> soziales Auftreten </SoftSkill> und <SoftSkill> hilfsbereites Auftreten gegenüber Kunden und Kollegen </SoftSkill>
Sie <SoftSkill> agieren in schwierigen Situationen ruhig </SoftSkill>, <SoftSkill_C> vertrauenswürdig </SoftSkill_C> und <SoftSkill_C> mit souveräner Umsicht </SoftSkill_C>.	Sie <SoftSkill> agieren in schwierigen Situationen ruhig </SoftSkill>, <SoftSkill> **agieren in schwierigen Situationen vertrauenswürdig** </SoftSkill> und <SoftSkill> agieren in schwierigen Situationen mit souveräner Umsicht </SoftSkill>.	Sie <SoftSkill> agieren in schwierigen Situationen ruhig </SoftSkill>, <SoftSkill> agieren vertrauenswürdig </SoftSkill> und <SoftSkill> agieren mit souveräner Umsicht </SoftSkill>.
Hast du <SoftSkill>Freude am Beraten</SoftSkill>, am <SoftSkill_C>Umgang</SoftSkill_C> und der <SoftSkill_C>Betreuung von Menschen</SoftSkill_C> und <SoftSkill>kannst gut in einem Team arbeiten</SoftSkill>	Hast du <SoftSkill>Freude am Beraten</SoftSkill>, <SoftSkill>**Freude am Umgang mit Menschen**</SoftSkill> und der <SoftSkill>Freude am Betreuung von Menschen</SoftSkill> und <SoftSkill>kannst gut in einem Team arbeiten</SoftSkill>	Hast du <SoftSkill>Freude am Beraten</SoftSkill>, am <SoftSkill> Freude am Umgang </SoftSkill> und der<SoftSkill>Freude am Betreuung von Menschen</SoftSkill> und <SoftSkill>kannst gut in einem Team arbeiten</SoftSkill>
Im Bewusstsein, dass Ihr Bereich im Hintergrund einen wertvollen Beitrag zu einem optimalen Umfeld schafft, setzen Sie Ihre <SoftSkill_C> fachlichen </SoftSkill_C> und <SoftSkill> ausgeprägten sozialen Kompetenzen </SoftSkill> zielorientiert ein.	Im Bewusstsein, dass Ihr Bereich im Hintergrund einen wertvollen Beitrag zu einem optimalen Umfeld schafft, setzen Sie Ihre <SoftSkill> **fachlichen sozialen Kompetenzen** </SoftSkill> und <SoftSkill> ausgeprägten sozialen Kompetenzen </SoftSkill> zielorientiert ein.	Im Bewusstsein, dass Ihr Bereich im Hintergrund einen wertvollen Beitrag zu einem optimalen Umfeld schafft, setzen Sie Ihre <SoftSkill> fachlichen Kompetenzen </SoftSkill> und <SoftSkill> ausgeprägten sozialen Kompetenzen </SoftSkill> zielorientiert ein.

Table 41: Results where mT5-XXL performed better than GPT-3

Based on the analyses conducted on the performance of GPT-3 and mT5-XXL, we can draw the following conclusions:

1. GPT-3 exhibits a higher level of intelligence and robustness when processing input sentences. Even in cases where the model generating the input sequences is imperfect, GPT-3 continues to perform accurately, showcasing its ability to handle faulty input and generate meaningful outcomes.

2. mT5-XXL also demonstrates commendable performance, particularly in cases where the soft skills need to be completed with comprehensive reformulations.

It excels in generating exact representations based on contextual information from the corresponding *<SoftSkill>* tags.

3. Both models possess unique strengths and weaknesses. GPT-3 showcases an impressive ability to modify text beyond the *<SoftSkill>* tags, while mT5-XXL exhibits a tendency to maintain information completeness based on context.

4. When considering the overall quality of sentences and the ability to generate accurate soft skill representations, GPT-3 remains the top-performing model.

5. Despite its limitations, mT5-XXL still performs well in most cases and competes closely with GPT-3 in terms of overall performance.

In conclusion, GPT-3 and mT5-XXL represent powerful language models with distinctive attributes, and while GPT-3 demonstrates superiority in various aspects, mT5-XXL remains a viable alternative with considerable capabilities in expanding condensed soft skill expressions.

7 Conclusions and Future Work

7.1 Conclusions

This thesis delved into two distinct tasks: the first one, the *Noun Completion Task* involved evaluating the capability of Large Language Models (LLMs) to complete truncated words, while the second task, the *Phrase Expansion Task* focused on reformulating complex coordinated expressions into simpler redundant formulations. The objective was to paraphrase condensed coordinated phrases, which lacked semantic completeness, into self-contained paraphrases by adding contextual words.

We had the following research questions at the beginning of the thesis:

1. Is there a measurable performance difference among modern state-of-the-art transformer architectures (BART, T5, GPT3) for learning to solve text generation problems that reformulate coordinated phrases with elided material into completed, but slightly redundant formulations?

2. What are the differences in the accuracy between few shots classification using GPT3 vs fine-tuning the GPT3 model for reformulating complex coordinated phrases into simpler formulations?

3. Can Large Language Models (LLM) be taught how to generate redundant and elaborate simpler phrases from the original coordinated expressions text without semantically changing the input sentence

4. Can LLMs (Large Language Models) be taught how to solve the problem of ellipsis completion? Ex. Haus- und Gartenarbeit —> Hausarbeit und Gartenarbeit

Addressing the research questions, the following conclusions were drawn:

1. **Performance Difference among Transformer Architectures:** Among the modern transformer architectures (BART, T5, GPT3), it was observed that BART struggled with the simpler task of splitting elided compound nouns. Therefore, BART was not tested in the more complex second task. T5 large

and Flan T5 large models outperformed both variants of GPT3 in the *Noun Completion Task*. However, GPT3 demonstrated outstanding performance across the entire dataset and problem types in the *Phrase Expansion Task*. mT5-XL model was comparable to GPT-3, but since its training data was generated by GPT-3, potential biases might exist. Overall, the open-source models demonstrated comparable accuracy compared to GPT-3.

2. **Accuracy in Few-Shots Classification vs. Fine-Tuning:** Few-shot classification for the *Phrase Expansion Task* yielded poor results for all models. Only chatGPT exhibited superior performance in In-Context Learning. However, fine-tuning with PEFT showed significant improvement in accuracy compared to few-shot learning.

3. **Generation of Redundant and Elaborate Phrases:** The results demonstrated that LLMs can be effectively taught to generate redundant and elaborate formulations from original coordinated expressions without altering the sentence's semantics. Qualitative analysis in the section 6.2.3 supported this observation, indicating the models' ability to reformulate text to generate more elaborate expressions while retaining the original meaning.

4. **Problem of Ellipsis Completion:** The *Noun Completion Task* successfully addressed the problem of ellipsis completion, transforming phrases like "Haus- und Gartenarbeit" into "Hausarbeit und Gartenarbeit." All models exhibited proficiency in solving this problem, showcasing LLMs' robustness compared to traditional methods, which were sensitive to spelling errors and based on morphological splitting and corpus statistics.

7.2 Future Work

In the future, there are many exciting developments happening in the field of NLP, and newer and better language models are being introduced regularly. We used one particular technique called LoRA for training our models, but there are other methods, like the Prompt Tuning approach by Lester et al. [2021], that could be interesting to try out.

Both tasks conducted in this thesis required data extraction from various sources and the manual creation of Gold Standard (GS) datasets. Due to this process's resource-intensive nature, the amount of available data was limited. For the *Phrase Expansion Task* in this thesis for German, more than 50% of the data was the output of the GPT-3 model and was not human-verified. Additionally, the English datasets

for this task consisted of only 20 samples, which were insufficient for fine-tuning the pre-trained LLMs. To address these limitations, future research could focus on generating more data and converting it into Gold Standard format to augment the datasets. Moreover, for better replicability, the English datasets for the *Phrase Expansion Task* should be utilized to fine-tune the pre-trained LLMs.

For the larger models, there are different parameters we can adjust to get better results potentially. We can experiment with different settings and gradient accumulation to see how it affects the models' performance.

In our current approach, we use specific tags (<SoftSkill>, </SoftSkill>, <SoftSkill_C>, and </SoftSkill_C>) to mark the texts that need modification. In the future, we can explore alternative methods that don't rely on these tags, making the data representation more natural. This might help the models better understand the tasks.

References

N. Aepli and M. Volk. Reconstructing complete lemmas for incomplete german compounds. In I. Gurevych, C. Biemann, and T. Zesch, editors, *Language Processing and Knowledge in the Web*, pages 1–13, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

F. Cap, A. M. Fraser, M. Weller, and A. Cahill. How to produce unseen teddy bears: Improved morphological processing of compounds in SMT. In G. Bouma and Y. Parmentier, editors, *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, April 26-30, 2014, Gothenburg, Sweden*, pages 579–587. The Association for Computer Linguistics, 2014. doi: 10.3115/v1/e14-1061. URL https://doi.org/10.3115/v1/e14-1061.

H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, S. Narang, G. Mishra, A. Yu, V. Y. Zhao, Y. Huang, A. M. Dai, H. Yu, S. Petrov, E. H. Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. V. Le, and J. Wei. Scaling instruction-finetuned language models. *CoRR*, abs/2210.11416, 2022. doi: 10.48550/arXiv.2210.11416. URL https://doi.org/10.48550/arXiv.2210.11416.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep

bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

B. Ding, C. Qin, L. Liu, Y. K. Chia, B. Li, S. Joty, and L. Bing. Is GPT-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11173–11195, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.acl-long.626.

C. P. Escartín. Chasing the perfect splitter: A comparison of different compound splitting tools. In N. C. C. Chair), K. Choukri, T. Declerck, H. Loftsson, B. Maegaard, J. Mariani, A. Moreno, J. Odijk, and S. Piperidis, editors, *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may 2014. European Language Resources Association (ELRA). ISBN 978-2-9517408-8-4.

A. Hätty, U. Heid, A. Moskvina, J. Bettinger, M. Dorna, and S. Schulte Im Walde. *AkkuBohrHammer* vs. *AkkuBohrhammer*: Experiments towards the Evaluation of Compound Splitting Tools for General Language and Specific Domains. In *Proceedings of the 15th Conference on Natural Language Processing*, pages 59–67, Erlangen, Germany, 2019.

E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685, 2021. URL https://arxiv.org/abs/2106.09685.

B. Lester, R. Al-Rfou, and N. Constant. The power of scale for parameter-efficient prompt tuning. In M. Moens, X. Huang, L. Specia, and S. W. Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.emnlp-main.243. URL https://doi.org/10.18653/v1/2021.emnlp-main.243.

M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

M. Luo, K. Hashimoto, S. Yavuz, Z. Liu, C. Baral, and Y. Zhou. Choose your QA model wisely: A systematic study of generative and extractive readers for question answering. In *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*, pages 7–22, Dublin, Ireland and Online, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.spanlp-1.2. URL https://aclanthology.org/2022.spanlp-1.2.

M. Moradi, K. Blagec, F. Haberl, and M. Samwald. GPT-3 models are poor few-shot learners in the biomedical domain. *CoRR*, abs/2109.02555, 2021. URL https://arxiv.org/abs/2109.02555.

C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL http://jmlr.org/papers/v21/20-074.html.

T. L. Scao, A. Fan, C. Akiki, E. Pavlick, S. Ilic, D. Hesslow, R. Castagné, A. S. Luccioni, F. Yvon, M. Gallé, J. Tow, A. M. Rush, S. Biderman, A. Webson, P. S. Ammanamanchi, T. Wang, B. Sagot, N. Muennighoff, A. V. del Moral, O. Ruwase, R. Bawden, S. Bekman, A. McMillan-Major, I. Beltagy, H. Nguyen, L. Saulnier, S. Tan, P. O. Suarez, V. Sanh, H. Laurençon, Y. Jernite, J. Launay, M. Mitchell, C. Raffel, A. Gokaslan, A. Simhi, A. Soroa, A. F. Aji, A. Alfassy, A. Rogers, A. K. Nitzav, C. Xu, C. Mou, C. Emezue, C. Klamm, C. Leong, D. van Strien, D. I. Adelani, and et al. BLOOM: A 176b-parameter open-access multilingual language model. *CoRR*, abs/2211.05100, 2022. URL https://doi.org/10.48550/arXiv.2211.05100.

H. Schmid, A. Fitschen, and U. Heid. SMOR: A German computational morphology covering derivation, composition, and inflection. In *Proceedings of the IVth International Conference on Language Resources and Evaluation (LREC 2004)*, pages 1263–1266, 2004. URL http://www.ims.uni-stuttgart.de/www/projekte/gramotron/PAPERS/LREC04/smor.pdf.

D. Testa, E. Chersoni, and A. Lenci. We understand elliptical sentences, and language models should too: A new dataset for studying ellipsis and its interaction with thematic fit. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3340–3353, Toronto, Canada, July 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.acl-long.188.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 483–498. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.41. URL https://doi.org/10.18653/v1/2021.naacl-main.41.