Extracting diagnoses from Dutch psychiatric text using pre-trained Large Language Models

Marit Hoek (1528858)

SUPERVISORS

Rosanne J. Turner & Magali de Rooy

University of Utrecht

UMC Utrecht

Department of psychiatry, PsyData

04-07-2024

Master Thesis Applied Data Science

Word count: 9685

Abstract

In psychiatry, the diagnostic journey of the patient is recorded by documenting it in comprehensive clinical notes, making it a vital source of diagnostic information. This study explores the automation of extracting diagnostic categories through the implementation of Large Language Models (LLMs). Specifically, Dutch-trained Transformers were fine-tuned to predict eight diagnostic categories across three types. Three attributes that could have an impact on model performance are compared with one another: the training data of a pre-trained Transformer, the classification method (Named Entity Recognition vs. Multi-Label classification) and the implementation of an oversampling technique. Findings show that RobBERT, a model pre-trained on the general Dutch language, performs better than MedRoBERTa.nl, a model pre-trained on Dutch medical data. Additionally, making classifications per token (Named Entity Recognition) performs better than predicting the presence of each label per text (Multi-Label classification). The impact of oversampling remains inconclusive. This thesis aims to establish a foundation for utilizing Transformers in psychiatric text processing.

Introduction

Imagine that you have broken your arm in a minor incident. After you visit the general practitioner, a clear diagnosis and corresponding treatment plan is delivered to you based on physical symptoms and evidence. Now, imagine seeking help for an overwhelming sense of dread or persisting feeling of emptiness. This may be an indication of a mental illness, but one that is much harder to categorize due to the lack of physical indicators. This is a challenge that patients and professionals within psychiatry face every day, as they try to navigate and treat these complex situations.

To tackle this issue, professionals make a distinction between classifications and diagnoses (Nederlandse vereniging voor psychiatrie, 2014; Stein et al., 2013). Classifications, such as those found in the DSM or ICD-10, categorize symptoms based on standardized criteria, providing a concrete framework for labeling mental health conditions. Diagnoses capture the situation of an individual more broadly by describing symptoms and relevant personal events. In this case, the representation of the mental illness provides more personal context and does not need to be tied to a pre-defined category. Unfortunately, when the differences between these concepts are not made clear, they tend to be used interchangeably.

In the current age of automatization, documenting classifications is as easy as letting psychiatrist take care of some drop-down menus, as these are captured within pre-defined categories. However, with diagnoses, this may not be as easy. They provide context and personal details of a patient’s condition that cannot be captured within automated systems. Therefore, the documentation of clinical notes is necessary, enabling a psychiatrist to adequately record a patient’s diagnosis.

Regrettably, certain diagnoses carry a negative social stigma, presenting difficulty for psychiatrists when assigning them and for patients when receiving them. One example of this is Borderline Personality Disorder (BPD). Often, the assumption is made that people with BPD are irresponsible, weak and unstable, regardless of their personality or history (Clearview Treatment Programs, 2019). Another general belief is that the mental illness is untreatable, even though this is not the case anymore. These stigmas can lead to negative consequences for the patient, such as being misunderstood, bullied, or misdiagnosed. Additionally, they can encounter various challenges caused by their undiagnosed condition that arise from reluctance to seek help.

The PsyData team in the Department of Psychiatry at University Medical Center Utrecht (UMCU) aims to investigate whether diagnoses that are socially stigmatized are truly more difficult to provide. Evidence is sought through quantifying instances of classifications, misclassifications, and second opinions in the field. Yet, there is one main obstacle that needs to be overcome before this task can be properly executed. Classifications of patient’s (mental) illnesses can be found in their Electronic Health Records (EHR). However, these classifications do not often represent the situation of the patient accurately. This situation generally arises from the requirement imposed by insurance companies for patients to receive treatment only after providing classifications. These classifications are often inaccurate, serving more as a procedural necessity rather than an accurate representation of the patient’s situation. Consequently, documentations are often rushed without proper confirmation of the diagnosis and can become outdated when subsequent diagnoses are confirmed but not updated. Additionally, they do not keep track of classifications that have been considered during the diagnostic journey of the patient. The most comprehensive and accurate representation of the diagnosis and classifications of a patient often lies within the detailed clinical notes of a psychiatrist. Unfortunately, gathering extensive diagnostic information from these clinical notes would consume too much valuable time. Therefore, the aim of this research will be to automate this process of extracting classifications from clinical notes by using current state of the art Large Language Models (LLMs). First, the theoretical framework of language processing will be discussed, including the evolution of Language Models (LLMs) and why the mechanism called ‘Transformers’ will be implemented in the final assessments. Secondly, methods for text analysis will be outlined, providing the data and analytical approaches used. Thirdly, results will be showcased in terms of evaluation metrics and detailed interpretations of predictions. Lastly, conclusions will be drawn based on the findings, along with recommendations for future implementations.

Theoretical framework

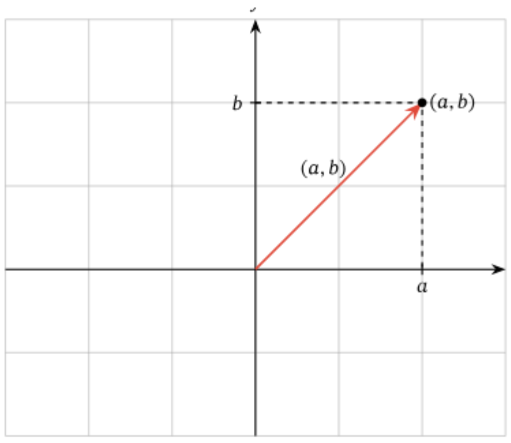
When machine learning started to take over the world, it was mostly designed to process structured data, such as numerical data places within columns and rows. Eventually, the question arose if one could process unstructured data as well, like text. Researchers started making advances in that area, which is now called Natural Language Processing (NLP). NLP tasks include the interpretation of human language with computer science methods. The aim of these tasks is to make processes more efficient by saving time spent on manual labor and better allocating resources. Tasks such as text summarization, text classification, sentiment analysis, entity recognition, question answering, and text generation are all methods that can help automate text analysis.

Bag of Words model

The processing of text within the field of computer science can be traced back to the early 1950s, with the introduction of one of the most fundamental processing techniques called the Bag of Words (BOW) model (Harris, 1954). This method can analyze a set of documents, called a corpus, and numerically represents each document with a fixed-length vector. For illustration, figure 1 shows a two-dimensional vector, which consists of two values. While vectors can have many more dimensions, visualizing such high-dimensional data can often be too complex.

Figure 1

A Two-Dimensional Vector



Note. From *Lesson explainer: Vectors in terms of fundamental unit vectors mathematics*, by Nagwa, n.d., (https://www.nagwa.com/en/explainers/578165351487/)

In the simplest version of the BOW model, the length of the vector corresponds to the number of unique words that are present in the corpus. The numbers within the vector represent the count of the unique word within the document, creating a numerical representation of the document. This principle is illustrated with a relatively simple example in table 1, but these vectors often grow to extremely large sizes when it covers a collection of texts containing many different unique words, taking up an excessive amount of memory.

Table 1

Text Vectors in the Bag Of Words (BOW) Model

	The	Cat	Dog	And	Other	I	Love	Saying
The cat and the dog	2	1	1	1	0	0	0	0
The dog and the other dog	2	0	2	1	1	0	0	0
I love saying cat: cat cat cat	0	4	0	0	0	1	1	1

It is expected that documents with similar vectors will be similar in content as well, as they will contain similar words that are represented in the vector space in a resembling manner.

Using mathematical measures like cosine similarity, vectors of documents can be compared with the aim of finding documents similar in word usage (Hackeling, 2017).

Although a good first step in the right direction, the method is far from perfect. The method can give an idea about the discussed topics in the documents, but it has a harder time differing between documents with different contexts. For example, a document that is pro-cats and a document that is anti-cats could still receive similar vectors, as they are both discussing the same topic: cats.

The TF-IDF framework

A popular method for enhancing the performance of the BOW model is by using the TF-IDF method (Hackeling, 2017). By emphasizing rarer words, such as 'cat' or 'train', documents are more likely to be matched on similar topics, as the focus on common words that do not relate to specific topics is reduced. These include words such as ‘the’, ‘a’ or ‘I’. Sometimes, these common words known as ‘stopwords’ are removed during preprocessing altogether. The main limitation of the BOW model remains, as the model still is unable to process the order of words and the context in which they are used.

Tokenization

Additional enhancements were made to the BOW model, including the implementation of different tokenization methods. Tokenization is the process of breaking up a piece of text into multiple ‘tokens’, such as words, subwords or characters. For instance, in the most basic format, a text can be broken up into separate words, where every token is then equal to a word. The same applies to using subwords or characters as tokens.

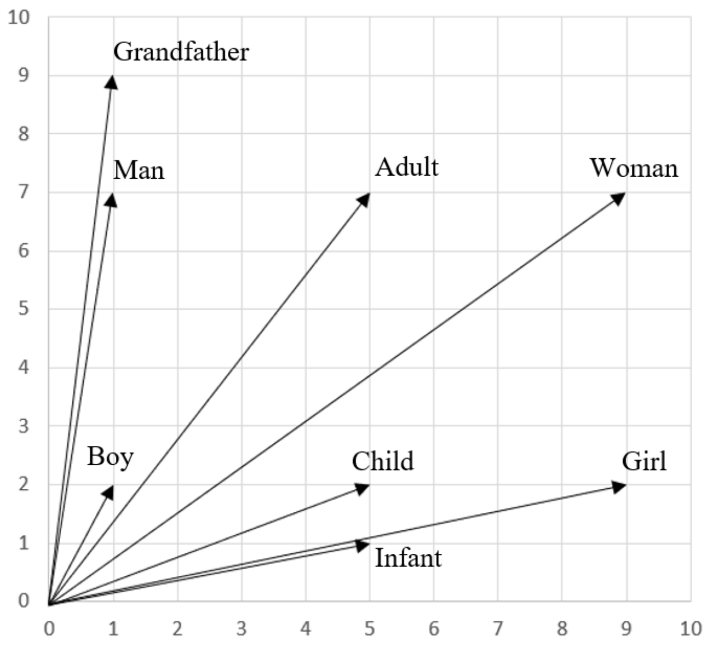
Another option is to not only break up the text into parts, but also to homogenize words that are different in spelling, but similar in meaning. One example of such a collection could be ‘walking’, ‘walk’ and ‘walked’. Some techniques, amongst others, that exist to address these phenomena are stemming or lemmatization (Hackeling, 2017). Using stemming might change the word ‘walked’ to ‘walk’, whereas implementing lemmatization is more relevant with special conjugations, such as changing ‘are’ and ‘is’ to ‘be’. In the aforementioned collection, both methods would shorten the words so they may all be represented the same way. This way, multiple variations of the same term may be grouped together in the vector space, increasing the accuracy when comparing documents with one another.

Word2vec

With the rise of neural networks in computer science, researchers began exploring their implementation for text processing. A crucial method that paved the way for future advancements was Word2Vec (Mikolov et al., 2013). It trains artificial, data-driven neural networks to create vector representations of words based on context words, also known as word embeddings. It trains on a large collection of text data, and consists of two main architectures: Continuous Bag-of-Words (CBOW) and Skip-Gram. CBOW predicts a target word based on the context words surrounding it using a hidden layer and an output layer, whereas Skip-Gram predicts the context words given a target word. This way, words that are used in similar context will have similar meaning, and therefore similar vector representations. This notion is visualized in figure 2, using two-dimensional vectors to represent specific words, where similar words end up being close to each other within the vector space. Note that in Word2Vec, it is common practice to implement vectors ranging between 100 and 300 dimensions (Kwan, 2023).

Figure 2

Word embeddings of similar words



Note. From *Finding the optimal number of dimensions for word embeddings,* by M, Kwan, 2023, Medium (https://medium.com/@matti.kwan/finding-the-optimal-number-of-dimensions-for-word-embeddings-f19f71666723)

A common example of showcasing the ideas behind CBOW is using a sentence with made-up words to exemplify how humans intuitively estimate what the word represents. Like, ‘Susan ate the blorple and it was delicious’, or ‘I petted the flumbar and it wagged its tail’. Humans are able to predict that a blorple might be a kind of food, whilst a flumbar might be an animal, purely based on the words that surround them. They would, respectively, receive similar embeddings as ‘cake’ or ‘banana’, and ‘giraffe’ or ‘dog’. Skip-gram works the other way around, which is attempting to predict context words given a specific word. For example, if a person mentions a particular animal (such as a flumbar), others might think of associated words like 'pet' and 'tail'. Combining both methods results in word embeddings that have a more nuanced understanding of context and these embeddings can in turn be used to perform NLP tasks such as sentiment analysis or text classification. By taking words’ context into account while creating contextually relevant word embeddings, Word2Vec outperforms the BOW model (Jurafsky and Martin, 2019).

Even though the word embeddings are created based on the surrounding context, an embedding eventually ends up being a fixed, static representation of a word. This means that it may not work as well on so-called ‘polysemous’ words that can differ in meaning based on context. An example of this is ‘bank’, which can be able to refer to either a financial institution or the side of a river.

Recurrent Neural Networks

To address the limitations of the Word2Vec method, Recurrent Neural Networks (RNNs) were created. RNNs were designed to further optimize the use of word embeddings and turned out to be specifically useful for performing translation tasks. This accomplishment is mainly achieved by their use of the encoder-decoder architecture and a feedback loop (Tunstall et al., 2022).

The encoder-decoder framework

The encoder-decoder framework consist of an encoder and a decoder. These are two separate neural networks with their own distinct responsibilities, which can then collaborate to perform NLP tasks, such as text translation. The encoder serves to create a numerical representation of the input sequence, called ‘the last hidden state’, which are then forwarded to the decoder that performs its job of generating an output sequence. For this thesis, the focus is on text processing, but the encoder-decoder framework can also handle other types of input, such as images.

The feedback loop

The feedback loop was invented to take the word order into account within text sequences, as these sequences may differ in meaning depending on the order of the words it contains. For example: “The girl liked only him” and “Only the girl liked him” consist of the same words, but are interpreted differently. The feedback loop gathers information while creating an embedding of a word or token and uses it to create the next embedding of the subsequent word or token in the sequence. By taking into account the word order, the model creates word embeddings that more accurately represent the sequence at hand.

The main limitation of the RNN is the presence of an ‘information bottleneck’ at the final hidden state created by the encoder, which is also referred to as the ‘vanishing gradient’ problem. The aim of the encoder is to represent an entire input sequence, allowing the decoder to generate an output from it. Unfortunately, the model processes the sequence, earlier information can gradually diminish and be forgotten. This becomes specifically challenging with longer sequences, as the gap between the beginning of the input and the creation of the final representation at the end of the sequence increases.

Long Short-Term Memory Networks (LSTMs)

To address this limitation, Long Short-Term Memory Networks (LSTM) were introduced. LSTMs are based on RNNs, with a slight modification. LSTMs are designed to better capture context in longer sequences and minimize the effect of the vanishing gradient problem. The models make use of gates, placed at every feedback point, that can decide which information is relevant to feed forward and which information can be forgotten about. By only transmitting the most relevant information, it is less likely that the context that is present in the start of the sequence will be forgotten.

The attention mechanism

Another technique to lessen the impact of the vanishing gradient problem is ‘attention’. Attention is based on the notion to give the decoder more access to information of the input sequence than solely the last hidden state of the encoder. This would enable the decoder to receive a more comprehensive representation of the context within the input sequence and therefore, generate output sequences with greater accuracy.

However, granting access to all the existing hidden states the encoder will create an input for the decoder that is too large, which might increase computational load significantly (Tunstall et al., 2022). To keep this impact to a minimum, the decoder is enabled to assign different weights to the hidden states of the encoder, called ‘attention’. For instance, within the context of translation, the decoder will find a higher level of attention between a word in the original sequence and its translation, even if these are positioned differently within the sentence. For example, the Dutch translation for “I am eating a banana” is “Ik ben een banaan aan het eten”. Regardless of the varying position of the word ‘banana’, attention will still link both instances together. Even though the quality of translations increased due to this implementation, the time it took for computation was far from ideal. Because each processing step must wait for the preceding step to complete, this "sequential processing" can consume a significant amount of time.

Transformers

In 2017, Vaswani et al. introduced the Transformer architecture, which outperformed RNNs in NLP tasks and is currently considered the state of the art in language processing. The Transformer's goal is to overcome the limitations of sequential processing by utilizing a special form of attention called ‘self-attention’ (Tunstall et al., 2022). This mechanism is based on ‘attention’, but applies the method within the input sequence, as opposed to linking words in the input sequence to the output sequence. It weighs the importance of each word within a sentence based on its relevance to other words in the same sentence, depending on context. For instance, in the following sentence: “The banana was put on the table and it turned brown”, the self-attention mechanism will display a higher level of self-attention between ‘banana’ and ‘it’. This technique is not bound to sequential processing, as, it can consider the connection between any two words in the sentence, regardless of their distance. The encoder and decoder both have their own self-attention mechanisms.

A Transformer model can implement multiple self-attention mechanisms simultaneously and this is referred to as ‘multi-head attention’. The use of multiple heads allows for the model to compute the self-attention within the input data synchronously, which is called ‘parallel processing’. Consequently, the Transformer provides a more comprehensive interpretation of the input sequence and reduces the runtime significantly.

Transfer learning

Another advantage of the Transformer architecture is that it enables transfer learning, making the need of training the model from scratch for every single task redundant. The model is divided into a ‘body’ and a ‘head’, which differs from the concept of a ‘head’ within self-attention, each having their own responsibilities. The body is designed to be ‘pre-trained’ on a large dataset, such as millions of rows of unsupervised text, from which it will create an underlying understanding of the text embeddings. After the pretraining is done, the head can undergo a process called ‘fine-tuning’, as the head is meant to be trained on a specific NLP task based on a relatively smaller labeled dataset. The head of an existing model can be reinitialized to be fine-tuned on a variety of NLP tasks, which makes this method specifically useful within domains where there is a lack of labeled data.

Encoder and Decoder models

There are transformers that only use the decoder part of the architecture, such as Generative Pre-trained Transformers (GPT), meaning it only performs the job of decoding sequential representations. Additionally, transformers exist that solely employ the encoder part, like Bidirectional Encoder Representations from Transformers (BERT), which is pretrained on the BookCorpus and English Wikipedia. BERT utilized a unique form of language modelling called ‘masked language processing’, of which the aim is to predict randomly masked words in a text. For instance, such a sentence may look like this: “I watched a movie on [MASK], and it was called the [MASK]”.

Whereas decoder models like GPT might be utilized to create new text, encoder models like BERT are usually preferred for classification tasks due to their superior knowledge of text embeddings (Raschka, 2023), which is also observed in research (Benayas et al., 2024). Researchers have begun implementing their own modifications and enhancements to these models since their release, such as DistilBERT (Sahn et al., 2019) or RoBERTa (Liu et al., 2019). DistilBERT is, as is said in the name, a distilled version of BERT. By only keeping the most effective and relevant parameters, the model requires less computational resources while keeping most of BERT’s original performance. As for RoBERTa (Robustly Optimized BERT Approach), it is an improved version of BERT. Though maintaining the same architecture as BERT, it implements several training and optimization techniques. These techniques include more training data, larger batch sizes and elimination of the next-sentence prediction task, which allows for more computational resources to focus on the masked language processing task. Due to these methods, RoBERTa is able to outperform BERT on certain NLP tasks (Raschka, 2023).

Performance factors

Language of the training data

The performance of a model might not only depend on its underlying architecture, but also on the data that is used as input for the model. Firstly, a distinction can be made between models that are trained solely on one particular language (monolingual models) and models that contain multiple languages within their training data (multilingual models). Whereas multilingual models are ideal for translation tasks, research shows that monolingual models perform better than a multilingual model on a multitude of tasks that requires the text analysis of a singular language (Martin et al., 2019). The same is shown for Dutch language with models like BERTje (de Vries et al., 2019). BERTje is a Dutch model based on the original BERT architecture, but trained on Dutch text sources like Wikipedia, books and web News. This model outperformed mBERT (multilingual BERT), which is based on the same architecture, but trained on data from multiple languages. Another well-performing Dutch model is RobBert (Delobelle et al., 2020), which consists of the RoBERTa architecture, but trained from scratch on a larger proportion of the OSCAR corpus, which is an online repository of multilingual resources and datasets designed for applications in machine learning (ML) and artificial intelligence (AI) (Abadji, 2022). This model was trained on a dataset four times as big as the one BERTje was trained on and outperformed the model on various tasks.

Domain of the training data

Secondly, there is a preference for selecting a model that is trained on domain-specific text. Studies in the biomedical sector indicate that this approach enhances the performance by capturing contextual representations more effectively, due to the inclusion of domain-centric linguistic features within the text (Lee et al., 2019; Alsentzer et al., 2019; Si et al., 2019). This implies that to optimize text analysis within psychiatry, a Transformer should be trained on a substantial volume of psychiatric text data. However, data available for training in the clinical sector is scarce, as the data is protected due to their sensitive nature. The data includes information about patients' physical and mental health, as well as confidential details about their personal lives and potentially, their families. If this information were to be leaked to the public, the patient or their family members could endure severe negative consequences, such as social stigmatization, financial exploitation and emotional distress (Corrigan, 2002). Within psychiatry, the impact could be even more severe, as individuals undergoing treatment are typically mentally vulnerable and could be less resilient to the potential emotional fallout. To minimize the risk of such occurrences, the General Data Protection Regulation (GDPR, 2018) was implemented, placing strict conditions under which sensitive data can be accessed. This thesis aims to investigate whether privacy concerns can be circumvented by assessing the performance of available pre-trained Transformers in the task of extracting diagnoses[1] from psychiatric texts. If this proves to be insufficient, the need to train a Transformer from scratch on domain-specific data within psychiatry remains.

[1] From this point onwards the term ‘diagnoses’ will be used in this paper, acknowledging that these can be technically defined as ‘classifications’. This choice was made to emphasize the psychiatric environment and avoid repetitive references to 'classifications based on diagnostic criteria'.

Data and Methods

Data

Due to both privacy concerns and constraints in time and resources, the decision was made to choose a pre-trained transformer model to be fine-tuned to extract diagnoses from clinical, psychiatric notes. For this objective, text data was provided by the psychiatry department of University Medical Center Utrecht (UMCU), through the research-based subdivision known as PsyData. PsyData aims to conduct research that enhances the effectiveness and accessibility of data analysis within the psychiatry. Among other things, the team is responsible for the creation of an anonymization algorithm called 'DEDUCE' (DE-identification method for DUtch mediCal tExt). The algorithm can be applied to Dutch medical texts to mask names, places, institutions, ages, dates, patient numbers, contact information and URLs (Menger et al., 2018). This was done with the intent of minimizing the amount of personal and potentially retraceable information.

The obtained data consisted of the conclusion sections extracted from anonymized Dutch psychiatric letters, containing summarizing information about the treatment of a patient. The conclusions were extracted from the texts by identifying the conclusion headers and retrieving the text that followed up to four line breaks. A distinction can be made between discharge letters and outpatient letters, which are concluding letters about patients that have been discharged after admission or those that had treatment appointments without admission, respectively. These letters were chosen as they contain the most concise and complete overview of the final established diagnoses, along with the diagnoses that were considered throughout the diagnostic process. These texts are deemed suitable for fine-tuning the transformer, as they comply with the intent to retrieve the most accurate diagnostic representations from texts. Additionally, by only retrieving the conclusions, duplicate patient information can be avoided.

Annotation and classification categories

Before these conclusions can be used as input data for the transformer model, the conclusions had to be annotated and labeled so the model could train on such a classification task. Within the available time frame, it was deemed realistic to randomly sample 500 conclusions from the psychiatric letters and annotate them based on a set of predetermined diagnostic categories. These categories were determined with the help of a psychiatric professional to decide which collection of categories were encapsulating the most relevant diagnoses, without making them too specific to optimize model performance. This process is outlined within the master's thesis of van Ginkel, E. (2024), which is unpublished at the time of writing this thesis. It will be available in the Utrecht University Student Theses Repository (UU, 2024) after this thesis is submitted. Eventually, the following categories were created along with their labels based on their Dutch namesake: Attention Deficit Hyperactivity Disorder (ADHD), Autism Spectrum Disorder (ASS), Bipolar I Disorder (BIP1), Borderline Personality Disorder (BORD), Major Depressive Disorder (DEPR), Post-Traumatic Stress Disorder (PTSS), Schizophrenia (SFR), and 'Other Diagnoses' (ANDI).

An additional distinction was also made for the type of diagnosis, with the aim of capturing the context in which the diagnosis was mentioned. This consisted of following categories: Diagnosis, Consideration and Not relevant. The category 'diagnosis' (d) covers all the diagnoses which are definite, whereas 'consideration' (o) covers all diagnoses that are still being assessed. Lastly, the 'not relevant' (nvt) category covers all diagnoses that were discarded or concern family members or friends of the patient that are described in the conclusion.

Eventually, every classification label that is processed by the transformer model consists of a combination of the diagnosis and its accompanying type, meaning the model will be trained to make distinctions between 24 different labels, as there are 8 diagnosis classes and 3 possible types.

Model selection

Based on the information previously stated in this paper, it becomes clear that multiple aspects should be taken into account when choosing the pretrained model to optimize the performance of the NLP tasks in question. It is expected that the performance of a model increases if it is trained on a vast amount of both domain and language specific data. In addition, an encoder model is preferred when training for a classification task, as it has been shown that their general performance in classification is superior to that of decoders (Benayas et al., 2024; Raschka, 2023).

One such model which meets these requirements, is MedRoBERTa.nl. The model is based on the RoBERTa architecture, but trained on 13GB of hospital notes from the Amsterdam University Medical Center (Verkijk and Vossen, 2021). The model was created with the intention of capturing the specific terminology and context that is more prominent in clinical settings, which can differ significantly from generically used Dutch language. It seems to be ideal choice for this research, as the discharge conclusions may contain clinical terms. Additionally, it meets the requirements of serving as an encoder and being language-specific.

However, it remains unclear what proportions of the clinical training data actually fall under the psychiatric domain, and this may affect model performance due to domain-specific nuances. Therefore, all models executed with MedRoBERTa.nl were also tested with RobBERT, which shares the same architecture but is trained on general Dutch language (Delobelle et al., 2020). This approach enables a direct comparison of model performance. Both MedRoBERTa.nl (Computational Lexicology & Terminology Lab VU, 2023) and RobBERT (Delobelle, 2023) are publicly available for implementation on the Hugging Face website. Hugging Face also developed the Transformers library, containing programming tools to easily download and train the available pretrained models (Hugging Face, 2024).

Method selection

The goal of the project is to extract diagnoses from psychiatric clinical texts. Various approaches can be utilized to accomplish this. One straightforward approach would be to implement a token classification model that tries to predict the diagnosis for each individual token. Token classification, more commonly known as Named Entity Recognition (NER), seems to perform well to distinguish between different types of entities, like organizations, names and places (Barney, 2023). However, since all diagnoses fall under the same entity of 'diagnosis', it remains uncertain whether the model can adequately differentiate between the various classifications. Therefore, Multi-Label Classification (MLC) was implemented as a second method, to allow for a comparison in performance. This method does not classify per token; rather, it classifies the text it receives as a whole and can assign multiple labels to it. This is applicable to the task at hand, as multiple diagnoses may be stated within the same text. However, this method may have drawbacks as well, as the model will not receive input on which terms belong to which diagnosis. The model would have to figure this out independently.

Pre-processing

After annotating the 500 discharge conclusions, some pre-processing steps had to be implemented before the data was suitable for fine-tuning both the MedRoBERTa.nl and RobBERT model. The first step that was taken was resetting the text to its original format by replacing <br> (the symbol representing a line break) with a space. Following steps included creating a train-test split, formatting the data, extracting labels and tokenization.

Train-test split

First, the available 500 clinical notes were split into a train and a test set. 20% of the texts were to be set aside as the test set, and 80% was used for training the model. To ensure that diagnoses with relatively low frequencies appear in both the train and test set, the split was implemented while retaining the same distribution of label presence in both sets. This was done through an iterative train-test split, which ensures similar class distributions in multi-labeled datasets. The split does not always result in the same outcome, but as the differences are minimal, it has marginal impact on model performance.

Model size limit

Firstly, both models are based on the RoBERTa architecture, which has a token limit of 512 tokens, each token representing a subword. This means that the model cannot process the context of an entire unbroken conclusion if it exceeds this token limit. The psychiatric conclusions vary in sizes and it is observed that 24% of the 500 texts surpass this limit. When texts exceed the token limit, it is standard practice to truncate the text, which basically means that the ‘leftover’ text will be thrown away. This is not an optimal solution, as it can lead to a major loss in information, such as missed diagnoses.

Another option is to split the text into multiple chunks to retain all the information. Note, though, that this should be done while leaving overlapping text between the chunks belonging to the same original conclusion to preserve context (Dai et al., 2022). However, this method may increase the runtime significantly, likely due to processing relatively large texts. As the text lengthens, the self-attention mechanism needs to determine a greater number of self-attention weights between the input tokens, which leads to a quadratic increase in memory requirements and consequently, runtime (Dao, 2023). Regarding these text size considerations, different decisions were made for both MLC and NER.

Named Entity Recognition

During the implementation of NER, the text was processed in chunks to retain as much context as possible. The chunks were appropriately sized to ensure they would not exceed the token limit of the model. Full sentences were retained within the chunks to prevent loss of context that might occur if a sentence were truncated mid-sentence. The effect of chunking would have on the model’s runtime is expected to be minimal, as only 24% of the 500 texts exceed the token limit, resulting in 120 additional chunks.

Next, every individual word in the text data needed to be classified. After splitting the text into separate words, a column was added with a list of labels that correspond to the order of the words in the sentence. This was achieved by linking annotated words to their annotated label. The label ‘NONE’ was created for words that were not annotated and did not represent any type of diagnosis. After completing this process, the annotations were removed from the texts, to enable the model on training and testing on real-life data, enhancing the generalizability of the results. However, to fine-tune a model, it requires numerical inputs.

Therefore, each label was mapped to a unique number, and then these numerical representations were used to replace the original labels. The mapping was stored separately so that numerical predictions could be converted back to their original label names when necessary. An example of the final format is shown in table 2.

Table 2

Example Data Structure for NER

Text	Labels
De diagnose voor **ADHD** wordt overwogen. Hij werd doorverwezen.	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
John werd gediagnosticeerd met **PTSS**. Zijn broer was bekend met een **leerstoornis.**	[0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 3, 0.]
Er werd **PTSS** vastgesteld met een eventuele overweging voor **ADHD**.	[0, 0, 2, 0, 0, 0, 0, 0, 0, 1, 0]

Note. Punctuation is labeled as well

Multi-label classification

To enable the transformer to attach multiple labels to a singular instance of text, multi-label classification was implemented. Instead of training the transformer to retrieve one classification, the transformer was trained to retrieve all the classifications deemed relevant by the model. This is relevant for the current research, as a text can contain multiple diagnoses. Instead of creating a list of labels corresponding for every word, as is done with NER, a column is added to the text for every label where its value represents the labels absence (0) or presence (1). After this process, the annotations are removed from the text, just as with NER. It is expected that after fine-tuning a Transformer, the model will be able to identify the presence of a label within each text, disregarding its quantity.

It has previously been established that the texts can be split into chunks to retain context and information. However, this may not be the preferred technique to use with this particular method. As we apply multi-label classification, the aim is to retrieve diagnoses that are explicitly stated. An issue that may occur if the model were to be processing the conclusions in their entirety, is that the model might eventually associate the described symptoms with the attached labels, even though the same symptoms could be present in multiple diagnoses. This might increase the chance of incorrectly labeling certain texts based on the described symptoms, even though no exact diagnoses are mentioned. Therefore, the text was split into separate sentences, as it expected to minimize the amount of text, or 'noise', that surrounds the explicitly stated diagnoses. Table 3 exemplifies the final format of the labeled data.

Figure 3

Example Data Structure for MLC

Text	ADHDo	ANDInvt	PTSSd
De diagnose voor **ADHD** wordt overwogen.	1	0	0
Hij werd doorverwezen.	0	0	0
John werd gediagnosticeerd met **PTSS**.	0	0	1
Zijn broer was bekend met een **leerstoornis**.	0	1	0
Er werd **PTSS** vastgesteld met een eventuele overweging voor **ADHD**.	1	0	1

Label imbalance

It is expected that not all diagnoses are encountered with similar frequencies during annotation, as some are encountered more frequently than others at the Psychiatry department of UMCU. This could potentially cause imbalances in the labeled dataset, which might impact model performance for the less common diagnoses. This is caused by the model being exposed more frequently to common diagnoses, allowing it to learn and predict these more accurately. To account for this imbalance, weighting of the loss function was implemented (Shrivastava, 2020). Evaluation of the loss function is implemented while training the model, where a lower loss represents a better performance. By weighting the function according to label occurrences, the model will place more emphasis on the loss of diagnoses with fewer occurrences, thereby compensating for the imbalanced data.

In addition, the technique of oversampling was utilized, where the model is exposed more often to the texts containing uncommon diagnoses by duplicating a proportion of these texts one or multiple times. Although this technique aims to enhance performance by increasing the model's exposure to minority classes, it also heightens the risk of eventual overfitting (Shrivastava, 2020). The models were run both with and without oversampling so enable comparison on model performance. Due to the non-uniform outcomes of the oversampling technique, label occurrences within the dataset varied for each model run.

Tokenization

After splitting the data, the text was tokenized. Both MedRoBERTa.nl and RobBERT implement the same tokenizer as RoBERTa, which is a subword tokenizer applying Byte-Pair Encoding (BPE) (Liu et al., 2019).

BPE starts with a base vocabulary of individual characters and progressively merges the most frequent pairs of characters or subwords to form longer subwords. This process continues until a predefined vocabulary size is reached. Eventually, tokens can consist of either words or subwords. For instance, tokenization may be split into ‘token’ and ‘ization’, and RoBERTa may be represented as ‘Ro’, ‘bert’ and ‘a’. By using a subword tokenizer, the model can handle rare and complex words by splitting them into familiar chunks, making it more effective at understanding and processing language. To signify for every new word that is added to the sentence after the first one, a ‘Ġ’ is added to the token that belongs to the start of the word. Special tokens are used as well, to represent the beginning and ending of a sentence, respectively captured by <s> and </s>.

As a final example, the following sentence: "Tokenizing text is a core task of NLP." Will be tokenized as such: ['<s>', 'T', 'oken', 'izing', 'Ġte', 'xt', 'Ġis', 'Ġa', 'Ġcore', 'Ġt', 'ask', '.', '</s>']. For the NER models, tokenization will cause for the number of labels per text to increase after tokenization, as a labeled word can be split into multiple subwords matched to the label.

Training the model

The MedRoBERTa.nl model was initialized for sequence classification, meaning that only the classification head responsible for the multilabel classification task would be modified during fine-tuning. The initial number of epochs was set to two, indicating that the model would learn from the train set by iterating over it twice. Upon evaluating the runtime of the models, it was found that the NER model ran considerably faster than the MLC model. As a result, the number of epochs for the NER models was increased to three. Both models implemented a learning rate of 2e^-5 and the weight decay was set to 0.01.

Evaluating the model

The model was evaluated based on its performance on the test data. The following metrics were collected per model for evaluation: Accuracy, precision, recall and the F1 score. The accuracy represents the proportions of the instances that were correctly classified. Precision and recall take into account the correctly predicted labels (True Positives and True

Negatives) and the incorrectly predicted labels (False Positive and False Negative) per label to create a more nuanced performance indication. These concepts are visualized in table 4 in what is known as a confusion matrix.

Table 4

A Confusion Matrix with True/False Positives (TP/FP) and True/False Negatives (TN/FN)

		Predicted	
		Negative	Positive
Actual	Negative	TN	FP
	Positive	FN	TP

Precision indicates the proportion of correct predictions among all predictions made for a specific label (TP/TP+FP), whereas recall indicates the proportion of the actual instances of the label that were correctly predicted out of all instances with that label (TP/(TP+FN). The goal is to maximize both of these metrics, as reflected in a high F1 score, which is displayed in figure 3.

Figure 3

The Formula for the F1 Score

$$\textbf{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Note. From *F1 score in Machine Learning: Intro & Calculation*, by R, Kundu, 2022, V7 (https://www.v7labs.com/blog/f1-score-guide)

Further explorations with the top-performing model

The models may exhibit poor performance due to various factors, such as insufficient training data for specific labels or challenges in distinguishing between different types within the same diagnosis category. To investigate whether mitigating these issues improves performance, the top-performing model will be reselected and run without any distinction between types among the labels. This step aims to determine if the model can provide a more accurate basis by differentiating between the general diagnoses.

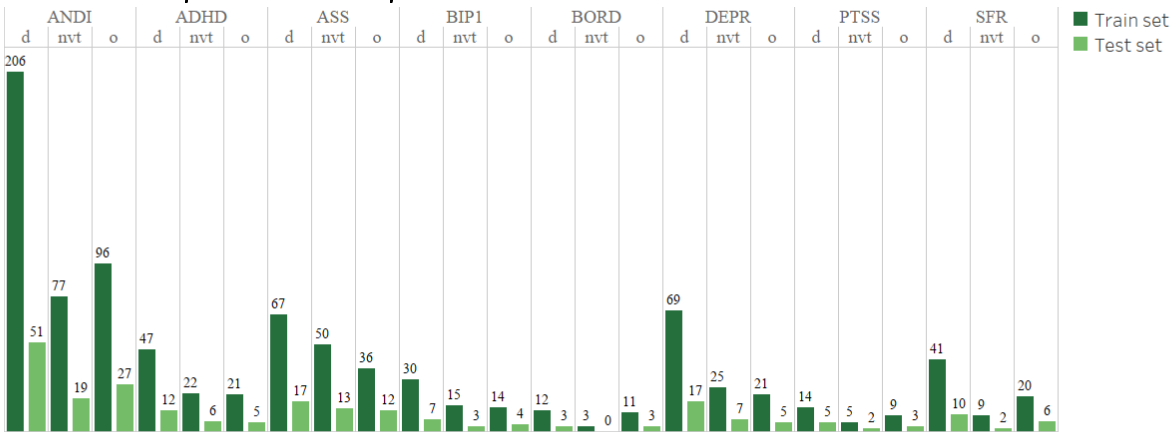
Results

Iterative train test split

The dataset used to fine-tune the models comprised 500 clinical texts, of which 457 contained relevant diagnoses. The texts contained 221 words on average, including numbers, with a standard deviation of 134. After labeling each text within this dataset with the presence of each label, the iterative train test split was performed. The resulting distribution is visualized in figure ? and shows that the distributions of present diagnostic labels between the train and test split are indeed quite similar. From the figure it can also be deduced which labels are quite uncommon, such as Borderline Personality Disorder, PTSS or Schizophrenia. The most common label seems to be ANDI, referring to 'other diagnoses'.

Figure 4

Distribution of the Presence of Labels Within the Train and Test Set



Data descriptives and label distribution before oversampling

For Named Entity Recognition, the texts in both sets were split into chunks, resulting in 113 additional text chunks in the train set and an addition of 17 in the test set. These chunks contained 182 words on average in the train set and 178 words on average in the test set.

Moreover, for Multi-Label Classification, the texts were split into 5503 separate sentences in the train set, consisting of an average of 21 words, and 1310 sentences in the test set, consisting of 20 words on average.

Data descriptives and label distribution after oversampling

For Named Entity Recognition, oversampling resulted in 34 additional duplicates within the train set, and an additional 13 duplicates for in the test set. For these sets, chunking resulted in 119 additional text chunks in the train set and an addition of 26 in the test set. These chunks consisted of an average of 232 words in the train set and 230 words in the test set.

Moreover, for Multi-Label Classification, oversampling resulted in 47 additional duplicate sentences within the train set, and 13 additional duplicate sentences within the test set. The sentences hold an average of 16 words in the train set and 15 words in the test set.

As for the distribution of labels, the effect of oversampling for both NER and MLC is visualized in figure 5 and 6, respectively. Note, that NER shows the frequency of words with certain labels in all the created chunks, whereas MLC shows the presence of these labels in separate sentences. From the figures, it is observed that the number of labels with lower occurrences have indeed increased as expected, but very slightly. It seems that the labels with higher occurrences experience a stronger increase, likely due to the facts that texts contain more common labels alongside less common labels. Therefore, efforts to increase one label often increase the other as well.

Figure 5

Distribution of Word Occurrences per Label for NER, Differing Between Oversampling Method

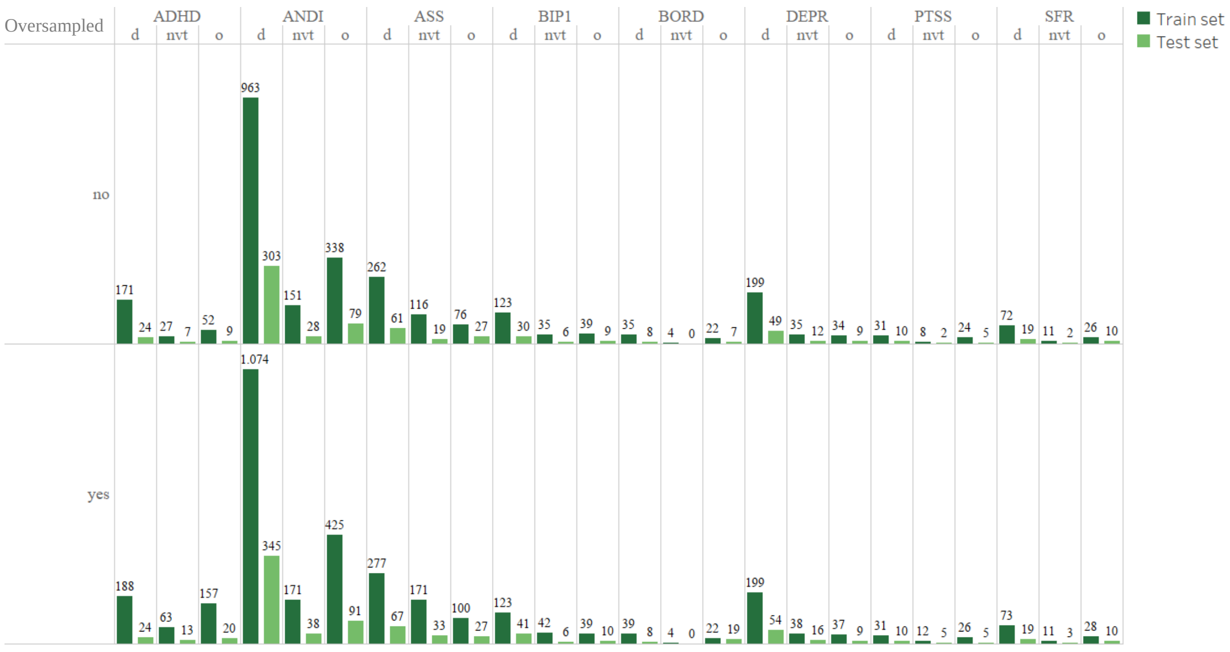
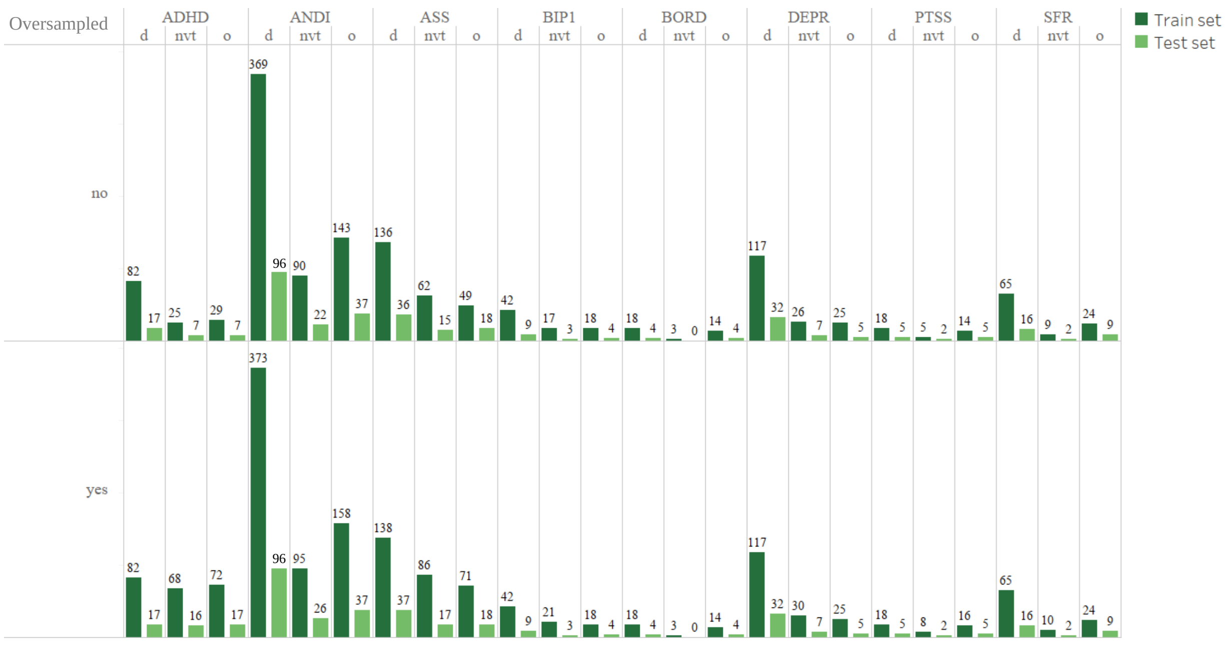


Figure 6

Distribution of Label Presence per Sentence for MLC, Differing Between Oversampling Method



For each of the eight models, the evaluation metrics are displayed in table 5, sorted on the F1 score while taking into account four decimals.

Table 5

Evaluation Metrics of the Fine-Tuned Models

Model	Method	Oversampled	F1 score	Precision	Recall	Accuracy
RobBERT	NER	Yes	0.18	0.16	0.21	0.96
MedRoBERTa.nl	MLC	No	0.17	0.63	0.10	0.99
RobBERT	NER	No	0.17	0.16	0.19	0.96
RobBERT	MLC	No	0.16	0.62	0.09	0.99
MedRoBERTa.nl	MLC	Yes	0.14	0.59	0.08	0.99
MedRoBERTa.nl	NER	No	0.01	0.08	0.12	0.96
MedRoBERTa.nl	NER	Yes	0.01	0.08	0.11	0.97
RobBERT	MLC	Yes	0.01	1.00	0.01	0.99

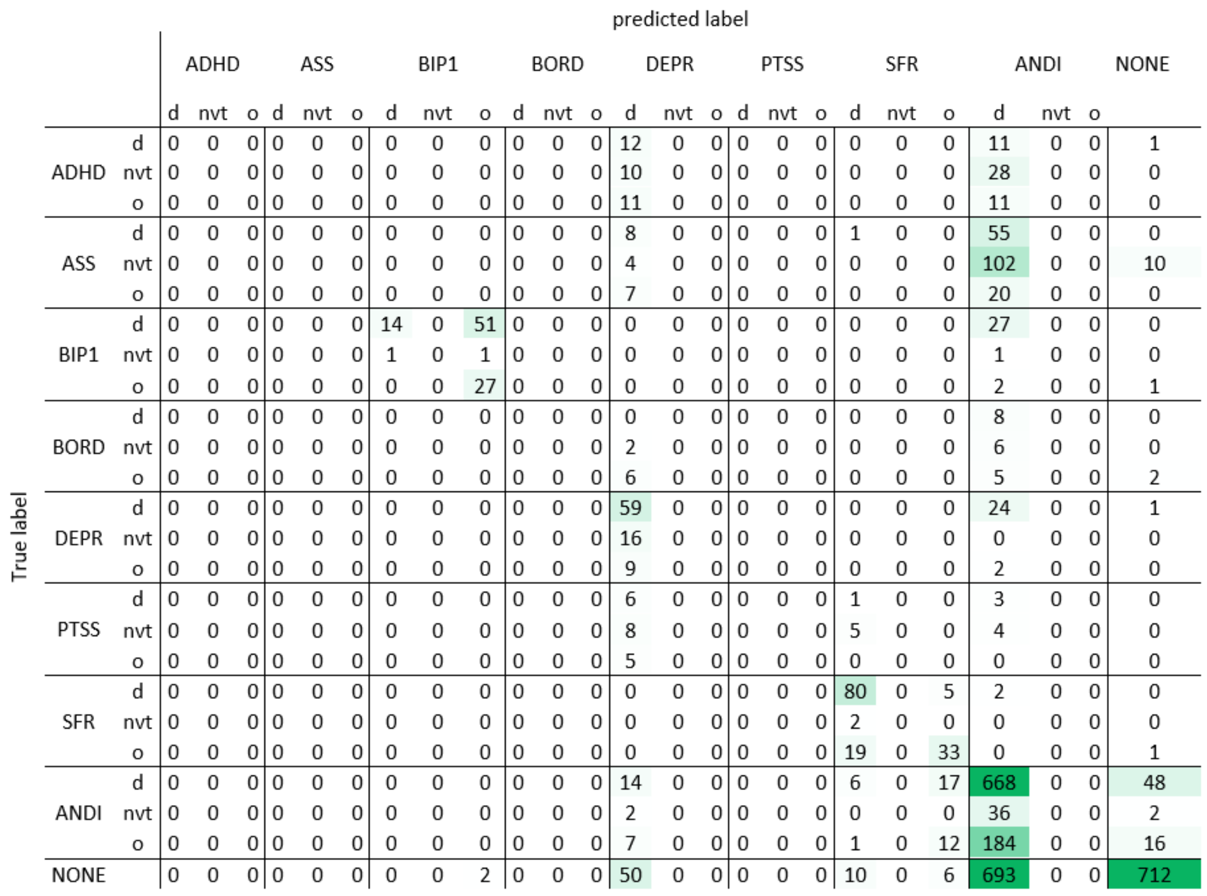
Although evaluation metrics provide a general indication of performance, they do not offer specific practically relevant information about prediction accuracy for individual labels. To address this, the precision and recall of individual labels are visualized for further interpretation. The necessary input needed for these calculations can be retrieved from the confusion matrices that correspond with specific models. Table 5 reveals that MLC models obtain a noticeably higher precision than NER models. To examine the predictions in greater detail and explore this difference further, the precision and recall for each label will be compared between the two best performing models as based on the evaluation metrics of Table 4: RobBERT, NER, oversampled and MedRoBERTa.nl, MLC, not oversampled, which will be referred to as the 'NER' model and 'MLC' model from this point onwards.

For the MLC model, individual confusion matrices will be retrieved for each label, illustrating the prediction quality for both positive and negative classifications. In contrast, NER utilizes a set of predefined classification categories and does not classify the presence or absence per label. This means that the confusion matrix can be displayed within a single visualization, shown in table 6.

After reviewing the confusion matrices for the MLC model, it is revealed that no Positive predictions are made at all for all labels, except the label 'ANDId'. The confusion matrix for 'ANDId' is represented in table 7, whereas the confusion matrix for 'ADHDd' is displayed in table 8, serving as an exemplary table representing the layout of the other confusion matrices that lack positive predictions as well.

Table 6

Confusion Matrix for the Best Performing NER Model

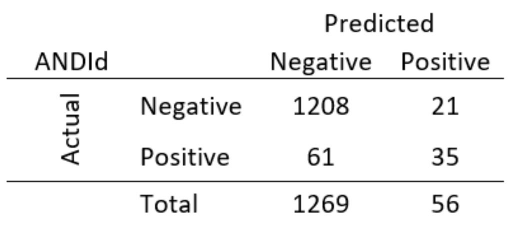


predicted label

True label		ADHD			ASS			BIP1			BORD			DEPR			PTSS			SFR			ANDI			NONE
		d	nvt	o	d	nvt	o	d	nvt	o	d	nvt	o	d	nvt	o	d	nvt	o	d	nvt	o	d	nvt	o	
ADHD	d	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0	0	0	11	0	0	1
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	28	0	0	0
	o	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	11	0	0	0
ASS	d	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	1	0	0	55	0	0	0
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	102	0	0	10
	o	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	20	0	0	0
BIP1	d	0	0	0	0	0	0	14	0	51	0	0	0	0	0	0	0	0	0	0	0	0	27	0	0	0
	nvt	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
	o	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	1
BORD	d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	6	0	0	0
	o	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0	5	0	0	2
DEPR	d	0	0	0	0	0	0	0	0	0	0	0	0	59	0	0	0	0	0	0	0	0	24	0	0	1
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0
	o	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0	0	2	0	0	0
PTSS	d	0	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	1	0	0	3	0	0	0
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	0	5	0	0	4	0	0	0
	o	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0
SFR	d	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80	0	5	2	0	0	0
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0
	o	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	33	0	0	0	1
ANDI	d	0	0	0	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0	6	0	17	668	0	0	48
	nvt	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	36	0	0	2
	o	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	1	0	12	184	0	0	16
NONE		0	0	0	0	0	0	0	0	2	0	0	0	50	0	0	0	0	0	10	0	6	693	0	0	712

Table 7

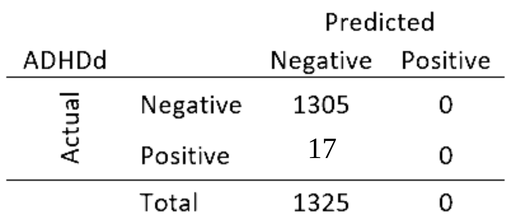
Confusion Matrix of ANDId for the Best Performing MLC Model



ANDId		Predicted Negative	Predicted Positive
Actual	Negative	1208	21
	Positive	61	35
	Total	1269	56

Table 8

Confusion Matrix of ADHDd for the Best Performing MLC model



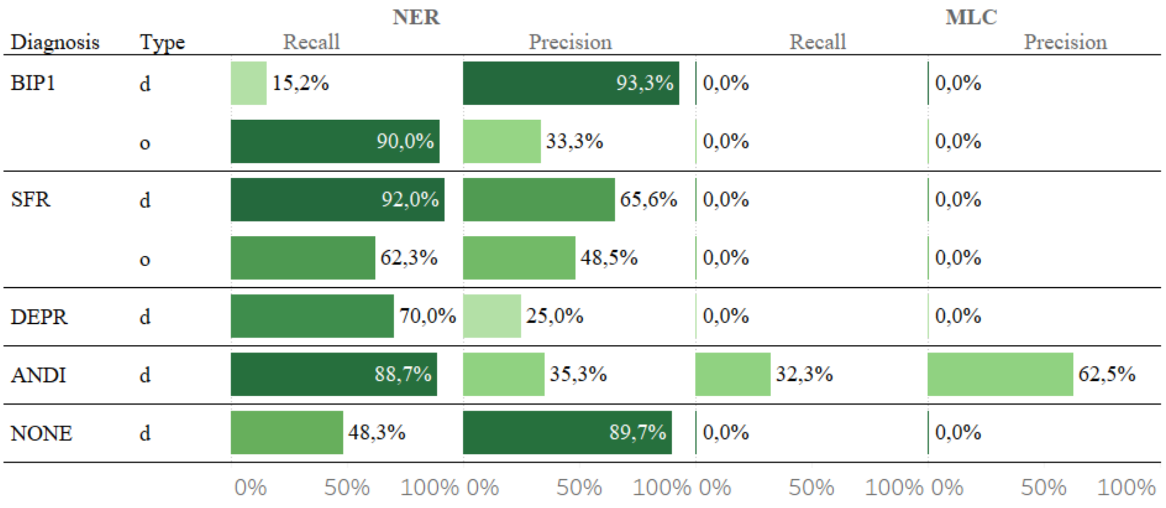
ADHDd		Predicted Negative	Predicted Positive
Actual	Negative	1305	0
	Positive	17	0
	Total	1325	0

The confusion matrices reveal that the NER model makes predictions of a higher variety than the MLC model, even though it also fails to make predictions for the following labels: ADHD, ASS, BORD and PTSS.

In figure 7, the precision and recall of specific labels between the MLC and the NER models are compared. It includes only those labels for which positive predictions were made in at least one of the models, as for labels with no positive predictions in either model, the precision and recall are automatically set to 0.

Figure 7

Precision and Recall of Predicted Diagnoses for the Top-Performing NER and MLC Models



		NER		MLC	
Diagnosis	Type	Recall	Precision	Recall	Precision
BIP1	d	15,2%	93,3%	0,0%	0,0%
	o	90,0%	33,3%	0,0%	0,0%
SFR	d	92,0%	65,6%	0,0%	0,0%
	o	62,3%	48,5%	0,0%	0,0%
DEPR	d	70,0%	25,0%	0,0%	0,0%
ANDI	d	88,7%	35,3%	32,3%	62,5%
NONE	d	48,3%	89,7%	0,0%	0,0%

0% 50% 100% 0% 50% 100% 0% 50% 100% 0% 50% 100%

Based on figure 7, it can be observed that the NER model is able to make more accurate predictions than the MLC model. Therefore, this model is selected as the top-performing method out of the selection of eight models, using NER, RobBERT and an oversampling technique. Unfortunately, the model is unable to score relatively high for both recall and precision on a specific label as the model still makes a substantial number of misclassifications.

Thus, this model is utilized in the next step, where this model is fine-tuned once more on a dataset without subtypes in an attempt to increase its performance. The evaluation metrics of this model are displayed in table 9.

Table 9

Evaluation Metrics of the Top-Performing Model without Types

F1 score	Precision	Recall	Accuracy
0.44	0.42	0.48	0.98

At first sight, the model seems to be able to make better predictions in general based on an increased F1 score. To obtain a more detailed view of their predictions, the confusion matrix of the model is displayed in table 10.

Table 10

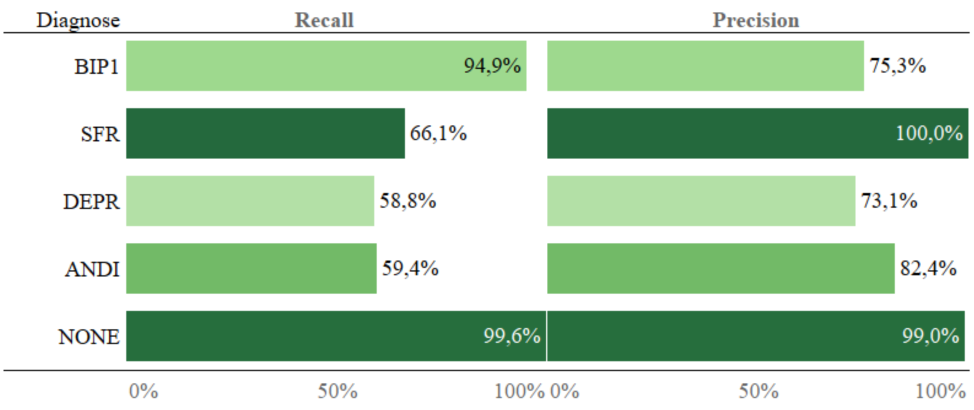
Confusion Matrix of the Top-Performing Model without Types

Label	ADHD	ASS	BIP1	BORD	DEPR	PTSS	SFR	ANDI	NONE
ADHD	0	0	0	0	0	0	0	41	16
ASS	0	0	5	0	7	0	2	101	7
BIP1	0	0	168	0	2	0	5	36	12
BORD	0	0	0	0	11	0	0	6	4
DEPR	0	0	0	0	87	0	0	31	1
PTSS	0	0	0	0	0	0	17	0	0
SFR	0	0	0	0	0	0	156	0	0
ANDI	0	0	0	0	16	0	34	799	121
NONE	0	0	4	0	25	0	22	330	38112

The models appears to improve the performance of the original NER model by eliminating misclassifications within the same diagnoses, but with incorrect types. However, it still struggles with many labels being classified solely under the 'ANDI' label. To once again provide a more detailed comparison of the predictive accuracy per label, the recall and precision of the predicted labels are visualized in Figure 8.

Figure 8

Precision and Recall of Predicted Diagnoses for the Top-Performing without Types



Conclusion

This research aimed to investigate the performance of a Transformer to extract diagnoses from Dutch psychiatric text, differing between the effect of model selection, classification method and oversampling technique.

Classification method

When comparing MLC to NER as the utilized classification method, a clear difference is observed. Although the performance metrics of MLC didn't seem to fall short when compared to the NER models, the higher scores were solely based on the predictions for 'Other diagnosis' with type 'd'. This is caused by the model's inability to make False Positive predictions by making no positive predictions for the other labels at all. This explains why the precision is so high for the MLC models in table 4. The only positive predictions are made for 'ANDId', for which the precision is 62.5%. Based on these observations, it is concluded that the NER model shows more potential in making accurate classifications.

Model selection

After establishing NER as the preferred classification method, model performance can be compared exclusively within this method. A noticeable divide can be seen in terms of model selection, as RobBERT takes the lead in performance over MedRoBERTa.nl. Although the F1 are all generally low, RobBERT's F1 scores seem to be a substantial improvement over MedRoBERTa.nl's, going from 0.09 to 0.18. This aligns with the theory that MedRoBERTa.nl has primarily been trained on technical medical texts, whereas RobBERT may be better adapted to analyzing psychiatric language. This difference could stem from psychiatrists writing their texts in a style similar to the general Dutch language, which resembles the public text data used to train RobBERT. And where MedRobBERTa.nl is more familiar with technical terms that are absent in psychiatry, RobBERT may have been exposed to psychiatric terms through public articles and conversations about mental health.

Oversampling

If we look at the oversampling methods within the NER methods, we can see that it makes no considerable difference for MedRoBERTa.nl, whereas it makes for a slight improvement for RobBERT. However, the difference is too practically small to decide whether this is due to the oversampling technique or due to chance, given the non-uniform outcomes.

Top performing model

Based on these results, a top-performing model was selected. This model implemented the NER method, utilized RobBERT as the pre-trained Transformer and applied an oversampling technique. The model is unable to make any predictions for the following diagnoses: ADHD, Autism Spectrum Disorder, PTSD and Borderline Personality Disorder. However, the model seems to perform quite reasonable on the following diagnostic categories: Bipolar I Disorder (type 'd' and 'o'), Schizophrenia (type 'd' and 'o'), Depressive Disorder (type 'd'), Other Diagnosis (type 'd') and NONE. However, the predictions for every label fail to simultaneously score high on both their precision and recall scores. Meaning that either some instances are missed in the label predictions or the predictions may be unreliable due to misclassifications.

Removing types from labels

After removing the types from the labels, the top-performing model was fine-tuned once more on the task at hand. This implementation increased model performance, increasing the F1 score from 0.18 to 0.44. Predictive accuracy has increased now that the model does not have to struggle with differing between types of the same diagnosis. However, it still fails to make predictions for the same collection of diagnoses by mainly identifying them as 'other diagnoses'.

Discussion

Based on these findings, it seems that the performance of a Transformer in extracting diagnoses from Dutch psychiatric depends on numerous factors. One of which is the training data of the utilized pre-trained transformer. Based on this research, RobBERT comes forward as a potential candidate for future implementation. To increase the performance of a Transformer within the psychiatric field, it should be trained on a vast number of psychiatric texts so it may capture the domain-specific language. However, due to privacy concerns and the need for extensive computational resources, pre-trained models like RobBERT seem to be an adequate alternative. Additionally, NER shows more potential as a classification method than MLC.

The top-performing NER model, utilizing RobBERT as its pre-trained transformer while using an oversampling technique. Although the model scored relatively well on Bipolar I Disorder and Schizophrenia, it can be concluded that it cannot be implemented in practice due to unreliable results stemming from shortcomings in either precision or recall. The model performance on these labels increases when the model does not have to distinguish between the types of the diagnoses anymore. This may however reduce practical utility, as it does not provide the context in which the diagnoses is discussed. Both of these models can serve as a base for future improvements.

Misclassifications of labels

An interesting phenomenon is observed in both the original top-performing NER model and its revised version after the removal of subtypes in the labels. It can be observed that a substantial number of correct predictions were made for Bipolar I Disorder (type ‘o’) and Schizophrenia (type ‘d’). This observation is remarkable, as there were relatively few instances present in the training data that belonged to these categories. It is expected that the model would excel in classifications with more instances to learn from, such as Autism Spectrum Disorder (ASS). However, contrary to expectations, no predictions for ASS are observed at all. This is likely caused by the use of terminology and the context in which it is written. It seems like Bipolar I Disorder and Schizophrenia are discussed in specific and distinguishable situations, allowing for the model to pick up on these more easily. The opposite might be true for diagnoses like ASS, ADHD, Borderline Personality Disorder and PTSD, which are often identified as ‘other diagnoses’ (ANDI). These labels seem to be discussed in similar contexts, such as events or symptoms, making them harder to distinguish. The tendency for these labels to be classified under ANDI is likely due to the model being more adapted to it, having been exposed to this label more frequently than the others.

Another reason could be the usage of the word ‘stoornis’ (disorder), which frequently occurs within both diagnosis categories. However, this remains speculative as the word ‘stoornis’ also occurs relatively often within the diagnosis of Bipolar I Disorder (BIP1), for which an adequate number of predictions was correct.

Future research

This research aimed to provide a baseline for extracting diagnoses from Dutch psychiatric text with Large Language Models (LLMs). After fine-tuning a selection of eight models, the top-performing model was selected with the best predictive performance. The model performs adequately, but leaves enough room for improvement. To increase performance, it is recommended to keep experimenting with different pre-trained models and oversampling techniques. Implementing other well-performing Dutch pre-trained Transformers could also prove to be effective. Towards the end of this research, BelabBERT (Wouts et al., 2021), a pre-trained model known for its effectiveness with Dutch psychiatric data, was identified. Although it was not initially included in this study due to unfamiliarity with its existence, it should be considered for future research endeavors.

Another area of improvement regards the frequency and distribution of the training data. The model struggles when having to distinguish between the types of diagnoses, most likely due to contextual similarities and insufficient training data. Gathering more annotated data can potentially enhance the model's ability to distinguish between these types. Another approach could involve separating the task of extracting diagnoses and classisying their respective types, allowing the model to focus more deeply on each task individually rather than concurrently.

Next to this, these is an overrepresentation of instances labeled as ANDI, potentially leading to many labels being falsely predicted as ANDI. This is likely due to similarities in terminology and the observed class imbalance. The potential impact of terminology on the classification of specific labels could be an intriguing subject for psychiatric research to explore, aiming to uncover any observable differences in how professionals document certain diagnoses and potentially the underlying reasons behind these differences. As for the label imbalance, methods need to be implemented that handle this effectively. In the research, an oversampling method was implemented. While no clear effect has been found regarding the oversampling method, no negative impacts were observed either. Further testing is encouraged, as the technique has only shown a slight increase in the frequency of uncommon labels. Ideally, the technique should be adapted for implementation without increasing the frequency of common labels in multi-labeled data.

Based on these considerations, this thesis provides the groundwork for the integration of LLMs within the field of psychiatry.

Acknowledgements

Firstly, I want to express my gratitude to my daily supervisors, Rosanne J. Turner and Magali de Rooy. Your regular feedback has been invaluable in shaping my work, and I truly appreciate your patience with my countless questions and sending me lightning-fast responses to my emails. I also owe a big thank you to Bram van Es, who generously reviewed my Python script, even though we've never met. Your encouragement gave me the confidence to run my code without that nagging feeling of doubt. Thirdly, a shout-out to Adriaan Hoogendoorn, Lucas Veenema, and Marjolijn Das for always being there to provide feedback on my thesis and its corresponding poster whenever I needed it. Your support has been instrumental and greatly appreciated. Lastly, I want to thank my mom. You may not understand any of this, but you have been consistent in your loving mental support.

Declaration of AI and AI-assisted technologies in the writing process

During the preparation of this work the author used ChatGPT only in order to find synonyms for words, refine the structure of the text and to generate code to pre-process the data properly. After using this tool, the author reviewed and edited the content as needed an takes full responsibility for the content of the publication.

References

Abadji, J., Ortiz Suarez, P., Romary, L., & Sagot, B. (2022, January 17). Towards a Cleaner Document-Oriented Multilingual Crawled Corpus. https://arxiv.org/abs/2201.06642

Alsentzer, E., Murphy, J., Boag, W., Weng, W.-H., Jindi, D., Naumann, T., & McDermott, M. (2019, June). Publicly available clinical BERT embeddings. *Proceedings of the 2nd Clinical Natural Language Processing Workshop*. https://doi.org/10.18653/v1/w19-1909

Benayas, A., Sicilia, M. A., & Mora-Cantallops, M. (2024). *A Comparative Analysis of Encoder Only and Decoder Only Models in Intent Classification and Sentiment Analysis: Navigating the Trade-Offs in Model Size and Performance*. https://doi.org/10.21203/rs.3.rs-3865391/v1

Barney, N. (2023, March 10). *What is named Entity recognition (ner)?: Definition from TechTarget*. TechTarget. https://www.techtarget.com/whatis/definition/named-entity-recognition-NER

Clearview Treatment Programs. (2019, November 20). *Why we need to talk about borderline personality disorder*. https://clearviewtreatment.com/resources/blog/why-we-need-to-talk-about-borderline-personality-disorder/#:~:text=People%20with%20BPD%20are%20often,this%20is%20no%20longer%20true.

Computational Lexicology & Terminology Lab VU. (2023). *MedRoBERTa.nl* (Revision 13f225b). Hugging Face. https://huggingface.co/CLTL/MedRoBERTa.nl. https://doi.org/10.57967/hf/0960

Corrigan, P. W., & Watson, A. C. (2002, February). *Understanding the impact of stigma on people with mental illness*. World psychiatry. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1489832/

Dai, X., Chalkidis, I., Darkner, S., & Elliott, D. (2022, October 25). Revisiting transformer-based models for long document classification. *Findings of the Association for Computational Linguistics: EMNLP 2022*. https://doi.org/10.18653/v1/2022.findings-emnlp.534

Dao, T. (2023, January 13). *Flashattention: Fast transformer training with long sequences*. Hazy Research. https://hazyresearch.stanford.edu/blog/2023-01-12-flashattention-long-sequences

Delobelle, P., Winters, T., & Berendt, B. (2020, September 16). RobBERT: A Dutch RoBERTa-based language model. https://doi.org/10.48550/arXiv.2001.06286

Delobelle, P. (2023). *robbert-v2-dutch-base* (Revision 271b8bf). Hugging Face. https://huggingface.co/pdelobelle/robbert-v2-dutch-base. https://doi.org/10.57967/hf/1425

de Vries, W., van Cranenburgh, A., Bisazza, A., Caselli, T., Noord, van, G., & Nissim, M. (2019). BERTje: A Dutch BERT Model. *ArXiv,* Article 1912.09582. https://arxiv.org/abs/1912.09582

Hackeling, G. (2017). *Mastering machine learning with scikit-learn: Learn to implement and evaluate machine learning solutions with scikit-learn*. Packt Publishing.

Harris, Z. S. (1954). Distributional structure. *WORD*, *10*(2–3), 146–162. https://doi.org/10.1080/00437956.1954.11659520

Hugging Face. (2024). *Hugging Face Transformers*. https://huggingface.co/docs/transformers/index

Jurafsky, D., & Martin, J. H. (2019). Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition (3rd ed.). Prentice Hall.

Kwan, M. (2023, September 6). *Finding the optimal number of dimensions for word embeddings*. Medium. https://medium.com/@matti.kwan/finding-the-optimal-number-of-dimensions-for-word-embeddings-f19f71666723

Kundu, R. (2022, December 16). *F1 score in Machine Learning: Intro & Calculation*. V7. https://www.v7labs.com/blog/f1-score-guide

Legal text. General Data Protection Regulation (GDPR). (2018, May 25). https://gdpr-info.eu/

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H., & Kang, J. (2019). BioBERT: A pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, *36*(4), 1234–1240. https://doi.org/10.1093/bioinformatics/btz682

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019, July 26). RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://arxiv.org/abs/1907.11692

Martin, L., Muller, B., Ortiz Suárez, P. J., Dupont, Y., Romary, L., de la Clergerie, É., Seddah, D., & Sagot, B. (2020, July). Camembert: A tasty French language model. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. https://doi.org/10.18653/v1/2020.acl-main.645

Menger, V., Scheepers, F., van Wijk, L. M., & Spruit, M. (2018). Deduce: A pattern matching method for automatic de-identification of Dutch Medical Text. *Telematics and Informatics, 35*(4), 727–736. https://doi.org/10.1016/j.tele.2017.08.002

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv Preprint arXiv:1301.3781*.

Nagwa. (n.d.). *Lesson explainer: Vectors in terms of fundamental unit vectors mathematics.* https://www.nagwa.com/en/explainers/578165351487/

Nederlandse vereniging voor psychiatrie. (2014, April). Vijftien veel gestelde vragen over DSM-5. https://www.nvvp.net/stream/qa-lijst-dsm-5

Raschka, S. (2023, June 17). *Understanding encoder and decoder llms*. Ahead of AI. https://magazine.sebastianraschka.com/p/understanding-encoder-and-decoder

Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019, October 2). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. https://arxiv.org/abs/1910.01108

Shrivastava, I. (2020, December 17). *Handling class imbalance by introducing sample weighting in the loss function*. Medium. https://medium.com/gumgum-tech/handling-class-imbalance-by-introducing-sample-weighting-in-the-loss-function-3bdebd8203b4

Si, Y., Wang, J., Xu, H., & Roberts, K. (2019). Enhancing clinical concept extraction with contextual embeddings. *Journal of the American Medical Informatics Association*, *26*(11), 1297–1304. https://doi.org/10.1093/jamia/ocz096

Stein, D. J., Lund, C., & Nesse, R. M. (2013). Classification Systems in psychiatry. *Current Opinion in Psychiatry*, *26*(5), 493–497. https://doi.org/10.1097/yco.0b013e3283642dfd

Tunstall, L., Werra, L. von, & Wolf, T. (2022). *Natural language processing with transformers*. O'Reilly Media, Incorporated.

Utrecht University. (2024). *Utrecht University Student Theses Repository.* https://studenttheses.uu.nl/

Van Ginkel, E. (2024). *Annotation of Psychiatric Data.* [master's thesis, Utrecht University]. https://studenttheses.uu.nl/

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. Advances in Neural Information Processing Systems, 30. https://proceedings. neurips.cc/paper files/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aaAbstract.html

Verkijk, S., & Vossen, P. (2021). MedRoBERTa.nl: A language model for Dutch electronic health records. *Computational Linguistics in the Netherlands Journal*, 11, 141–159

Wouts, J., de Boer, J., Voppel, A., Brederoo, S., van Splunter, S., & Sommer, I. (2021, June 2). BelabBERT: a Dutch RoBERTa-based language model applied to psychiatric classification. https://arxiv.org/abs/2106.01091

Appendix

The appendix consists of the unshown confusion matrices per label of the best performing MLC model. Mind, the model was run a second time to retrieve these confusion matrices. Due to the non-uniform outcomes of the train test split, these numbers might differ slightly with the reported label distribution for MLC.

```
ADHDo
[1318    0]
  [   7    0]

ADHDd
 [1305    0]
  [  20    0]

ADHDnvt
 [1319    0]
  [   6    0]

ASSo
 [1307    0]
  [  18    0]

ASSd
 [1289    0]
  [  36    0]

ASSnvt
 [1310    0]
  [  15    0]

BIP1o
 [1321    0]
  [   4    0]

BIP1d
 [1316    0]
  [   9    0]

BIP1nvt
 [1322    0]
  [   3    0]
```

```
BORDo
 [1321    0]
  [   4    0]

BORDd
 [1321    0]
  [   4    0]

DEPRnvt
 [1325    0]
  [   0    0]

DEPRo
 [1320    0]
  [   5    0]

DEPRd
 [1296    0]
  [  29    0]

DEPRnvt
 [1318    0]
  [   7    0]

PTSSo
 [1320    0]
  [   5    0]

PTSSd
 [1320    0]
  [   5    0]

PTSSnvt
 [1323    0]
  [   2    0]

SFRo
 [1316    0]
  [   9    0]

SFRd
 [1309    0]
  [  16    0]
 SFRnvt
[1323    0]
  [   2    0]

ANDIo
 [1288    0]
  [  37    0]

ANDId
 [1208   21]
  [  61   35]

ANDInvt
 [1303    0]
  [  22    0]
```