h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Hochschule Darmstadt

Fachbereiche Mathematik und Naturwissenschaften & Informatik

Analyzing Sentiments and Aspects of Fertility Drug Discussions in Social Media: A Comparative Study of Few-Shot Learning and Fine-tuning Techniques with BERT and Large Language Models

Abschlussarbeit zur Erlangung des akademischen Grades

Master of Science (M.Sc.)

vorgelegt von

Sara El-Beit Shawish

Matrikelnummer: 770001

Referent	:	Prof. Dr. Markus Döhring
Korreferent	:	Prof. Dr. Antje Jahn

Ausgabedatum	:	31.05.2023
Abgabedatum	:	29.11.2023

DECLARATION

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe.

Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht.

Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen.

Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, December 5, 2023

Sara El-Beit

ABSTRACT

MOTIVATION: Fertility plays a crucial role in life and has direct effects on the quality of life and mental well-being. Many people face challenges related to fertility disorders and, therefore, getting medical therapies to cope with them. Individuals confronted with fertility issues and getting medical treatment tend to actively engage in social media, sharing and evaluating these treatments. The analysis of these patient reviews, capturing not only emotions but also various aspects of medication, is called aspect-based sentiment analysis. It provides insights into which aspects of the therapy are effective or problematic. However, the lack of domain specificity in existing datasets poses a challenge for aspect-based sentiment analysis. Therefore, developing a method to generate such a dataset is of great importance to train supervised models that analyze emotions and aspects.

METHODS: For the generation of the artificial dataset, a combination of GPT-3.5 and Few-Shot Learning was applied. This involved leveraging the extensive knowledge embedded in the numerous parameters of GPT-3.5 to generate synthetic data. Two BERT models were then trained for the classification of emotions and aspects using this data. Concurrently, manual annotation of data took place, used to train another pair of BERT models. The performance of these BERT models was subsequently compared and evaluated to determine the suitability of an artificially generated dataset for model training. Finally, a dataset comprising 50% manually annotated data and 50% artificially annotated data was created and the experiment repeated.

RESULTS: The evaluation of a Bidirectional Encoder Representations from Transformers (BERT) model fine-tuned with only GPT-3.5 annotated data yields noteworthy findings. The model demonstrates strong performance in aspect prediction with a test accuracy of 0.92, while sentiment prediction slightly lags behind at 0.87. The highest result was observed for a model trained with a mixed dataset, comprised of 50 ratio between human and Generative Pre-trained Transformer 3.5 turbo (GPT-3.5 turbo) annotations with a test accuracy of 0.95 for the aspect prediction.

CONCLUSION: We recommend a nuanced approach that involves GPT-3.5 turbo labeling and human validation.

ZUSAMMENFASSUNG

MOTIVATION: Die Fruchtbarkeit spielt eine wesentliche Rolle im menschlichen Leben und hat direkte Auswirkungen auf die Lebensqualität und das psychische Wohlbefinden. Viele Menschen stehen vor Herausforderungen im Zusammenhang mit Fruchtbarkeitsstörungen und greifen daher auf medikamentöse Therapien zurück, um diese zu bewältigen. Personen, die mit Fruchtbarkeitsstörungen konfrontiert sind und daher auf medikamentöse Therapien zurückgreifen, neigen dazu, sich in sozialen Medien aktiv auszutauschen und die Therapien zu bewerten. Die Analyse dieser Patientenbewertungen, die nicht nur Emotionen, sondern auch verschiedene Aspekte eines Medikaments erfassen, nennt sich aspektbasierte Sentimentanalyse und liefert Erkenntnisse darüber, welche Aspekte der Therapie wirkungsvoll oder problematisch sind. Die fehlende Domänenspezifität in vorhandenen Datensätzen stellt jedoch eine Herausforderung für die aspektbasierte Sentimentanalyse dar. Daher ist die Entwicklung einer Methode zur Generierung eines solchen Datensatzes von großer Bedeutung, um ein überwachte Modelle zu trainieren, die Emotionen und Aspekte analysieren.

METHODEN: Für die Erzeugung des künstlichen Datensatzes wurde eine Kombination aus GPT-3.5 turbound Few-Shot Learning angewendet. Hierbei wurde auf das umfangreiche Wissen, das in den zahlreichen Parametern von GPT-3.5 turbo vorhanden ist, zurückgegriffen, um synthetische Daten zu generieren. Mit diesen Daten wurden zwei BERT-Modelle für die Klassifikation der Emotionen und der Aspekte trainiert. Parallel dazu erfolgte die manuelle Annotation von Daten, mit denen ebenfalls zwei BERT-Modelle trainiert wurden. Die Performance dieser BERT-Modelle wurde anschließend verglichen und evaluiert, inwiefern ein künstlich erzeugter Datensatz sich für das Training eines Modells eignet. Schließlich wurde ein Datensatz aus 50% manuell annotierten Daten und 50% künstlich annotierten Daten erstellt und der Versuch wiederholt.

ERGEBNISSE: Die Auswertung eines BERT-Modells, das mit GPT-3.5 turbo -annotierten Daten trainiert wurde, ergibt bemerkenswerte Ergebnisse. Das Modell zeigt eine starke Performance in der Aspektvorhersage mit einer Testgenauigkeit von 0,92, während die Vorhersage von Emotionen mit 0,87 leicht dahinter liegt. Optimale Ergebnisse werden bei der Aspektvorhersage in einem gemischten Datensatz beobachtet, der zu 50 % aus menschlichen und zu 50 % aus GPT-3.5 turbo Annotationen besteht, mit einer Testgenauigkeit von 0,95 für die Aspektvorhersage.

FAZIT: Wir empfehlen einen nuancierten Ansatz, der die Annotation durch GPT-3.5 turbo und menschliche Validierung einbezieht.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

API	Application Programming Interface
RNNs	Recurrent Neural Networks
LSTMs	Long Short-Term memory
BERT	Bidirectional Encoder Representations from Transformers
GPT-3.5	Generative Pre-trained Transformer 3.5
GPT-3.5 turbo	Generative Pre-trained Transformer 3.5 turbo
NLP	Natural Language Processing
MLM	Masked Language Moddeling
QA	Question-Answering
T5	Text-to-Text-Transformer
LLMs	Large Language Models
LLM	Large Language Mode
ICL	In-Context-Learning
SQuAD	Stanford Question Answering Dataset
Regex	Regular Expressions
SQuAD	The Stanford Question Answering Dataset
FLAN-T5	Fine-tuned Language Net - Text-To-Text Transfer Transformer
Regex	Regular Expression

1 INTRODUCTION

In Germany, nearly every tenth couple aged 25 to 59 is unintentionally childless, according to the Federal Ministry for Family Affairs, Senior Citizens, Women and Youth [19]. Pharmaceutical companies such as Merck provide reproductive medications that assist in fulfilling the desire for parenthood. Patients oftentimes share their experiences about therapies or certain drugs on social media platforms and are therefore a valuable source of information for pharmaceutical companies. Is important from a corporate perspective to analyse social media data to understand which sentiments patients express towards the medications and what aspects of them should be improved.

1.1 MOTIVATION

Until now, Merck uses a model that does the current sentiment analysis on a document/review level. Review level means that a whole review is classified as positive or negative which doesn't reflect reality in a lot of cases. Users can talk about multiple aspects of a product, therefore we aim to perform an aspect-based sentiment analysis. If a user/patient addresses multiple aspects of a medication and expresses different emotions towards them, this information is lost during the analysis. Analyzing social media conversations on a more fine-granular level, holds great potential for companies to better respond to customer demands and continuously improve their products. However, most machine learning algorithms focus on structured information and, therefore, can't handle the high-dimensional, unstructured, and sometimes sparse information in social media text data. The rapid development in the field of Natural Language Processing (NLP) in recent years has made it possible to analyze such text data and perform various tasks such as text classification. Current language models can be used here to perform sentiment analysis and, beyond that, answer questions such as towards wich aspect of the medication a patient expresses negative or positive emotions. This analysis can be carried out in the form of a supervised classification problem using LLMs such as BERT. However, it is challenging to create a domain-specific dataset that is suitable for this specific classification task. Although LLMs are trained on a diverse range of data, they may not have encountered specific domain-related nuances or terminology. Therefore a custom dataset is needed, resulting in the key challenge of investigating methods to automatically generate labeled data that are of high quality and can be used for supervised training.

1.2 OBJECTIVES OF THE THESIS

The overall goal is to get an understanding of patient's opinions regarding certain fertility drugs. Therefore, we need to capture all the opinions expressed in a single review, that contain the relevant information. Relevant means, that the opinion is about one of the pre-defined aspects, that are of interest to the stakeholder. A drug can have many aspects. However, stakeholders prioritize five aspects that play a significant role in their analysis. These aspects are listed and described in table 1.2. The opinions reduce a long review to its most important parts. This is important, cause user reviews can be long and contain a lot of noise, which can affect the analysis in a negative way. After the opinions/text spans have been extracted and labeled, two classification models must be trained, which receive the text spans as input and return a probability for the sentiments and for the five aspects as output. The manual creation of a dataset that satisfies the named requirements, is a time-intensive step.

Therefore, two research questions can be derived from this use case:

1. Can the use of large language models such as GPT-3.5 turbo help to create a domain-specific dataset and therefore reduce labeling costs? The dataset must contain the relevant text passages of a social media review as well as their sentiment and aspect class. The background of the question is that manual labeling for this specific NLP task is very time-consuming
2. Can a mix of human-generated- and GPT-3.5 turbo generated labels further boost performance at a lower cost?

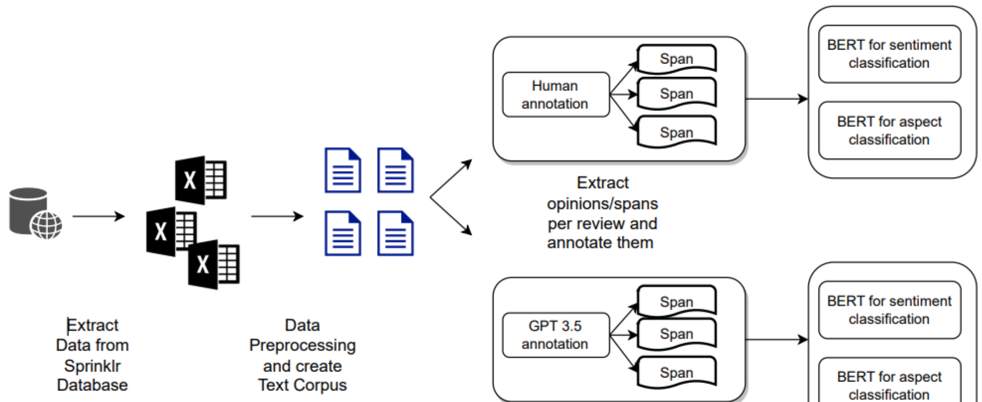
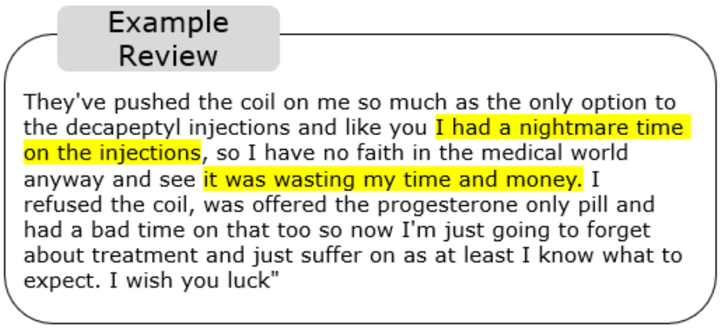


Figure 1.1: Workflow High-Level, describing the process from retrieving the raw data until performing the classification tasks.

The overall workflow on a high level can be obtained from figure 1.1. The raw data gets extracted from a sprinklr database and stored in multiple Excel


Example Review

They've pushed the coil on me so much as the only option to the decapeptyl injections and like you I had a nightmare time on the injections, so I have no faith in the medical world anyway and see it was wasting my time and money. I refused the coil, was offered the progesterone only pill and had a bad time on that too so now I'm just going to forget about treatment and just suffer on as at least I know what to expect. I wish you luck"

Figure 1.2: Example of a typical user review with highlighted opinions.The opinions function as the input for the BERT models.

files. We extract the user reviews from the Excel sheets and apply multiple pre-processing steps which will be discussed in more detail in 3.2.2. Figure 1.2 shows a typical example of a user review. The highlighted text represents the relevant opinions.

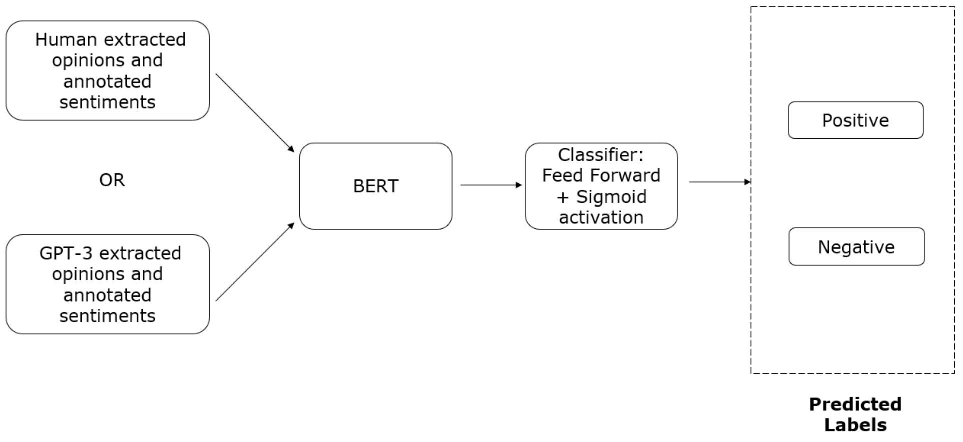

Human extracted opinions and annotated sentiments
OR
GPT-3 extracted opinions and annotated sentiments
BERT
Classifier: Feed Forward + Sigmoid activation
Positive
Negative
Predicted Labels

Figure 1.3: Sentiment Classification. First we use human annotated data as input, then GPT-3.5 turbo annotated data. We then compare the performances of both models.

Those extracted opinions/text spans form the basis for two classification tasks:

1. a binary classification of the sentiment (positive or negative) visualized in figure 1.3
2. a multi-label classification of aspect categories, that were pre-defined by the stakeholder visualized in figure 1.4

The outputs of both classification models should then be concatenated.

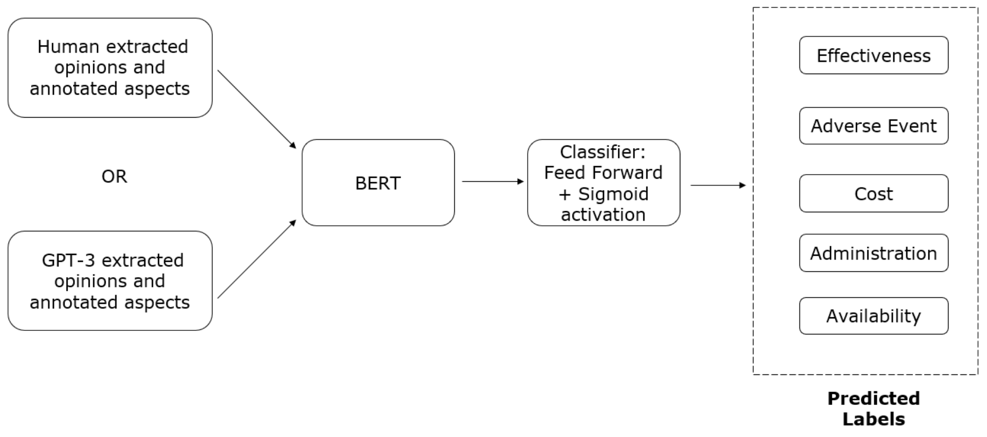

Human extracted opinions and annotated aspects
OR
GPT-3 extracted opinions and annotated aspects
BERT
Classifier: Feed Forward + Sigmoid activation
Effectiveness
Adverse Event
Cost
Administration
Availability
Predicted Labels

Figure 1.4: Aspect Classification. We have five pre-defined and distinct aspect categories/classes. First, we use human annotated data as input, then GPT-3.5 turbo annotated data. We then compare the performances of both models.

In table 1.1 you can see an example of how the final output should look like.

Table 1.1: Example of final output

Opinion/Span	**Aspect**	**Sentiment**
'I got 12 eggs, 6 of them fertilized...'	effectiveness	positive
'Menopur is not available in the Pharmacies right now. There seems to be a shortage'	availability	negative
'I am so frustrated with the injection. It always leaks!"	administration	negative
'...it gave me a really bad headache and I feel sick all the time!!'	adverse event	negative
'...unfortunately, this round failed'	effectiveness	negative
'I got pregnant!!'	effectiveness	positive

The stakeholders limited themselves to five pre-defined aspects which are described in table 1.2.

Table 1.2: Definitions of pre-defined aspects.

Aspect	**Definition**
effectiveness	The drug is considered to be effective, if the patient retrieved eggs, the eggs got fertilized, got embryos or got pregnant
adverse event	This aspect comes into play when the patient talks about complaints related to the medication, for example if they suffered from any kind of pain, insomnia etc.
cost	The patient talks about the money spent on the medication. Either the patient considered it as too expensive or affordable.
availability	Usually, patients talk about drug shortages and non-availability in the pharmacies
administration	The patient talks about the handling of the drug. For example, if the drug comes in form of different powders, the patient describes the procedure of mixing these powders or if the drug comes in the form of an injection, the patient shares the experience with the handling of the injection.

1.3 STRUCTURE OF THE THESIS

First, a brief overview of the medical basis is given in section 2 along with the theoretical background of transformer models. Both the architecture and the way in which they have been trained are explained. In 3 we will have a look on how we created our datasets by explaining pre-processing steps and the annotation process. The fine-tuning of the classification models and their results are then discussed in chapter 4. Finally, an outlook on future challenges is given and a final conclusion is drawn.

2 THEORETICAL BACKROUND

We will start with the medical background and then move on to the concept of language models. We will then take a closer look at one language model in particular, the transformer model. We look at its architecture and some of the special features that distinguish it from other language models. We then describe the transformer models used in this thesis: BERT, GPT-3.5 turbo and Text-to-Text-Transformer (T5). We look at the pre-training, the datasets on which they were trained and the fine-tuning.

2.1 MEDICAL FUNDAMENTALS

This section introduces the medical principles that are important for this study.
IVF, or In Vitro Fertilization, is a medical procedure designed to assist individuals or couples with fertility issues. It involves the conception of a child outside the body, typically in a laboratory setting. Two essential steps in the IVF process are:

1. **Ovulation Stimulation:** In the first phase, the woman undergoes treatment with fertility medications to stimulate the ovaries. This encourages the development of multiple follicles, each containing an egg. Regular monitoring, through blood tests and ultrasounds, ensures that the eggs within the follicles reach an optimal level of maturity.

2. **Egg Retrieval and Fertilization:** Once the eggs within the matured follicles are ready, a minor surgical procedure is performed to retrieve them from the ovaries. In the laboratory, these eggs are then fertilized with sperm. The fertilized eggs, now embryos, are closely monitored for several days before one or more are selected for implantation in the woman's uterus.

It is important to know that the mentions of follicles in the reviews don't determine the success of a fertility drug. More important are the number and quality of the eggs. If the patient mentions only the follicles, the review doesn't contain valuable information here, and therefore nothing should be labeled here. More information can be obtained from table 1.2.

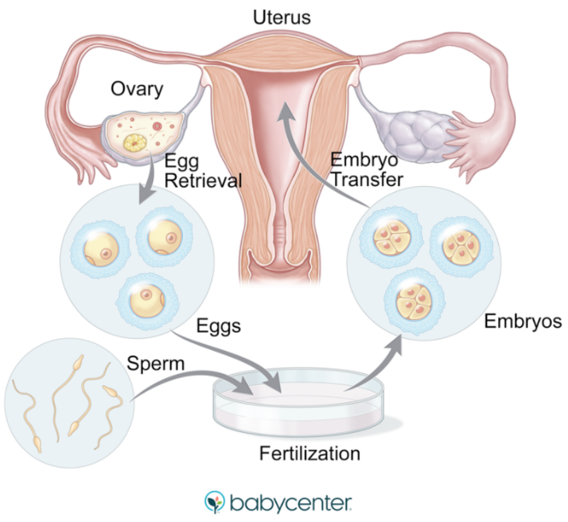


Figure 2.1: Representation of the IFV Process. Source: [14]

2.2 LANGUAGE MODELING

A language model has the objective to learn the true distribution of a text corpus and hence attempts to model a probability distribution over sequences of tokens or words. A popular approach to do so is through next word prediction [32]. Given a context, it predicts the distribution over the word to follow. For example for the context "The food was", the model could place high probabilities on words like "delicious" or "expensive" [32]. A way to compute probabilities of entire sentences would be to use the chain rule of probability to decompose the entire sequence and calculate its joint probability, as shown in the equation:

$$\mathrm{P}(x_1, .., x_n) = P(x_1)P(x_2|x_1)P(x_2)P(x_3|x_{1:2})...P(x_n|x_{1:k\text{-}1}) = \prod_{k=1}^{n} P(x_k|x_{1:k\text{-}1}) \tag{2.1}$$

Equation 2.1 suggests that we could estimate the joint probability of an entire sequence of words by multiplying together a number of conditional probabilities [9]. This is difficult to compute on large text corpora as it requires counting the number of times every word occurs following every long string[9]. To overcome this problem, the n-gram model is introduced, which approximates the history of words by just looking at the last n words where 'n' can be 1 (unigram), 2 (bigram), 3 (trigram), and so on. The bigram approximates the probability of a word given all the previous words by using only the conditional probability of the preceding word [9].

$$\mathrm{P}(w_i|w_{1:i-1}) \approx P(w_i|w_{i-1}) \tag{2.2}$$

The assumption, that the probability of a word depends only on the previous word is called Markov assumption. Markov models predict the probability of some future unit without looking too far into the past [9]. However, this assumption doesn't take into account that language is creative and can change and therefore some contexts might not have occurred before [9]. Another weakness is that this approach ignores long dependencies between words, i.e. a word that refers to a word from much earlier in a sequence. Therefore, they struggle to capture patterns of syntax and semantics in natural language well. Another fundamental problem is the curse of dimensionality, which limits modeling on larger corpora [18].

2.3 NEURAL NETWORK LANGUAGE MODELS

To overcome the curse of dimensionality, the conditional probability distribution is parametrized with the help of low-dimensional vectors, called embeddings. Embeddings learn a distributed representation for words, which constitutes a word as a low dimensional vector [18]. This continuous representation allows the model to capture semantic similarities between words and generalize better to unseen data. Neural networks such as Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks can use such embeddings as input. However, they encounter two key challenges: First, they process sequences one step at a time, making it difficult to parallelize and slowing down training. Second, they also struggle to capture meaningful relationships between distant elements in a sequence, as their influence diminishes quickly over time step. Hence, Transformer models were introduced to address this bottleneck problem. In contrast to N-grams, Recurrent Neural Networks (RNNs) and Long Short-Term memory (LSTMs), Transformers can capture complex patterns and dependencies in language. Notable examples of Transformer-based Models include Generative Pre-trained Transformer (GPT)[5] and Bidirectional Encoder Representations from Transformers (BERT)[10]. Both GPT-3.5 turbo and BERT are current state-of-the-art models and will be discussed in more detail throughout this thesis as they will be used to perform the automated annotation and supervised text classification.

2.4 TRANSFORMERS

Transformer models have significantly impacted the field of Natural Language Processing (NLP). They excel in capturing contextual relationships and long-term dependencies within sequential data, making them a notable choice compared to alternatives like recurrent networks. The paper that first introduced Transformer models was authored by Vaswani et al. in 2017, titled 'Attention is All You Need'[37] and serves as the primary source of reference in this section. This study demonstrated that transformers could achieve state-of-the-art results on a variety of natural language processing tasks, including machine translation, text summarization, and question answering. For example, in a machine translation application the transformer would convert one language to another, or for a classification problem will provide the class probability using an appropriate output layer. This original transformer model is often referred to as the 'vanilla' transformer model. Various architectural variations of this vanilla model have since been proposed in [24]. In the next sections, the transformer architecture is explained along with the transformer models, which were used to perform the aspect-based sentiment analysis: BERT, GPT-3.5 turbo, and T5.

2.4.1 *Transformer Model Types*

There are three categories in which transformer models can be classified:

- **Encoder-only:** These models are typically used to create a representation of the input sentence, primarily for tasks like classification or sequence labeling. Well-known encoder only models include BERT [10] and RoBERTa [25].
- **Decoder-only:** Such models are usually used for auto-regressive sequence generation tasks. GPT-3.5 turbo is an example for such a decoder only model which has shown exemplary results on NLP task.
- **Both Encoder and Decoder:** These models are widely employed in sequence-to-sequence generation tasks, such as neural machine translation, where the generation of tokens relies on both the original input and the tokens already generated. T5 [30] and BART[22] are popular examples of encoder-decoder transformers.

2.4.2 *Transformers Architecture*

The vanilla model architecture proposed by [37] consists of an encoder-decoder structure, which is stacked six times on top of eachother. One encoder-decoder layer is shown in 2.2 where the left side represents the encoder and the right side the decoder. The first layer is the embedding layer, which transforms the original data into a numerical representation, that will be fed into the encoder block.

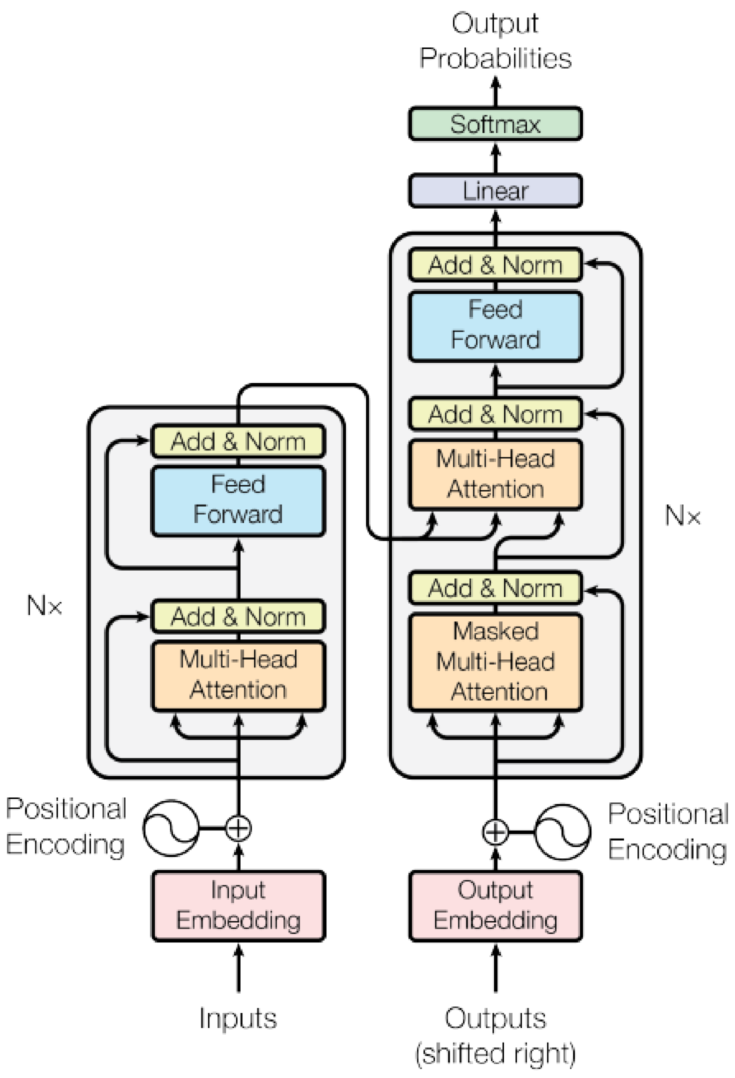


Figure 2.2: Illustration of the Transformer architecture by Vaswani et al. [37].

2.4.2.1 *Encoder*

The encoder aims to project an input sequence of representations to a sequence of learned contextualized embeddings. It consists of a stack of N=6 (proposed in the original paper) encoder blocks, where the output of one block is fed as the input to the next block. Each encoder block has two sub-layers: a multi-head self-attention mechanism and a position-wise feed-forward neural network. The encoder expects its inputs to be of the shape SxD (as shown in figure 2.3). S represents the input sentence length, and D is the length of the embedding, whose weights can be trained with the network.

In particular, BERT has a fixed input size of D=512 tokens. Then it processes these vectors by passing them to a self-attention layer followed by a position-wise feed-forward neural network, which captures local interactions. The output of the last encoder serves as the input for all decoder layers, enabling the decoder to access the fully processed information from the entire input sequence for generating the output sequence.

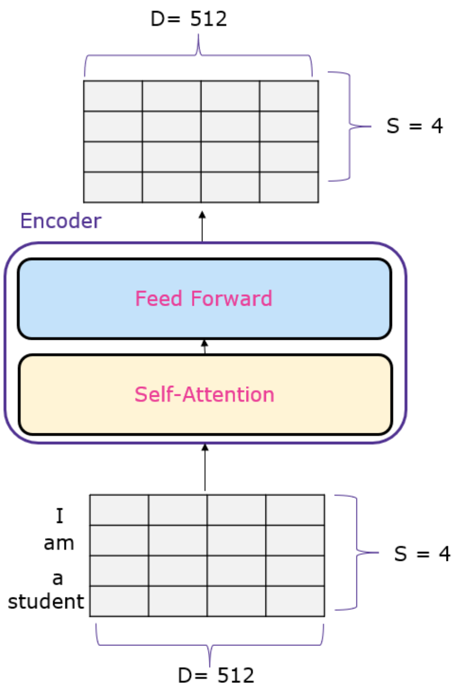


Figure 2.3: Input and Output representation in one encoder layer based on [1].

2.4.2.2 *Decoder*

Given these contextualized representations provided by the encoder component, the decoder component generates an output sequence one element at a time. By consuming the previously generated element as additional input when generating the next, the model is characterized as auto-regressive. [37]. Similar to the encoder, the decoder consists of multiple layers. Each decoder layer has three sub-layers: a self-attention layer, a position-wise feed-forward neural network, and additionally an ecoder-decoder attention layer, that helps the decoder to focus on relevant parts of the input sentence. The position-wise feed-forward neural network in the decoder performs the same function as in the encoder, capturing local interactions. The figures 2.4 and 2.5 provide a high-level visualization of the encoder-decoder structure.

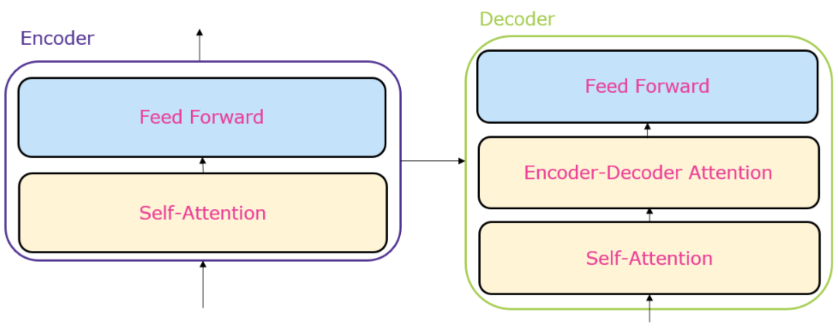


Figure 2.4: One encoder-decoder layer based on [2].

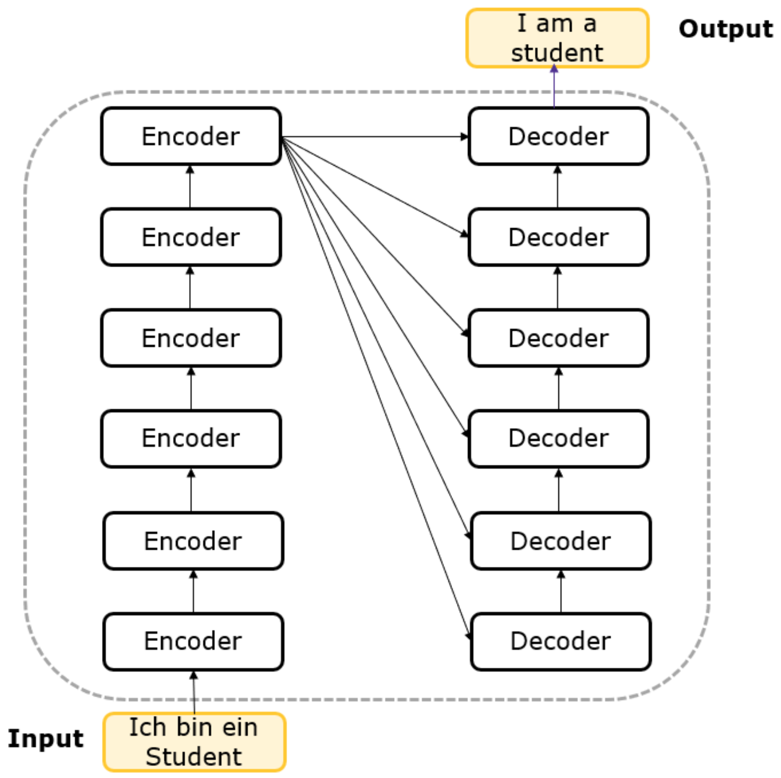


Figure 2.5: Data flow between the encoder and the decoder based on [2].

2.4.2.3 *Attention*

What makes Transformers so special is the the so called 'attention mechanism'. The concept of 'attention' has its roots in addressing the bottleneck problem encountered in machine translation systems. Recurrent Neural Networks (RNNs) and Long-Short Term Memory (LSTM) models encounter two key challenges. First, they process sequences one step at a time, making it difficult to parallelize and slowing down training. Second, they struggle to capture meaningful relationships between distant elements in a sequence, as their influence diminishes quickly over time steps[37]. Transformers address these challenges by enabling parallel processing and effectively modeling long-term dependencies through self-attention mechanisms. The core idea behind the attention mechanism is to allow a model to dynamically focus on the most relevant parts of the input sequence and therefore represent relations between words in a sentence [12]. It allows the model to focus more on the tokens that are most relevant for a given task, while downplaying the influence of irrelevant token. Therefore, it allows the model to let a sequence learn information about itself. For example let's consider the sentence: 'The cat chased the dog because it was hungry.' 'it' could refer to either the cat or the dog, and understanding the correct reference is essential for comprehension. Self-attention mechanisms enable a model to determine that 'it' refers to 'the cat' or 'the dog' based on the context. It captures the dependencies between words and identifies the relationships. Figure 2.6 visualizes the self-attention functionality.

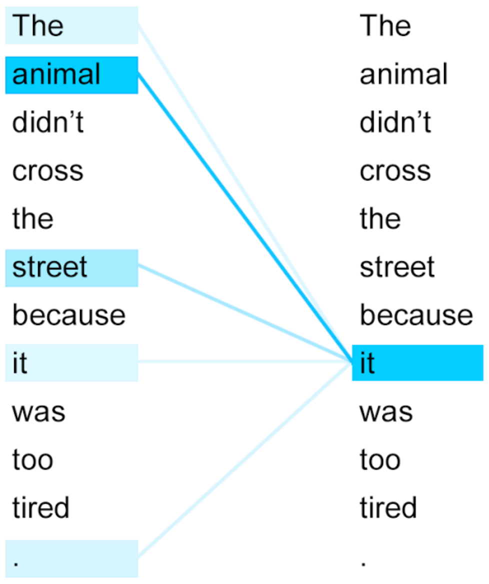


Figure 2.6: Visualization of the self-attention mechanism by [31].

The attention mechanism used by [37] is a dot-product attention, which can be described by the following equation:

$$\text{Attention}(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.3}$$

The equation calculates attention scores (compatibility) between Q and K, scales them, applies the softmax function, and computes a weighted sum using V. Q is a matrix that contains the set of queries packed together, K and V are keys and values matrices. The matrices are calculated by multiplying the input X with weight matrices W_q, W_k, W_v. These weight matrices are first iniliazed and get updated during training.

$$Q = X \cdot W_q \tag{2.4}$$
$$K = X \cdot W_x \tag{2.5}$$
$$V = X \cdot W_v \tag{2.6}$$

2.4.2.4 *Multi-head attention*

To enhance the representational power of self-attention, [37] employed a multi-head attention mechanism, which repeats its computations multiple times in parallel. The attention module splits its Query, Key, and Value parameters N-ways, and each split is passed independently through a separate head shown in 2.7. By using multiple attention heads, each focusing on different parts of the input sequence, the model can capture various types of dependencies and extract different types of information. The outputs from multiple attention heads are concatenated and linearly transformed to obtain the final representation/final attention score.

$$\text{MultiHead}(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{2.7}$$

$$\text{where } head_i = Attention(Q_i, K_i, V_i) \tag{2.8}$$

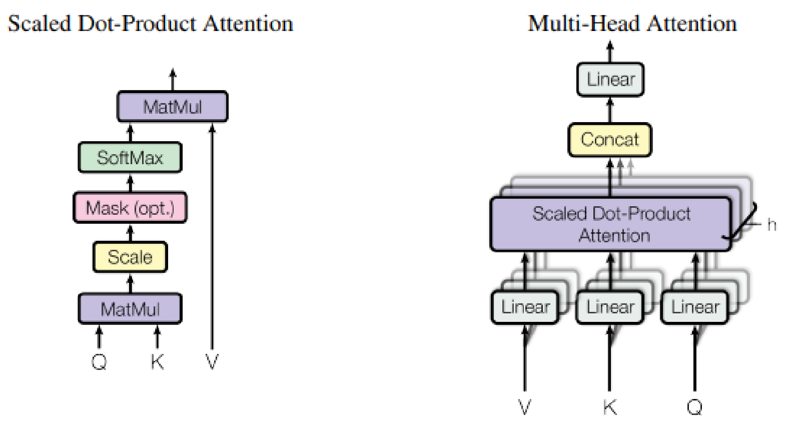


Figure 2.7: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel. Figure was taken from [37].

2.4.2.5 *Positional Encoding*

Positional encoding is another important component in the transformer model. It allows the model to consider the order of words in a sentence. In natural language, the order of words is crucial for understanding the meaning of a sentence. For example, "The cat chases the mouse" has a different meaning than "The mouse chases the cat", even though both sentences use the same words. Transformer models, unlike recurrent neural networks (RNNs), do not inherently consider the order of elements in a sequence. This is because transformers treat each word in the input sequence independently and apply the same operations to each word. Therefore, without positional encoding, a transformer model would not be able to distinguish between "The cat chases the mouse" and "The mouse chases the cat". Positional encoding solves this problem by adding additional information to each word in the input sequence, indicating its position relative to other words. This allows it to consider both the meaning of each word and its position in the sentence when making predictions. It has recently become more common to use relative position embeddings [16]. Raffel et al. (2019) further explain in [30] that instead of using a fixed embedding for each position, relative position embeddings produce a different learned embedding according to the offset between the "key" and "query" being compared in the self-attention mechanism [30].

2.4.3 *BERT*

There are several Large Language Models LLMs, that are built on the Transformer architecture. They differ in terms of whether they employ the encoder, decoder or both components, their training objectives and size. BERT (Bidirectional Encoder Representations from Transformers) is a specific example that employs only the encoder component. BERT was introduced in 2019 and back then one of the early models to leverage the Transformer architecture and demonstrating the power of capturing context in both directions within a sentence [10]. BERT was used in this work to perform the classification tasks and therefore it is important to spend some time exploring the architecture and pre-training objectives.

2.4.3.1 *What is BERT*

BERT is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both the left and right context in all layers. It utilizes MLM (Masked Language Model) and NSP (Next Sentence Prediction) as its pre-training objectives. These objectives are achieved through the incorporation of unique tokens, namely CLS (Classification), SEP (Separator), and MASK (Masked). As a result, Devlin et al. states that the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications [5]. BERT uses a transformer language model designed by [37] at its heart, but only deploys the encoder layers. It typically utilizes a stack of 12 or 24 encoder layers, depending on the specific variant ($BERT_{Base}$ or $BERT_{Large}$). $BERT_{Base}$ consists of 12 encoder layers, while $BERT_{Large}$ employs 24 encoder layers. These layers are responsible for processing and extracting contextual information from the input text. The choice of variant depends on the complexity and scale of the natural language processing task at hand.

2.4.3.2 *Input and Output Representation*

Input Representation: The input to BERT is a sequence of tokens, which is first converted into vectors and then processed in the neural network. The token sequence can represent a single sentence or a pair of sentences (e.g., for tasks that compare two sentences). Each token is embedded into a vector using WordPiece embeddings (a subword tokenizer, developed by Google), with a vocabulary size of 30,000 tokens. To allow the model to distinguish between sentences and to understand the order of sentences, BERT adds two types of embeddings to the token embeddings: segment embeddings and position embeddings. The first token of every sequence is always a special classification token ('[CLS]'). The final input representation for each token is the sum of its token embedding, segment embedding, and position embedding.

Output Representation: The output from BERT is a sequence of hidden state vectors, one for each input token. These vectors capture the contextual information for each input token based on all other tokens in the sequence (to both the left and the right). For tasks that require a single vector (e.g., text classification), the hidden state vector corresponding to the '[CLS]' token is

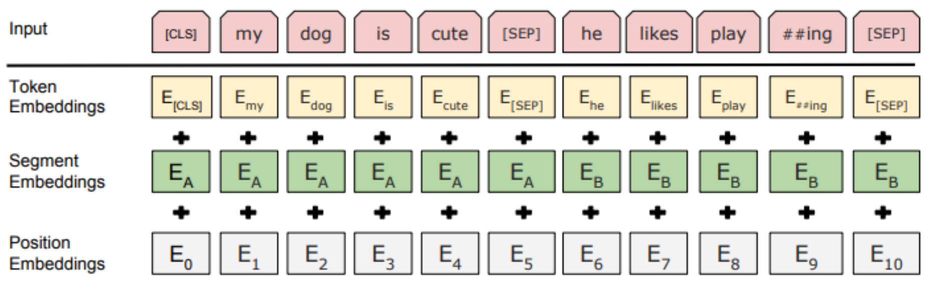

Input
[CLS]
my
dog
is
cute
[SEP]
he
likes
play
##ing
[SEP]
Token Embeddings
$E_{[CLS]}$ E_{my} E_{dog} E_{is} E_{cute} $E_{[SEP]}$ E_{he} E_{likes} E_{play} $E_{\#\#ing}$ $E_{[SEP]}$
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}

Figure 2.8: Overview of the steps involved in converting the input data into a numerical representation. Source: [10]

used. This vector goes through an additional layer of neural network to produce the final task-specific outputs. An overview of the described processes are illustrated in figure 2.8.

2.4.3.3 *Pre-Training objectives*

Pre-training involves training a language model on a massive corpus of unlabeled text to learn contextualized representations of words. This is done through pre-training objectives such as masked language modeling and next sentence prediction. This initial phase allows the model to capture rich semantic relationships and contextual information, providing a foundation for further fine-tuning on specific downstream tasks.

2.4.3.4 *Masekd Language Modeling*

The first pre-trainng objective is called Masked Language Modeling (MLM). In Masked Language Moddeling (MLM), a certain percentage of the input data is masked at random, and the model is trained to predict these masked tokens based on their context. IN figure 2.9 the word 'you' is masked in the sentence 'how are you today' and we want the model to predict it. This allows the model to learn a deep understanding of language structure and semantics [7] [3]. The masking process in BERT is not simply replacing selected tokens with a '[MASK]' token. Instead, 80 percent of the time the selected tokens are replaced with the '[MASK]' token, 10 percent of the time they are replaced with a random token, and 10 percent of the time they are left unchanged. This encourages the model to focus on the entire context rather than relying solely on the '[MASK]' token, making it more robust. The MLM approach enables BERT to learn bidirectional representations, as it has to understand the context on both sides of each token to make accurate predictions12. This is in contrast to traditional language models, which typically operate in a unidirectional manner.

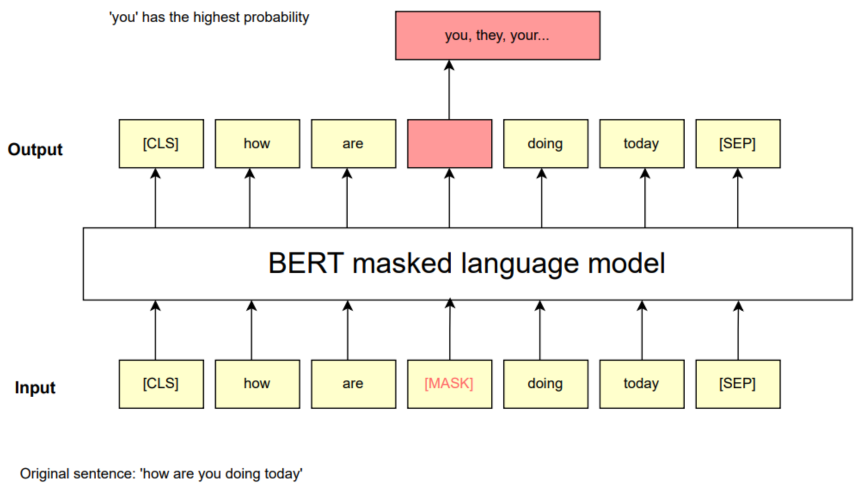

'you' has the highest probability
you, they, your...
Output
[CLS]
how
are
doing
today
[SEP]
BERT masked language model
Input
[CLS]
how
are
[MASK]
doing
today
[SEP]
Original sentence: 'how are you doing today'

Figure 2.9: Pre-training objective 'Masked Language Modeling' for BERT. Based to on [17].

2.4.3.5 *Next Sentence Prediction (NSP)*

Next Sentence Prediction (NSP) is the other important component in the pre-training process. This objective is shown in figure 2.10 and teaches BERT to understand longer-term dependencies across sentences. It involves feeding BERT 'sentence A' and 'sentence B'. Then we ask, 'Hey, BERT, does sentence B follow sentence A?' BERT outputs either IsNextSentence or NotNextSentence. In NSP the special token'[SEP]' is used to indicate where the first sentence ends and the second sentence starts.

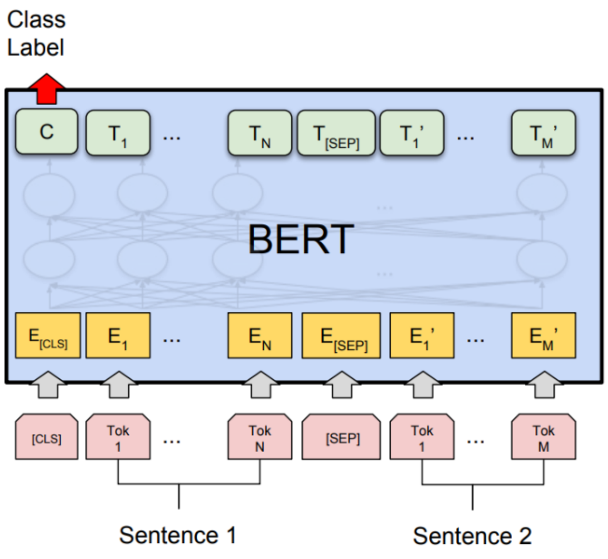


Figure 2.10: NSP Task in Bert [10].

2.4.3.6 *Fine-Tuning*

To apply transformers in different tasks, instead of starting the training process from the beginning each time, a commonly adopted method is to use the pre-trained transformers and subsequently adjust them using a downstream dataset. This step is called fine-tuning. It involves training the model on a specific task (like question answering , translation or summarization) using labeled data. The Transformer's internal weights as well as weights of the newly added classification layer are updated here. This two-step process allows Transformers to leverage both the information available in large text corpora and the specific details contained in task-specific datasets. For the fine-tuning, the BERT model is loaded with the pre-trained weights and further fine-tuned using labeled data in downstream tasks [10]. The pre-training corpus used for pre-training $BERT_{Base}$ and $BERT_{Large}$ was the BooksCorpus from [42] with 800M words and the English Wikipedia corpus with 2500M words.

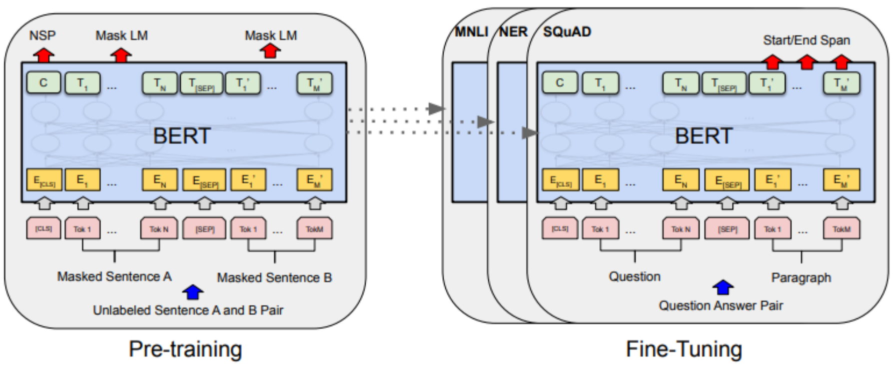


Figure 2.11: BERT is pre-trained with an NSP and MLM training objective and further fine-tuned on a specific downstream dataset like SQuAD for example [10].

2.4.4 *T5 and Flan-T5*

Whereas BERT requires fine-tuning on specific tasks, and different fine-tuned models are often used for those different tasks, T5 was trained to handle multiple tasks simultaneously [44] as illustrated in 2.12. T5 is an open-source, encoder-decoder-based, pre-trained language model, developed by Google, that reframes all NLP tasks into a unified text-to-text format where the input and output are always text strings [30]. This method is called 'prompting'. A prompt/input is a human-like message that instructs the model about the specific task to perform. The prompt typically includes a task description or directive that guides the model in generating the desired output. The flexibility of T5 allows it to handle various natural language processing tasks by framing them as text generation problems, with the input prompt providing the necessary context for the model to produce the corresponding output. For example, if you were to input a text into T5 and instruct it to "summarize this text," it would understand that it needs to generate a shorter version of the same text. T5 was trained in various settings using a mixture of unsupervised and supervised tasks and can be downloaded from Hugging Face. Google Research claims that T5 is flexible enough to be easily modified for application to many tasks beyond those considered in [30]. Fine-tuned Language Net - Text-To-Text Transfer Transformer (FLAN-T5) is an enhanced version of Google's T5 AI model. It was released in the paper "Scaling Instruction-Finetuned Language Models" and has been fine-tuned on more than 1000 additional tasks. Flan-T5 is introduced here, because it can be used for the span/opinion extraction task via a Question-Answering approach. Question-Answering is an important task that deals with the challenge to understand user queries in natural language and deliver accurate responses [20]. Question-Answering (QA) aims to simplify the human-machine interaction and was used in several products as AI assistant like in Amazon Alexa [1], Google Assistent [2] and Apple Siri [3]. FLAN-T5 was trained on the The Stanford Question Answering Dataset (SQuAD) dataset, which contains many paragraphs of text, different questions related to the paragraphs, their answers, and the start index of answers in the paragraph. Because in the Question-Answering setting, the model extracts the answer from the context, the task is called 'Extractive QA'. Generative Pre-trained Transformer 3.5 (GPT-3.5) in contrast performs a generative QA task. Both approaches will be tested and compared with each other.

1 Amazon Alexa: $https://alexa.amazon.de/$
2 Google Assistent: $https://assistant.google.com/intl/de_{d}e/$
3 Apple Siri: $https://www.apple.com/de/siri/$

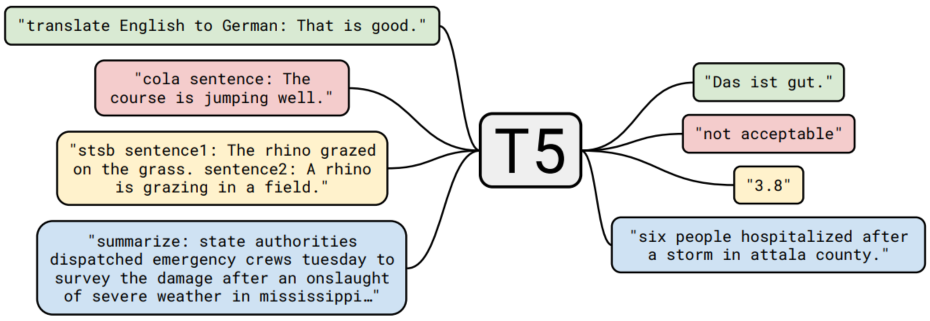

"translate English to German: That is good."
"cola sentence: The course is jumping well."
"stsb sentence1: The rhino grazed on the grass. sentence2: A rhino is grazing in a field."
"summarize: state authorities dispatched emergency crews tuesday to survey the damage after an onslaught of severe weather in mississippi…"
T5
"Das ist gut."
"not acceptable"
"3.8"
"six people hospitalized after a storm in attala county."

Figure 2.12: Illustration of the different tasks, including translation, question answering, and classification the Text-to-Text-Transformer Model is trained on. Figure based on [30].

2.4.5 *GPT*

2.4.5.1 *Evolution of the GPT model series*

A model that has revolutionized the field of NLP is GPT-3.5 turbo, also known as ChatGPT. It was developed by OpenAI and released in 2022 [4]. Models of the GPT model series rapidly evolved in different versions, being trained on a larger corpus of textual data and with a growing number of parameters. This development led to a number of milestones presented in figure 2.13. With each version, the understanding and generation capabilities improved, and with the release of GPT-3 in-context learning approaches could be leveraged [20]. The third version of GPT, with 175 billion parameters, is 100 times bigger than GPT-2. The premise is, that a higher number of parameters leads to better model performance. Brown et al. demonstrate, that scaling up language models greatly improves task-agnostic, few-shot performance, sometimes even reaching competitiveness with prior state-of-the-art fine-tuning approaches [5]. However, GPT-3 is not chat-optimized, which led to the development of GPT-3.5 turbo and its release in November 2022. The latest model of the GPT model series is GPT-4, with its ability to handle image data as input [20]. Besides the number of parameters and the amount of training data, there is also evidence that the data quality is a factor here [23]. OpenAI has made the model available to the public through a playground [27] and an Application Programming Interface (API). GPT-3.5 turbo can generate text that is appropriate for a wide range of contexts, but unfortunately, often expresses unintended behaviors such as making up facts, generating biased text, or simply not following user instructions [28], which can undermine the potential of real-world applications.

4 https://openai.com/blog/gpt-3-5-turbo-fine-tuning-and-api-updates

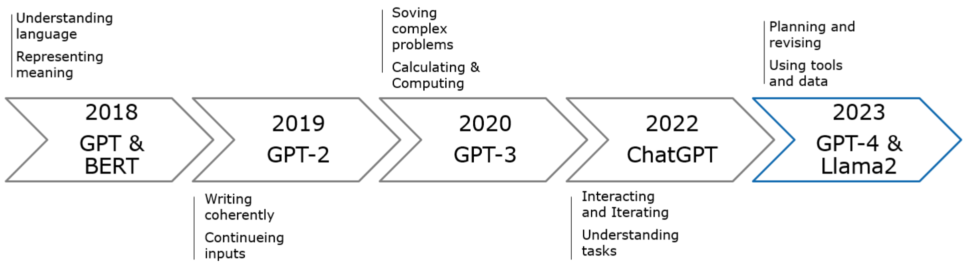


Figure 2.13: Important steps in the evolution of large language models.

2.4.5.2 *Few-Shot Learning*

LLMs are pre-trained on an enormous amount of data and hold a huge general knowledge in their parameter weights. Updating these weights via fine-tuning can be computationally very expensive. We can make use of the model's existing knowledge by providing the model a context with some instructions and examples at inference time. The objective is to manipulate the the input prompt in such a way, that the probability distribution of the next-tokens, that the model predicts, matches the intentions. This process is called 'Prompting' or 'In-Context-Learning'. It has been a new trend to explore In-Context-Learning (ICL) to evaluate and extrapolate the ability of LLMs [11].

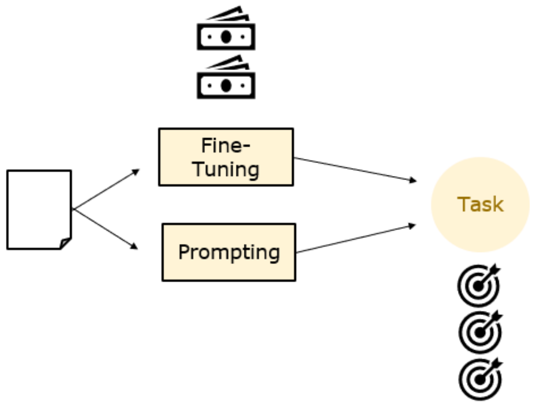
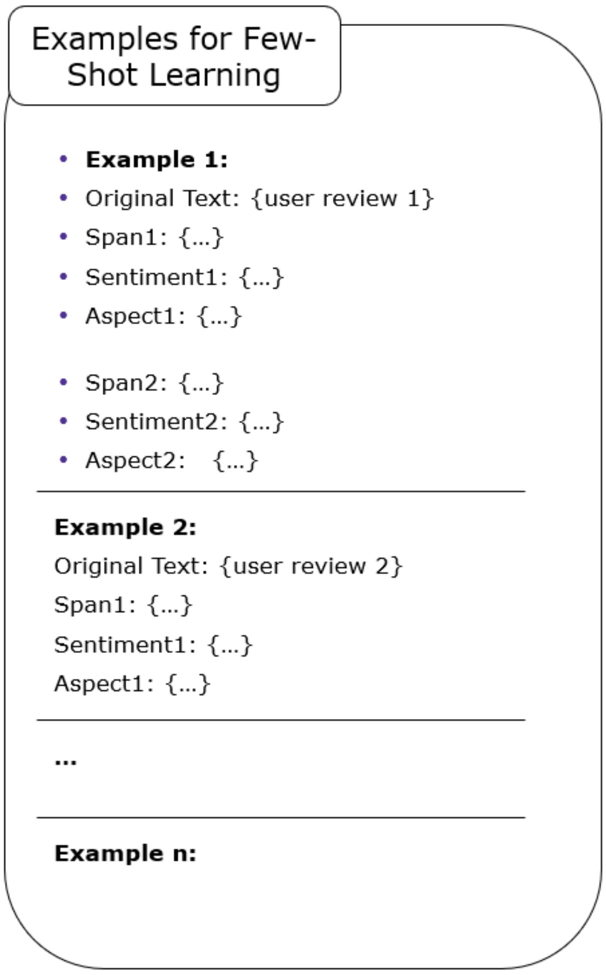


Figure 2.14: Fine-Tuning LLMs is usually more expensive than using Prompts

The key idea is to learn from analogy. The model is 'trained' on a small number of examples (more than one but still relatively few and depending on how many examples fit into the model's context window) for a given task. Brown et al. demonstrated in [5] the effectiveness of modulating a frozen GPT-3 model's behavior through text prompts. The approach of 'freezing' pre-trained models is appealing, especially as model size continues to increase. Instead of having a separate copy of the model for each downstream task, a single generalist model can simultaneously serve many different tasks. Prompt Engineering becomes an increasingly important skill to use LLMs in an effective way. White et al. (2023) state that prompts are not only instructions but also a form of 'programming' that can customize the outputs and interactions with an LLM[38]. He even compares prompts with software patterns that can be reused to solve recurring problems. He introduces 16 prompt patterns for more efficient interaction between the user and the LLM, specifically ChatGPT. We use LangChain to create dynamic prompts. To use the GPT-3.5-turbo-16k model, Merck provided an OpenAI API key. We select a few examples from our set of labeled reviews and format them in the way shown in figure 2.15.



Examples for Few-Shot Learning

- **Example 1:**
- Original Text: {user review 1}
- Span1: {...}
- Sentiment1: {...}
- Aspect1: {...}

- Span2: {...}
- Sentiment2: {...}
- Aspect2: {...}

Example 2:
Original Text: {user review 2}
Span1: {...}
Sentiment1: {...}
Aspect1: {...}

...

Example n:

Figure 2.15: Illustration of a .txt file with examples for few-shot learning.

An instance for the Few-Shot approach could look like in the example below. It shows, how we can reduce a text to it's relevant parts by extracting only spans.

Original text: Ellenn im at a private clinic but nhs funded. In Nottingham they outsource ivf to private clinics - they only do OI in the nhs hospital. We were self funding at our clinic before anyway for OI, and now switched to our funding now it's cleared. For ohss I think their process is only FA if someone is in ohss but defo will ask. Thank you! Ovaleap is a lot better and don't have to mix like Menopur. No same dose. Continue as I was as size wise they're not there yet. Collection Monday apparently but will see what's happening tomorrow in my next scan. remind me - this your first round or second?

Span1: Ovaleap is a lot better and don't have to mix like Menopur
Aspect1: positive
Sentiment1: administration

3 METHODOLOGY

In this chapter, we will explain how we conducted our empirical study. We'll talk about how we created the datasets, breaking down the process into manual and automatic labeling. We will also discuss the metrics we used to measure the quality of the automatically labeled dataset. After that, we will move on to fine-tuning the BERT model and check how well it performs in both sentiment prediction and aspect prediction tasks.

3.1 PIPELINE

The raw data is provided in the form of MS Excel files which the stakeholder, a representative of Merck Healthcare, exported from a Sprinklr database. It contains a significant amount of noise, which needs to be removed first before we can proceed with the actual labeling process. Therefore, we have to apply some data preparation steps to ensure that the data we label is of high quality in order to fine-tune BERT. The pipeline is illustrated in figure 3.2.

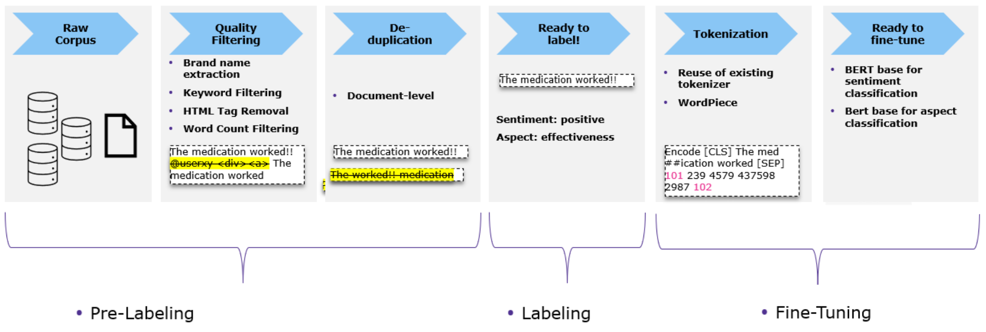


Figure 3.1: Pipeline with an overview of important steps involved in this thesis: Pre-Labeling, Labeling and Fine-Tuning.

3.2 CREATION OF A DATASET

In total, we manually label a set of 400 records by extracting the opinions and assigning an aspect category and a sentiment to it. We label the same amount of records in an automated manner by using a combination of GPT-3.5 turbo and a Few-Shot Learning approach. Here we define a prompt and modify it until the model outputs are stable. The labeling process is shown in figure 3.2

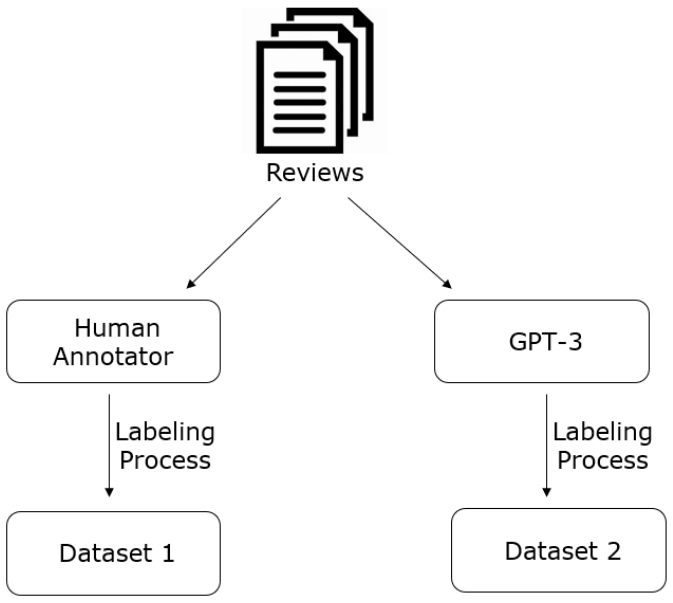


Figure 3.2: The two different labeling methods used to generate our datasets: human labeling and automated labeling with GPT-3.5 turbo.

3.2.1 *Annotation Guideline*

We begin with the definition of an annotation guideline, designed to ensure consistent human labeling. This guideline was created in close collaboration with the business stakeholders. It contains information like what should be labeled and what should not be labeled and therefore reduces conflicts during the annotation process.

3.2.2 *Pre-Labeling Steps*

The most important steps in the pre-labeling phase are further explained in the following:

- **Brand Name Extraction:** Each MS Excel file contains multiple sheets, each one for a specific medication. We have chosen a subset of these sheets based on our specific medication brands of interest:
  - Menopur
  - Gonal-F
  - Decapeptyl
  - Ovaleap
  - Puregon
  - Fostimon
  - Bemfola

- Not every review contains an explicit mention of the brand stated above. We systematically applied a case-insensitive Regular Expression (Regex) pattern across all reviews, filtering and retaining only those that contained the specified brands. This ensures that the dataset is centered around those brands, which is a requirement of the stakeholder.

Listing 3.1: Python Code for filtering data based on a regex pattern

```
regex_pattern = 'Menopur|Gonal|Ovaleap|bemfola|Pergoveris|
    puregon|Fostimon|Decapeptyl'

def filter_by_regex(df, regex_pattern):
   df = df[df['text'].str.contains(regex_pattern, flags=re.
       IGNORECASE, regex=True)]
  return df
```

  The pipe ('|') symbol acts as an "OR" operator, meaning the pattern will match any of these words. The re.IGNORECASE flag makes the pattern case-insensitive, so it will match regardless of the letter casing in the target text.

- **Irrelevant Data Filtering:** The selected drugs can not only be used in the fertility context, but also in the context of cancer treatment. For example, Decapeptyl can be used as part of a broader treatment plant in radiotherapy for certain types of cancers. Decapeptyl is a gonadotropin-releasing hormone (GnRH) agonist, and it works by suppressing the production of sex hormones. In the context of cancer treatment, it may be used to temporarily suppress the production of hormones such as estrogen or testosterone, which can be beneficial in certain cancers that are hormone-sensitive like prostate- or breast cancer. Decapeptyl may be used to reduce testosterone or estrogen levels, therefore slowing down the growth of the cancer cells[36]. The issue is, that cancer patients may talk about different topics and certain aspects of the drug such as 'effectiveness' for example is interpreted differently by them than by a patient who uses the drug for fertility reasons. To center our dataset around the fertility topic, we filter out the topics that are not about fertility by using a regex pattern with the following keywords: cancer, tumor, oncology, and endometriosis.

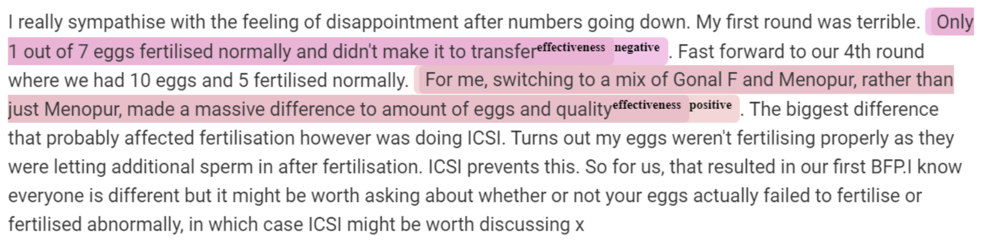
- **HTML Tag Removal:** We then remove any HTML tags present in the data by using the library 'BeautifulSoup'. This is crucial as these tags can interfere with the text analysis and do not provide meaningful information for our task.

- **Word Count Filtering:** We filter out data that contains less than 5 words or more than 350 words. This helps us focus on texts that are likely to provide substantial and relevant information. For example there are reviews that contain more than 1000 words, which most of the time, are copy-paste Wikipedia articles and not actual user reviews.

- **Duplicate Removal:** Finally, we remove any duplicate entries in our dataset. We perform a hard-duplucate removal which focuses on exact matches between records. This ensures that each piece of data we analyze is unique, improving the quality of the data.

3.2.3 *Manual Labeling*

The manual annotation is then carried out in the labeling framework 'NLP Labs' [21]. It is important to note, that some reviews can have opinions about multiple aspects and express different sentiments towards each of them like in figure 3.3. Here, the patient discusses the ineffectiveness of the initial medication. However, upon switching to a different brand, they experienced positive results. Consequently, we encounter the aspect of "effectiveness" twice in this context, each associated with a distinct sentiment—negative and positive, respectively.



I really sympathise with the feeling of disappointment after numbers going down. My first round was terrible. Only 1 out of 7 eggs fertilised normally and didn't make it to transfer effectiveness negative. Fast forward to our 4th round where we had 10 eggs and 5 fertilised normally. For me, switching to a mix of Gonal F and Menopur, rather than just Menopur, made a massive difference to amount of eggs and quality effectiveness positive. The biggest difference that probably affected fertilisation however was doing ICSI. Turns out my eggs weren't fertilising properly as they were letting additional sperm in after fertilisation. ICSI prevents this. So for us, that resulted in our first BFP.I know everyone is different but it might be worth asking about whether or not your eggs actually failed to fertilise or fertilised abnormally, in which case ICSI might be worth discussing x

Figure 3.3: Review with two relevant text spans. Each span is extracted separately and treated independently in the dataset.

Note: Some reviews might not contain any relevant opinions about our predefines aspects at all. In this case, nothing should be labeled here.

3.2.4 *Challenges and Limitations during Labeling*

There were several issues during the annotation process, partly related to the data itself and partly associated with the NLP Lab labeling tool 'JohnSnow Lab' [21]. Starting with the data, the initial amount of raw data was drastically reduced after applying the data preprocessing steps. As only the stakeholder had access to the Sprinklr database and thus control over the data volume, she had to be contacted repeatedly to provide more data. The MS Excel files had a significant overlap of data, resulting in a high number of duplicates that needed to be removed. Consequently, the quantity of labeled data increased only very slowly. Another challenge in annotating reviews is figuring out which parts of the texts are important and which aren't due to the lack of clear text boundaries. This requires reading carefully the reviews multiple times before annotating anything, which is also very timeconsuming. Additionally, users often share extensive and irrelevant drugrelated histories, such as experiences from a decade ago. Unfortunately, prefiltering this irrelevant information is not feasible and the process of reading through entire reviews can be exhausting, potentially leading to mistakes in annotation and impacting the accuracy of classification models. Also, the annotation tool has caused some difficulties. For example, there was only a "next" button and no "back" button. So, if you wanted to revisit a previous review to make some changes, you could only do so after navigating through the tool, which was a time-intensive process. Another issue was, that after a data record was labeled and submitted, the labeling tool didn't allow any changes, resulting in time-consuming manual post-processing efforts. Also, the order of the labeling sequence, whether starting with sentiment or aspect first, is crucial, as inconsistency can lead to intricate post-processing complexities as well.

3.2.5 *Automated Labeling with GPT-3 and Few-Shot Learning*

In this data labeling approach, Few-Shot learning is leveraged to annotate user reviews in the context of aspect-based sentiment analysis. We will give the model a few examples (here we used 10 examples) and a context, which includes descriptions about the aspects and sentiment. Both components will be chained together by using 'Prompt Templates' which are provided by the library 'LangChain' [29]. With our prompt, we can guide the model in extracting specific information related to fertility drugs.

3.2.6 *Final Datasets*

Both labeling approaches resulted in the following data distributions 3.4, 3.5, 3.6, 3.7.

First we compare the sentiments labels generated by a human and by GPT-3.5 turbo. First we notice, that GPT-3.5 turbo extracted in total 220 more opinions and therefore we got the same number of additional labels. Secondly, we notice an imbalance in both datasets. The negative class is overrepresented here, which GPT-3.5 turbo seemed to have recognized correctly.

Now we take a look at the aspect categories annotated by a human and GPT-3.5 turbo. We also have an imbalanced dataset here. In the ground truth dataset, the category 'effectiveness' has the highest number of samples, followed by 'adverse event'. Also GPT-3.5 turbo captured this imbalance, but compared to the ground truth dataset recognized more samples of 'administration'. The categories 'cost' and 'availability' seemed to have been captured correctly by GPT-3.5 turbo.

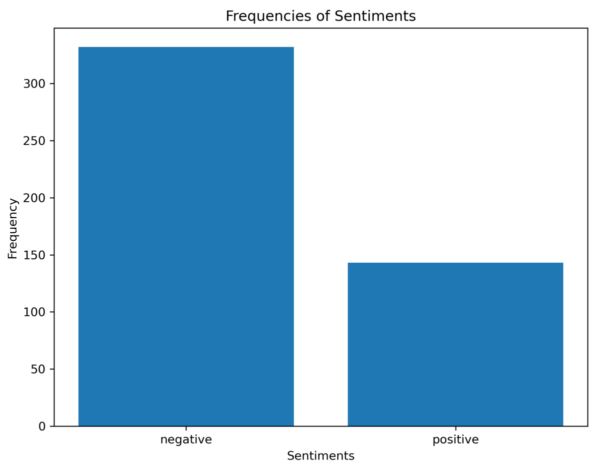


Figure 3.4: Count of negative and positive instances annotated by a human.

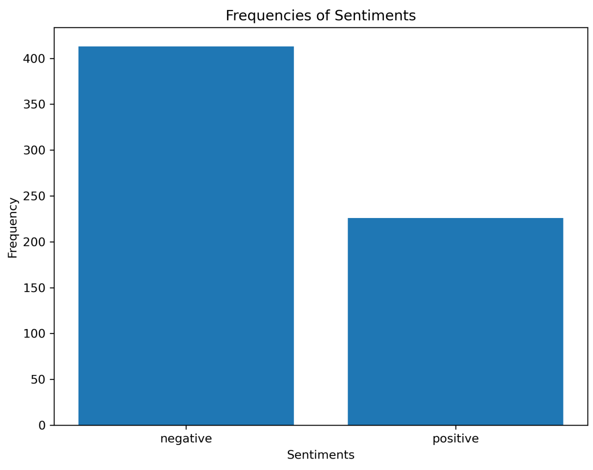


Figure 3.5: Count of negative and positive instances annotated by GPT-3.5 turbo.

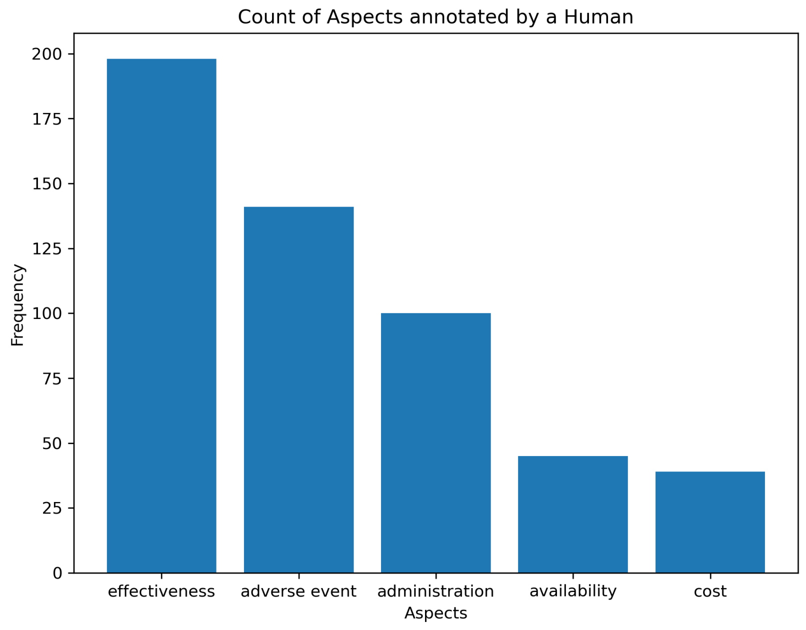


Figure 3.6: Counts of aspects annotated by a human.

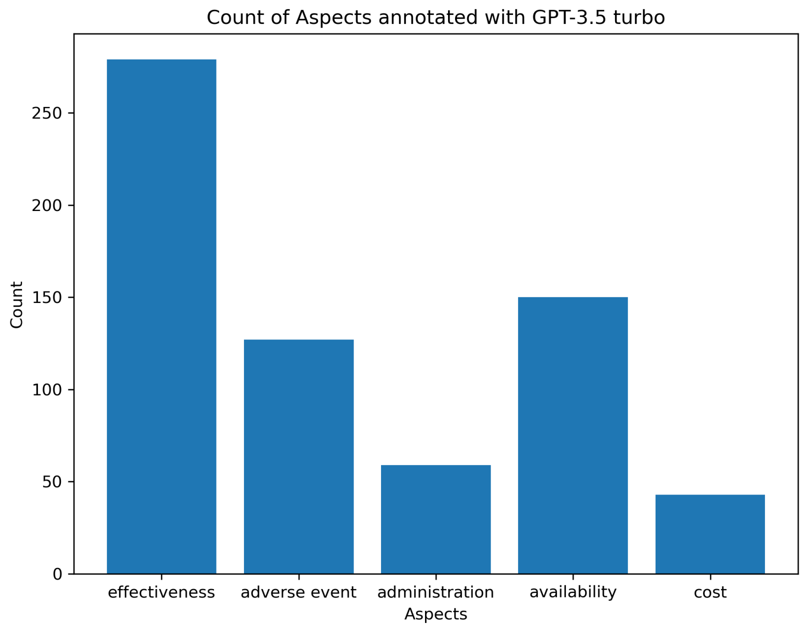


Figure 3.7: Counts of aspects annotated by GPT-3.5 turbo.

3.3 BASELINE AND EVALUATION

First, we want to evaluate the quality of the predicted text-spans by GPT-3.5 turbo and FLAN-T5. With texts presented as vectors, we measure the degree of similarity of two texts as the correlation between their corresponding vectors, which can be further quantified as the cosine of the angle between the two vectors [16]:

$$Cosine(x,y) = \frac{x \cdot y}{|x||y|} \tag{3.1}$$

That being said, we need to calculate an embedding vector for the true text span and the generated text span by FLAN-T5 and GPT-3. OpenAi recommends using text-embedding-ada-002 model for nearly all use cases. To retrieve the embeddings, we send our input to the embeddings API endpoint and get the embedding as a response [26].
Accuracy and F-1 score are two commonly used metrics for evaluating the performance of classification models. Accuracy is the most intuitive performance measure. It is simply a ratio of correctly predicted observations to the total observations. Its score ranges between 0 and 1. It works well only if there are an equal number of samples belonging to each class. The formula for accuracy is:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.2}$$

with TP = *True Positive*, TN = *True Negative*, FP = *False Positive* and FN = *False Negative*. However, accuracy is not a good choice with imbalanced datasets, where some classes have significantly more samples than others. For example in our case, we have more negative opinions than positive ones. In such cases, even a simple model that always predicts the majority class will have a high accuracy rate. The F1-Score was introduced to address this problem. It is a measure of a model's performance that considers both precision and recall. Precision is the number of true positives divided by the number of true positives and false positives. Recall (Sensitivity) is defined as the number of true positives divided by the number of true positives and the number of false negatives. The F1-Score is the harmonic mean of precision and recall, taking both false positives and false negatives into account. It reaches its best value at 1 (perfect precision and recall) and worst at 0 [43]. The formula for the F1-Score is given by:

$$Precision = \frac{TP}{TP + FP} \tag{3.3}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.4}$$

$$F1 - Score = 2\frac{Precision * Recall}{Precision + Recall} \tag{3.5}$$

An alternative approach is using class weights to handle imbalanced datasets. Each class j is assigned a weight w_j based on its frequency in the training dataset [34]. The equation to calculate the class weights w_j is given by:

$$\mathrm{w}_j = \frac{n_{\text{samples}}}{n_{\text{classes}} \times n_{\text{samples,j}}} \tag{3.6}$$

Here n_{samples} is the total number of samples or rows in the dataset, n_{classes} is the total number of unique classes in the target and $n_{\text{samples,j}}$ is the total number of rows of the respective class j.
In general, the less frequent a class j is, the higher its weight w_j will be. During the training process, the model adjusts its parameters to minimize a weighted loss function. The weighted loss gives more emphasis to the errors made on the underrepresented classes. This helps the model to focus more on improving its performance on the minority classes. With the class weights, we can continue to use the accuracy as metric [34].

4 FINE-TUNING BERT FOR PREDICTING THE SENTIMENT AND ASPECT

In the main chapter of this thesis we explain how we generated the data and how we conducted the fine-tuning of BERT for predicting the sentiment and aspects patients expressed towards certain fertility drugs. In the following, we will provide an overview of the models used, the tokenization method, the hyperparameters, the loss-function, and the hardware we used for the training.

4.1 MODEL STRUCTURE

For GPT-3 labeling we use the Azure OpenAI API service and select the gpt-35-turbo-0613 model. It has 6 billion parameters and is smaller than the DaVinci model, which has 175 billion parameters, but is less expensive. For the classification tasks we use BERT base which is a Transformer model with 12 encoder layers, 768 hidden size and 12 attention heads. The fine-tuning codes are mainly based on Pytorch and the Hugging Face Transformer library [45]. While BERT is traditionally used for binary-label classification tasks, it can be modified for multi-label tasks by changing the loss function used during training.

4.2 HARDWARE

We fine-tuned BERT on the hardware components shown in table 4.1

Component	**Specification**
GPU	Tesla V100-SXM2
Notebook Instance	
- Volume Size	150GB EBS (Elastic Block Store)
- Instance Type	ml.p3.2xlarge

Table 4.1: Specification of hardware components to fine-tune the used BERT model.

4.3 TOKENIZER

Before the actual fine-tuning can take place, we use the existing Tokenizer ’WordPiece’ by Google, which is a subword segmentation algorithm [39]. It basically converts the input data into an appropriate format so that each sentence can be fed to the pre-trained BERT model to obtain the corresponding embedding [40].

4.4 LEARING RATE AND OPTIMIZER

Sun et al. [35] suggest to use a lower learning rate to make BERT overcome the catastrophic forgetting problem, where the pre-trained knowledge is lost during the learning of new knowledge. Here, we use an Adam Optimizer with a learning rate lr = 1e-5.

4.5 EPOCHS

The term "epochs" refers to the number of times the entire dataset is passed through the training process. Selecting an appropriate number of epochs is crucial, as too few may result in underfitting, while too many may lead to overfitting. We chose 30 Epochs.

4.6 BATCH SIZE

The batch size, set to 32 in our configuration, defines the number of data samples processed in a single iteration during training, impacting both memory efficiency and the stability of the training process.

4.7 LOSS FUNCTION

We are using the BCEWithLogitsLoss (Binary Cross Entropy with Logits Loss) which is implemented by PyTorch [8]. Multi-label classification can be understood as a series of binary classifications like ’Is sample 1 in class A – yes or no? Is sample 1 in class B – yes or no?’ And so on. This allows for the model to predict multiple classes for each input. This loss function is beneficial for BERT classification tasks because it integrates seamlessly with the model’s architecture, producing logits as an output. Moreover, BCEWithLogitsLoss is highly efficient and numerically stable, making it a popular choice for training models like BERT in binary classification scenarios. BCEWithLogitsLoss can be calculated through:

$$\text{BCEwithLogitsLoss}(\hat{y}, y) = -[y \log(\hat{y}) + (1-y)\log(1-\hat{y})] \tag{4.1}$$

Here $\hat{y}$ represents the predicted probabiity and y is the ground truth label.

4.8 EXTRACTING EVALUATING THE RELEVANT TEXT SPANS

We will now investigate two approaches to extract the opinions/text spans of a given review, with FLAN-T5 and GPT-3.5 turbo. The extracted spans are treated independently from each other, meaning that each span will get an own row in the resulting dataset without any reference to it's original review. Afterwards, for each span we assign an aspect category and a sentiment.

4.8.1 *Flan T-5*

First, we want to analyze how well FLAN-T5$_{large}$ can extract the opinions by solving a Question-Answering Problem. We have a systematic approach to take advantage of the model's ability to perform Question-Answering tasks. First, formulate a clear and concise question that contains the information we are looking for. Then, we provide the context, which are the user reviews. It is important to ensure that the query and context combination matches the maximum Flan T5 token limit of 512 tokens. Once we have prepared the input, Flan T5 will generate a response, trying to answer the given question based on the given context. Finally, we check the responses made and adjust the question, if the results are not satisfying.
The formulated question reacts very sensitively to minor changes, meaning that small adjustments in the prompt/question can lead to significantly different outputs. We manually make adjustments by "Try and Error" until the responses look okay. The final question is formulated as follows: What does the patient say about the emotional experience with the fertility drug?" We then evaluate the responses by calculating the cosine similarity between the model's answer and the true answers. We first notice, that FLAN-T5 only returns one exact answer for a given context. This doesn't match our requirement of extracting multiple answers/opinions per a given context. Then we compare the extracted spans with the true spans. At first sight, the extracted spans don't seem to match the true spans. By calculating the cosine similarly and getting a value of 0.07, we get a confirmation for that. Therefore, we will not move one with these spans and instead use GPT-3.5 turbo for extracting the opinions.

4.8.2 *GPT-3 and Few-Shot Learning*

As discussed in 3.2.5, we use a few-shot learning approach by given 10 examples to the model. We make sure, that each aspect category is represented at least one time. The final prompt was relatively long, because it included information for each aspect category and the definitions of what a 'relevant' span is, that needs to be extracted. The process of extracting spans posed a challenge, as we noticed that GPT-3.5 turbo tends to extract more spans per review. This resulted in spans not being in the same order, making it difficult to compute cosine similarity for each span directly. To address this, we decided to combine all the spans for a specific review and then calculate the cosine similarity for the overall lists of spans. This approach helps to overcome the variations in the span extraction order, allowing for a more simplified comparison. The process of the evaluation is visualized in 4.1.

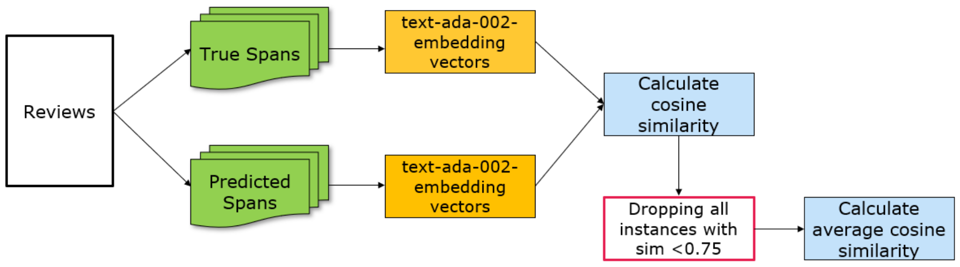

Reviews
True Spans
Predicted Spans
text-ada-002-embedding vectors
text-ada-002-embedding vectors
Calculate cosine similarity
Dropping all instances with sim <0.75
Calculate average cosine similarity

Figure 4.1: Evaluation process to measure the quality of extracted spans by GPT-3.5 turbo

We first concatenate the spans for each review:

Predicted Spans: [Span1, Span2, Span3...]
True Spans: [Span1, Span2, Span3...]

Then, we calculate the embedding for each list using OpenAI's text-embedding-ada-002-v2 embedding model, which is an advanced embedding model that represents large texts as a 1536-dimensional vector [13], [15].

embedding 1 = embed(Predicted Spans)
embedding 2 = embed(True Spans)

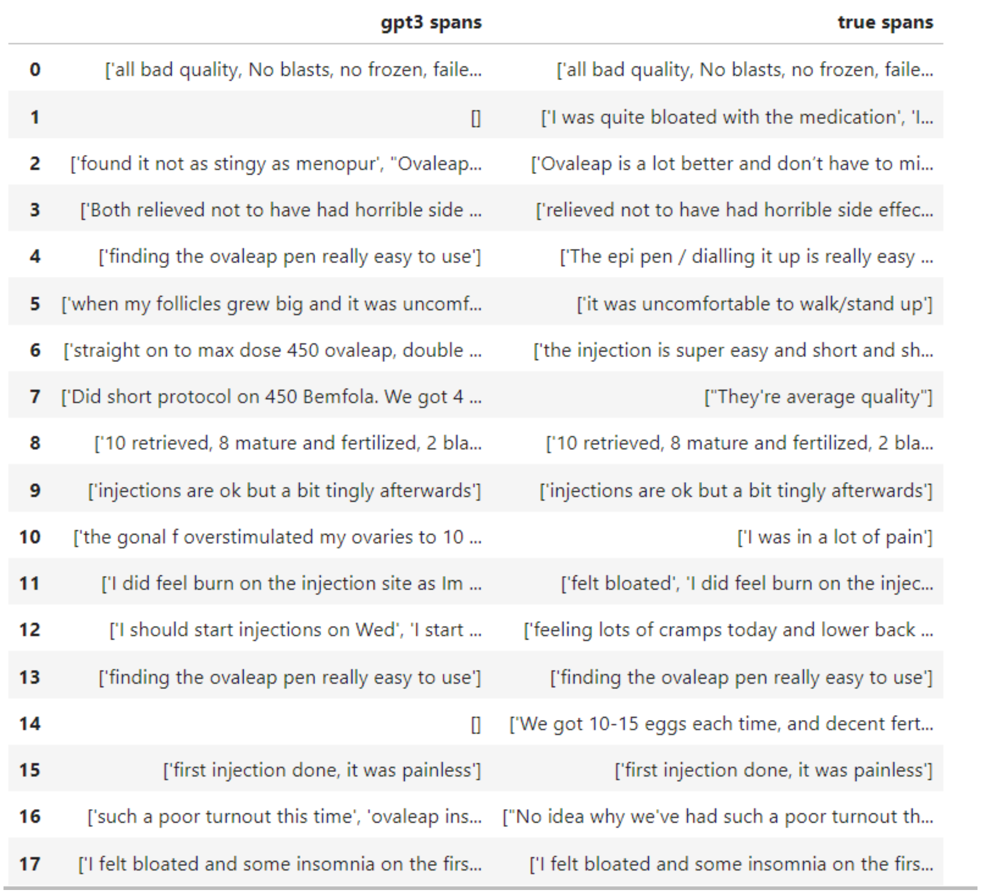
Finally, we calculate the cosinus similarity between both lists:

cosine similarity = np.dot(embedding1, embedding2)

In some instances, GPT-3.5 turbo doesn't extract any opinions at all. The error we get points to Azure OpenAI's policy and that the provided prompt does not align with it. Therefore we end up in the situation, that we have zero predicted spans for some reviews. Despite this, a similarity score between 0.6 and 0.75 is assigned to these cases, which actually should be 0 because if we compare an empty list with a non-empty list, we don't expect any similarity ar all. To address this anomaly, we implement a solution by setting all similarity scores below 0.75 to 0. This correction is crucial in ensuring a more accurate evaluation. Finally, the average cosine similarty between the GPT-3.5 turbo extracted labels and the true labels is calculated and returns a score of 0.66.

Average Cosine Similarity for Spans	0.66

Table 4.2: Opinion extraction results



	gpt3 spans	true spans
0	['all bad quality, No blasts, no frozen, faile...	['all bad quality, No blasts, no frozen, faile...
1	[]	['I was quite bloated with the medication', 'I...
2	['found it not as stingy as menopur', "Ovaleap...	['Ovaleap is a lot better and don't have to mi...
3	['Both relieved not to have had horrible side ...	['relieved not to have had horrible side effec...
4	['finding the ovaleap pen really easy to use']	['The epi pen / dialling it up is really easy ...
5	['when my follicles grew big and it was uncomf...	['it was uncomfortable to walk/stand up']
6	['straight on to max dose 450 ovaleap, double ...	['the injection is super easy and short and sh...
7	['Did short protocol on 450 Bemfola. We got 4 ...	["They're average quality"]
8	['10 retrieved, 8 mature and fertilized, 2 bla...	['10 retrieved, 8 mature and fertilized, 2 bla...
9	['injections are ok but a bit tingly afterwards']	['injections are ok but a bit tingly afterwards']
10	['the gonal f overstimulated my ovaries to 10 ...	['I was in a lot of pain']
11	['I did feel burn on the injection site as Im ...	['felt bloated', 'I did feel burn on the injec...
12	['I should start injections on Wed', 'I start ...	['feeling lots of cramps today and lower back ...
13	['finding the ovaleap pen really easy to use']	['finding the ovaleap pen really easy to use']
14	[]	['We got 10-15 eggs each time, and decent fert...
15	['first injection done, it was painless']	['first injection done, it was painless']
16	['such a poor turnout this time', 'ovaleap ins...	["No idea why we've had such a poor turnout th...
17	['I felt bloated and some insomnia on the firs...	['I felt bloated and some insomnia on the firs...

Figure 4.2: Comparison between spans (opinions) extracted by GPT-3.5 turbo and by a human

4.9 CLASSIFICATION

After the data has been annotated, we can now move forward with the classification task. We split the 400 annotated records into a train- and validation set (80/20 split). Additionally, we keep a separate human-annotated test set with 80 records frozen to prevent data leakage. The process is shown in figure 4.3. The results can be retrieved from table 4.3. A detailed discussion will take place in section 4.10.

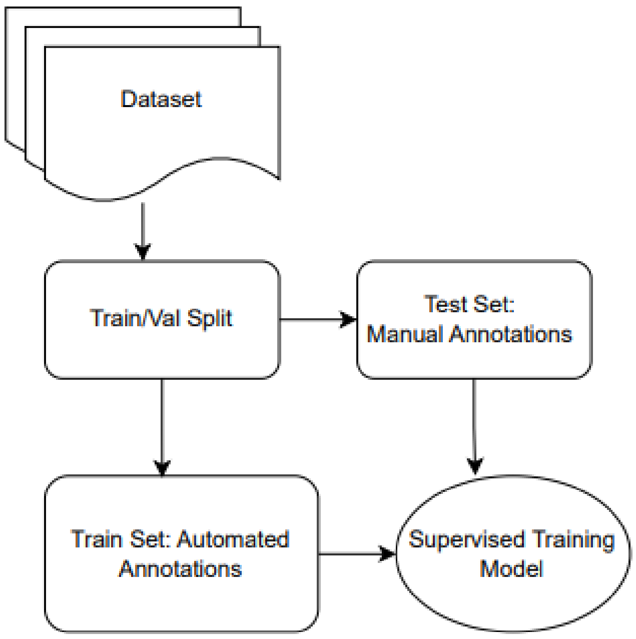

Dataset
Train/Val Split
Test Set: Manual Annotations
Train Set: Automated Annotations
Supervised Training Model

Figure 4.3: Train/Test Split

4.9.1 *Aspect Classification*

In the following we will discuss the aspect classification process. For this classification we will use the human-annotated and the GPT-3.5 turbo annotated dataset. With each of them, we will fine-tune two BERT models. We use the base-uncased version with a dropout of 0.3 and 5 output nodes in the linear classification head. We see the performance of the Aspect-Classification models in figure 4.4 and 4.5. The blue curve represents the training accuracy, and the orange curve the validation accuracy. We chose 30 epochs, which led to overfitting in both cases. An optimal number of epochs seems to be between 17 and 20.

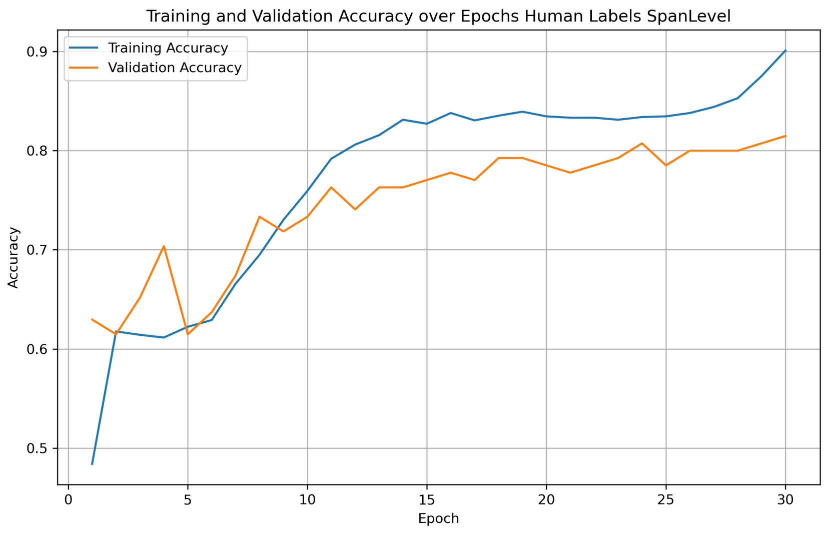


Figure 4.4: Performance of Aspect Classification with human annotated data

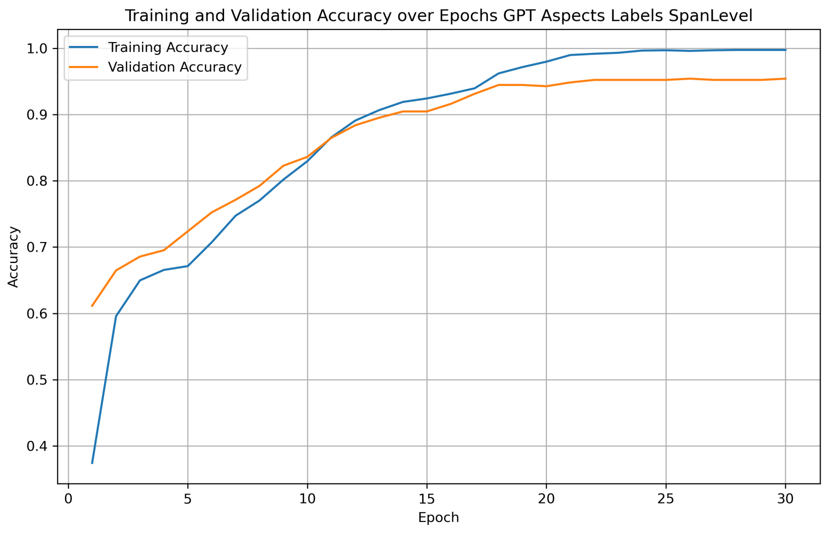


Figure 4.5: Performance Aspect Classification with GPT-3.5 turbo annotated

4.9.2 *Sentiment Classification*

Now we train a BERT model for sentiment classification in the same manner. We only change the number of output nodes from 5 to 2 and use the sentiment labels as targets. We observe, that overfitting takes place earlier, namely between 8 and 10 Epochs. The performance per epoch is shown for each model in figure 4.6 and in figure 4.7.

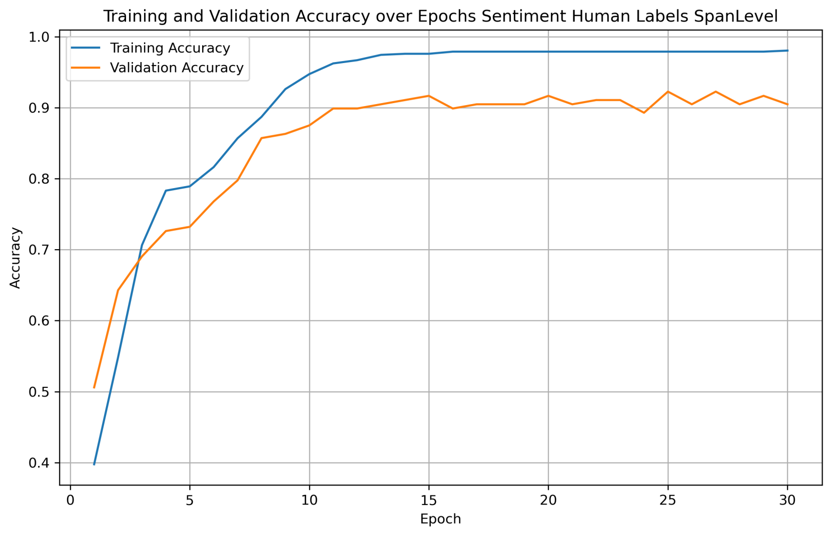


Figure 4.6: Performance of Sentiment Classification with Human annotated data.

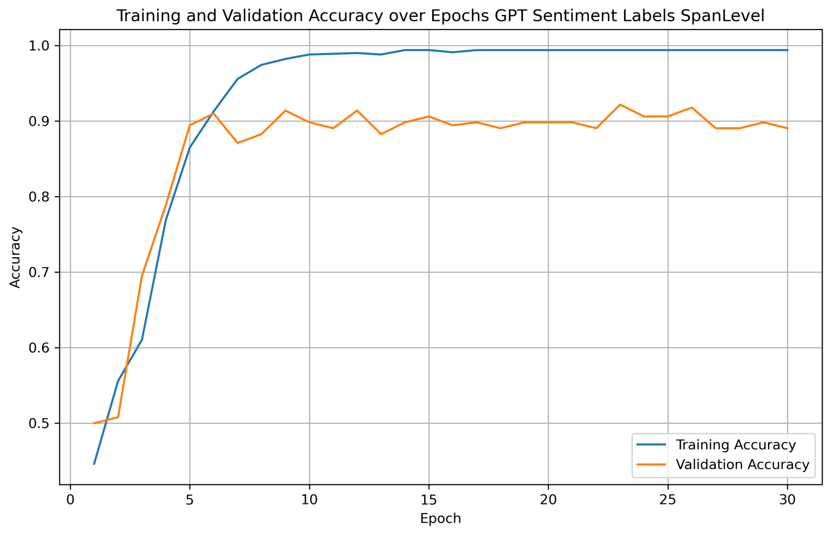
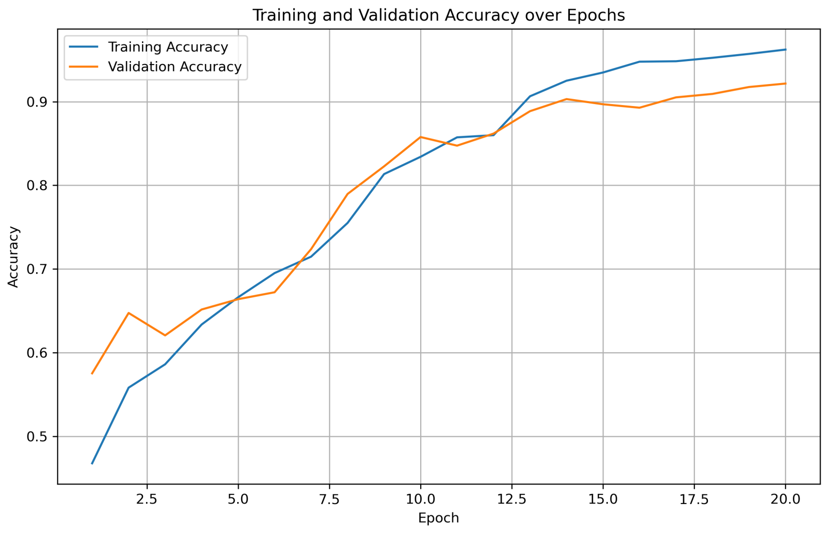


Figure 4.7: Performance of Sentiment Classification with GPT-3.5 turbo annotated data.

4.9.3 *Classification with Mix of Human Labels and GPT-3.5 turbo Labels*

Now we use an input of 50% human-annotated data and 50% GPT-3.5 turbo-annotated data and and fine-tune two BERT models, for the aspect-, and sentiment classification. The resulting training and validation accuracy over epochs is shown in the figures 4.8 and 4.9.



5

Figure 4.8: Performance of Aspect Classification with a mix of human- and GPT-3.5 turbo annotated data (ratio: 50/50).

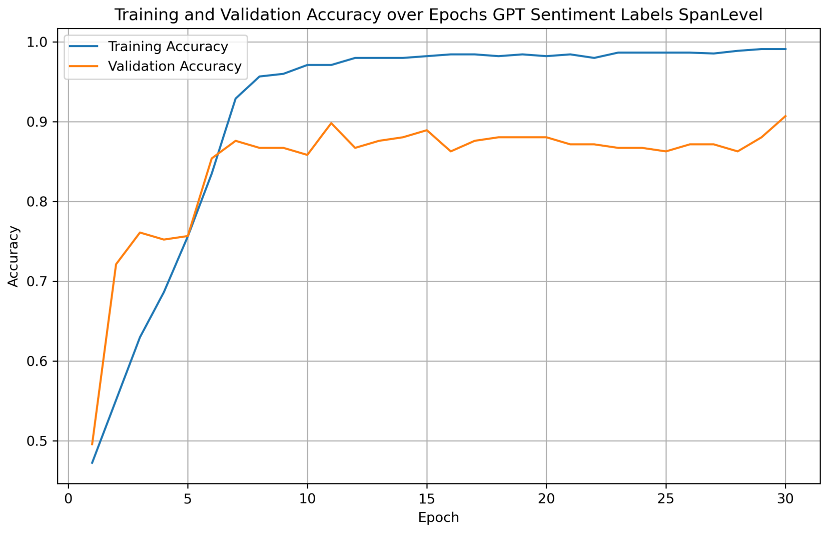


Figure 4.9: Performance of Sentiment Classification with a mix of human- and GPT-3.5 turbo annotated data (ratio: 50/50).

4.10 EXPERIMENTAL RESULTS

The test accuracy for each model is given in the table below 4.3.

Data	Human	GPT-3	Mix Human and GPT-3.5
Test Acc Sentiment	0.92	0.877	0.89
Test Acc Aspect	0.87	0.92	0.95

Table 4.3: Results for all models used in the experiments.

We are observing, that the aspect prediction task with a BERT model, that was fine-tuned with GPT-3.5 turbo annotated data, achieved a test accuracy of 0.92, which even outperformed the model trained on human data. There are two possible explanations. One of them is, that humans also tend to make mistakes during labeling. Even though we tried to reduce mistakes by defining an annotation guideline, they are not completely avoidable. This 'dirty data', affects the model performance in a negative way. Another explanation is, that Human performance on a task is not an upper bound on Large Language Mode (LLM) performance considering an LLM has seen much more data compared to a human [4]. In contrast, the sentiment prediction performance was slightly lower with a test accuracy of 0.87, which indicates that the prompt was defined better for the aspects while being more vague for the sentiment prompts. We get the best result with a mixed dataset comprising of 50% human-annotated data and 50% GPT-3.5 turbo annotated data resulting in a test acc of 0.95. This implies a potential 50% reduction in human annotation efforts.

The results raise the question whether GPT-3.5 turbo annotated data alone could be sufficient for the classification task without the need to fine-tune a smaller BERT model. There are both advantages and drawbacks to this approach. On the positive side, it eliminates the need to deploy an inhouse model like BERT. Maintenance of the deployment infrastructure, create additional cost when running BERT as an API and can be avoided by using GPT-3.5 turbo-, which only requires an effective prompt and a few examples. If the model is not used that much, the ROI (Return on investment) is lower. we can simply make some API Calls with GPT-3.5 turbo. which would be less expensive if the usage of the model is less. If the inhouse model experiences infrequent usage, it results in a lower ROI, opting for GPT-3.5 turbo through API calls becomes a more cost-effective alternative.

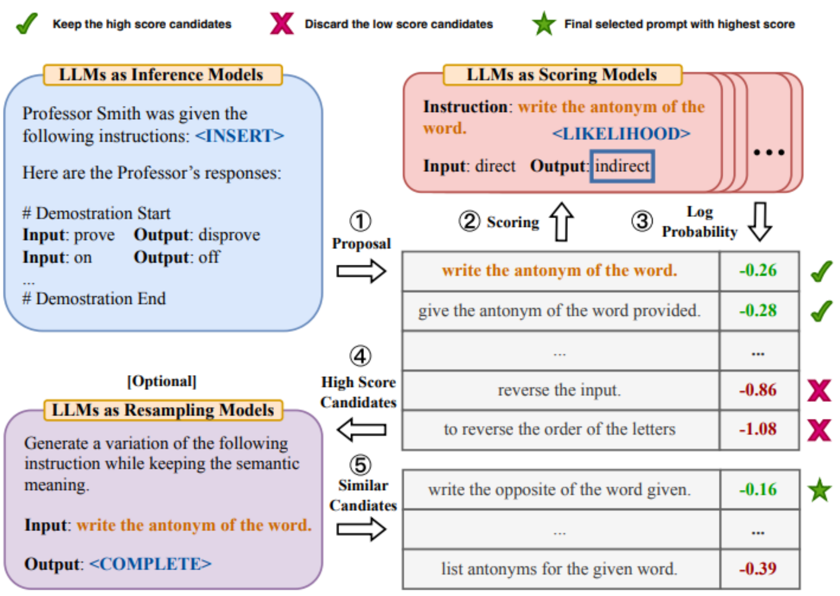
However, a potential drawback lies in the effectiveness of the prompt, with the risk of inaccurate predictions, as observed in our use case where sentiment classification suffered due to missing information in the prompt.
Another drawback is, that by only using GPT-3.5 turbo we highly depend on OpenAI. We must trust, that the model is constantly available and reliable. However, a performance and behaviour shift was observed [6] which indicated that the model has been changed in the backend. Therefore, in order to

get stable results, it would be better to have a frozen model like a fine-tuned BERT.

All in all, if the business is open to trading off accuracy, and are willing to take the risk of having a model, that might change over time, a direct approach utilizing GPT-3.5 turbo can be pursued. However, for those prioritizing higher accuracy, it is advisable to employ a strategy involvingGPT-3.5 turbo labeling coupled with human validation. The resultant labeled data can then be used for fine-tuning a more compact model like BERT.

5 FUTURE WORK

The main challenge when working with LLMs is to write an effective prompt so that the model returns the expected results. However, prompts are very sensitive, and changing one word can lead to major changes in the output. The way a user usually deals with this problem is by using a trial and error approach and manually adjusting the prompt until it outputs the desired results. It is sometimes unclear what exactly led to those changes and to manually adjust the prompt accordingly. Therefore, it is crucial to investigate methods, that help to create effective prompts in an automated way. [41] are facing this challenge and are introducing an approach called Automatic Prompt Engineering (APE). The idea is to let the LLM propose candidates of prompts by giving it a few input-output pairs as examples. Then they let the LLM create variations of those prompts and finally score them to determine the top candidate prompt. They used this approach on 24 different NLP tasks and achieved human-level or even better performance. However, the tasks were relatively simple and the question is how this approach would work on more challenging tasks like in our case. In any case, it would be good if the domain expert started to rewrite the prompt and include information, that was missing in our prompt. Exploring methods for choosing better examples in the Few-Shot learning approach also holds potential for further improvements. There are Few-Shot selection methods like those proposed in [33] which can be incorporated in the future. Another improvement for the future would be to also consider the sentiment 'neutral'. We did not label neutral instances yet, because it was difficult to determine the text span boundaries for neutral spans. By giving a sharp definition of what mentions are considered 'neutral', we could overcome this problem. Another point to keep in mind is to have an additional model, that decides which review is relevant or not relevant. Also, we need another model for the span extraction task, which comes before the final classification model. If we manage to build such a pipeline, we could move on to the next step and deploy the models.



✓ Keep the high score candidates
✗ Discard the low score candidates
★ Final selected prompt with highest score
LLMs as Inference Models
Professor Smith was given the following instructions: <INSERT>
Here are the Professor's responses:
# Demostration Start
Input: prove Output: disprove
Input: on Output: off
...
# Demostration End
LLMs as Scoring Models
Instruction: write the antonym of the word. <LIKELIHOOD>
Input: direct Output: indirect
① Proposal
② Scoring
③ Log Probability
write the antonym of the word. -0.26
give the antonym of the word provided. -0.28
reverse the input. -0.86
to reverse the order of the letters -1.08
④ High Score Candidates
[Optional]
LLMs as Resampling Models
Generate a variation of the following instruction while keeping the semantic meaning.
Input: write the antonym of the word.
Output: <COMPLETE>
⑤ Similar Candiates
write the opposite of the word given. -0.16
list antonyms for the given word. -0.39

Figure 5.1: Automatic Prompt Engineer (APE) workflow [41].

6 CONCLUSION

In this work we tried to answer the question, whether we could reduce human labeling costs for our specific task by leveraging LLM models. We explored an approach to reduce the costs of human annotation by utilizing GPT-3.5 turbo as annotator. We tested this approach on an aspect-based sentiment analysis tasks. The results have shown, that the model for aspect classification, which was fine-tuned on only synthetic data, outperformed the model that was fine-tuned with human data. The model for the sentiment classification, trained on synthetic data yieled a slightly lower performance compared to the one, trained on human data. Improving the prompt with the help of a domain expert who can describe the sentiments in a better way can potentially lead to a better performance here. Considering the improvements of more advanced LLMs like GPT-4 in generating human-like texts, we can expect even better results for synthetic annotation in the future. Also, we expect to be able to formulate shorter prompts. With larger models, the incorporated knowledge increases, and therefore the model could be able to understand short prompts. However, it is not recommended to use models like GPT-3.5 turbo directly for the classification due to the fact, that the model could change its behavior over time and also might not be always available. This is a general problem with models that are not open-source and building a whole application on such a model is risky and therefore not recommended for sensitive tasks.

BIBLIOGRAPHY

[1] Rahul Agarwal. *Understanding Transformers, the Data Science Way*. https://mlwhiz.com/blog/2020/09/20/transformers/. 2023.

[2] Jay Alammar. *The Illustrated Transformer*. http://jalammar.github.io/illustrated-transformer/. 2023.

[3] Gabriela Aranguiz-Dias and Janelle Cheung. *Fine-Tuning BERT with Multi-Task Learning, Gradient Surgery, and Masked Language Modeling for Downstream NLP Tasks*. Accessed: 2023-10-10. URL: https://stanford.edu/class/cs224n/final-reports/final-report-169370473.pdf.

[4] Sam Bowman. "Eight Things to Know about Large Language Models." In: *ArXiv* abs/2304.00612 (2023). URL: https://api.semanticscholar.org/CorpusID:257913333.

[5] Tom B. Brown et al. "Language models are few-shot learners." In: *Advances in Neural Information Processing Systems* 2020-December (2020).

[6] Lingjiao Chen, Matei Zaharia, and James Zou. *How is ChatGPT's behavior changing over time?* 2023. arXiv: 2307.09009 [cs.CL].

[7] Yu An Chung, Yu Zhang, Wei Han, Chung Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. "W2v-BERT: Combining Contrastive Learning and Masked Language Modeling for Self-Supervised Speech Pre-Training." In: *2021 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2021 - Proceedings* (2021). DOI: 10.1109/ASRU51503.2021.9688253.

[8] PyTorch Contributors. *New and improved embedding model*. Accessed November 28, 2023. URL: https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html.

[9] James H. Martin Daniel Jurafsky. *N-gram Language Models*. https://web.stanford.edu/~jurafsky/slp3/3.pdf. 2023.

[10] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of deep bidirectional transformers for language understanding." In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1 (2019).

[11] Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, Lei Li, and Zhifang Sui. "A Survey for In-context Learning." In: *arxiv* (2022).

[12] Andrea Galassi, Marco Lippi, and Paolo Torroni. "Attention in Natural Language Processing." In: *IEEE Transactions on Neural Networks and Learning Systems* 32.10 (2021), pp. 4291–4308. DOI: 10.1109/TNNLS.2020.3019893.

[13] Andrew Kean Gao. *Vec2Vec: A Compact Neural Network Approach for Transforming Text Embeddings with High Fidelity*. 2023. arXiv: 2306.12689 [cs.CL].

[14] Dr. Shivani Sachdev Gour. *In vitro fertilization (IVF) in Delhi*. https://www.drshivanisachdevgour.com/in-vitro-fertilization-ivf-in-delhi/. Accessed November 23, 2023.

[15] R. Greene, T. Sanders, L. Weng, and A. Neelakantan. *New and improved embedding model*. Accessed November 23, 2023. URL: https://openai.com/blog/new-and-improved-embedding-model.

[16] Zhiheng Huang, Davis Liang, Peng Xu, and Bing Xiang. "Improve transformer models with better relative position embeddings." In: *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020* (2020). DOI: 10.18653/v1/2020.findings-emnlp.298.

[17] Kim Hyeyeon. *Transformer based Model - BERT*. https://velog.io/@gpdus41/BERT. Accessed November 23, 2023.

[18] Kun Jing and Jungang Xu. *A Survey on Neural Network Language Models*. 2019. arXiv: 1906.03591 [cs.CL].

[19] Bundesministerium für Familie, Senioren, Frauen und Jugend. *Hilfe und Unterstützung bei ungewollter Kinderlosigkeit*. Accessed on [Nov 2023. URL: https://www.bmfsfj.de/bmfsfj/themen/familie/schwangerschaft-und-kinderwunsch/ungewollte-kinderlosigkeit#:~:text=In%20Deutschland%20ist%20fast%20jedes,Paare%20auf%20medizinische%20Hilfe%20angewiesen.

[20] Katikapalli Subramanyam Kalyan. "A Survey of GPT-3 Family Large Language Models Including ChatGPT and GPT-4." In: *ArXiv* abs/2310.12321 (2023). URL: https://api.semanticscholar.org/CorpusID:264305699.

[21] John Snow Labs. *Manual Annotation*. https://nlp.johnsnowlabs.com/docs/en/alab/annotation. 2023.

[22] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. "BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension." In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (2020). ISSN: 0736587X. DOI: 10.18653/v1/2020.acl-main.703.

[23] Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. *From Quantity to Quality: Boosting LLM Performance with Self-Guided Data Selection for Instruction Tuning*. 2023. arXiv: 2308.12032 [cs.CL].

[24] Tianyang Lin, Yuxin Wang, Xiangyang Liu, and Xipeng Qiu. "A survey of transformers." In: *AI Open* 3 (2022). ISSN: 26666510. DOI: 10.1016/j.aiopen.2022.10.001.

[25] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL].

[26] OpenAI. *API Reference*. https://platform.openai.com/docs/api-reference/chat/create. Accessed November 23, 2023.

[27] OpenAI. *Playground*. https://beta.openai.com/playground. Accessed November 23, 2023.

[28] Long Ouyang et al. "Training language models to follow instructions with human feedback." In: *Advances in Neural Information Processing Systems* 35 (2022). ISSN: 10495258.

[29] *Prompt Templates*. Accessed November 23, 2023. URL: https://python.langchain.com/docs/modules/model_io/prompts/prompt_templates/.

[30] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2023. arXiv: 1910.10683 [cs.LG].

[31] Google Research. *Transformer: A Novel Neural Network Architecture for Language Understanding*. https://blog.research.google/2017/08/transformer-novel-neural-network.html. Accessed November 23, 2023.

[32] Nikunj Saunshi, Sadhika Malladi, and Sanjeev Arora. *A MATHEMATICAL EXPLORATION OF WHY LANGUAGE MODELS HELP SOLVE DOWNSTREAM TASKS*. 2021.

[33] Junsup Shin, Youngwook Kang, Seungjin Jung, and Jongwon Choi. "Active Instance Selection for Few-Shot Classification." In: *IEEE Access* 10 (2022), pp. 133186–133195. DOI: 10.1109/ACCESS.2022.3231365.

[34] Kamaldeep Singh. *How to Improve Class Imbalance using Class Weights in Machine Learning?* Accessed November 28, 2023. URL: https://www.analyticsvidhya.com/blog/2020/10/improve-class-imbalance-class-weights/#h-what-are-class-weights.

[35] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. *How to Fine-Tune BERT for Text Classification?* 2020. arXiv: 1905.05583 [cs.CL].

[36] Cancer Research UK. *Triptorelin (Decapeptyl SR, Gonapeptyl Depot)*. Accessed November 23, 2023. URL: https://www.cancerresearchuk.org/about-cancer/treatment/drugs/triptorelin.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need." In: *Advances in Neural Information Processing Systems* 2017-December (2017). ISSN: 10495258.

[38] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C. Schmidt. *A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT*. 2023. arXiv: 2302.11382 [cs.SE].

[39] Yonghui Wu et al. *Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation*. 2016. arXiv: 1609.08144 [cs.CL].

[40] Albert Au Yeung. *BERT - Tokenization and Encoding*. Accessed November 23, 2023. URL: https://albertauyeung.github.io/2020/06/19/bert-tokenization.html/#summary.

[41] Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. *Large Language Models Are Human-Level Prompt Engineers*. 2023. arXiv: 2211.01910 [cs.LG].

[42] Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. *Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books*. 2015. arXiv: 1506.06724 [cs.CV].

[43] scikit-learn developers. *sklearn.metrics.f1-score*. Accessed: 2023-22-11. 2023. URL: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html.

[44] huggingface. *T5*. https://huggingface.co/transformers/v3.0.2/model_doc/t5.html. Accessed: 2023-11-22. 2023.

[45] huggingface. *transformers library*. https://huggingface.co/docs/transformers/index. Accessed: 2023-11-22. 2023.