Master Thesis

Using Semi-supervised Learning to Automatically Annotate Dutch Medical Notes for Patients' Functioning Levels

Cecilia B. Schramm

a thesis submitted in partial fulfilment of the requirements for the degree of

MA Linguistics
(Text Mining)

Vrije Universiteit Amsterdam

Computational Lexicology and Terminology Lab
Department of Language and Communication
Faculty of Humanities

Supervised by: Piek Vossen
2^{nd} reader: Hennie van der Vliet

Submitted: October 30, 2023

Abstract

This thesis set out to test whether a semi-supervised learning approach to fine-tuning a RoBERTa-based classification model fine-tuned on COVID-19 data would procure good enough results that extensive and costly human annotation would no longer be necessary to obtain new training and testing data. This was done by assessing whether it was the quality or the quantity of new training data that improved the classifier's performance, for which large amounts of available but unannotated data was labeled by the classification model, divided into "high" and "low" quality data, and used to train that same model. Due to the different types of data used to train the model, however, it seems the model un-learned previously learned patterns when trained on too much data, and returned more false negatives when trained on "low" quality data. This thesis also includes suggestions on how to construct a more robust and reliable model, so that future A-PROOF interns may use the findings of this paper to obtain more labeled data with ease.

Declaration of Authorship

I, Cecilia Benedictine Schramm, declare that this thesis, titled *Using Semi-supervised Learning to Automatically Annotate Dutch Medical Notes for Patients' Functioning Levels* and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a degree degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Date: October 30, 2023

Signed: Cecilia B. Schramm

Acknowledgments

I would like to thank Piek Vossen for being a fantastic and always thorough supervisor, Cecilia Kuan for her endless patience with my questions, Meruyert Nurberdikhanova for her tireless technical support, Bertjan for his never-ending availability, Sal Singh for their boundless emotional and professional support, and the A-PROOF team for the amazing opportunity I was given to conduct these experiments. Thank you all!!

List of Figures

Contents

Chapter 1

Introduction

This thesis was written in collaboration with the A-PROOF[1] project at the Amsterdam UMC (University Medical Centers) as a continuation of previous master students' theses, as well as a support project for future students working on this project.

The A-PROOF team's project is building a classifier that can read Dutch healthcare notes, written by either primary or secondary healthcare providers, and automatically categorize these notes into the WHO functioning categories of the *International Classification of Functioning, Disability and Health* (ICF), together with a corresponding level of either 0-4 or 0-5. The ICF is a framework used by the WHO to give precise descriptions and measurements of a patient's functioning levels. Some of the domains in there are, for example, energy level, attention functions, walking, and more. Please see figure A.1 in the appendix for a complete overview of the domains and levels that were used by the A-PROOF team for this project. The potential use of such a classifier varies greatly, from observing the long-term effects of diseases or new drugs, to predicting recovery patterns, and more. The A-PROOF team's final goal is to create a patient recovery timeline with this classifier, for recovery functioning and clinical treatments. By doing it this way, the team can create meta-data for patients outside of the standardized way of categorizing and may curate a more stream-lined and easily readable database.

My thesis[2] was run on a classifier (built by Kim (2021)) that is an extended, fine-tuned language model which was developed by Verkijk and Vossen (2021) and was a pre-trained and fine-tuned version of a RoBERTa-based model. This language model was eventually titled *MedRoBERTa.nl* and was fine-tuned in a supervised fashion and mostly trained on COVID-19 data from secondary healthcare providers, i.e. hospital notes. Galjaard (2022) and Badloe (2020) then took the classifier by Kim (2021) and tested and trained it for evaluating primary healthcare notes and non-COVID healthcare notes, respectively. Although both researchers were able to answer their research questions, they themselves claimed that their results were unreliable or did not even pass the baseline experiment. Both researchers stated that this was mostly due to the lack of annotated training data, which is where I gathered the inspiration for my own thesis from. Seeing where the previous researchers' projects could have gone, had they had enough training data, made me realize what this project needed the most. The A-PROOF team provided millions of hospital notes, though only a fraction

[1] `https://cltl.github.io/a-proof-project`

[2] The GitHub repository of this thesis can be found at `https://github.com/cltl-students/Cecilia_Schramm_ICF_semi_supervised_learning`

of those were annotated by human annotators, which makes training a new model rather difficult. The more data, the more reliable and well performing a model usually is, but the annotation process is a long and complicated process. However, thanks to semi-supervised learning methods, classifiers do not always need large amounts of labeled data to perform well, especially if large amounts of unlabeled data are available.

As mentioned above, there are millions of unlabeled hospital notes available, though of course each note differs in quality, which is determined by how clearly a note depicts a certain category and the corresponding level of said category. Nevertheless, there is no guarantee that semi-supervised learning can improve this classifier's performance, since there are many steps and decisions along the way that will heavily determine the outcome. Not to mention the data itself plays a large role in the outcome, as both quantity and quality of data influences what the model learns. That is why this thesis aims to answer the following research question:

Research Question: Does using semi-supervised learning to train a model improve the model's performance in automatically annotating unlabeled hospital notes?

In order to fully explore this question, I will also be focusing on these **sub-questions** within my main question:

- How much does the quality of the training data influence the model's performance?
- How much does the quantity of the training data influence the model's performance?
- What specific categories do not benefit from semi-supervised learning?

All three of these sub-questions will be answered in a single continuous process that will be testing all three simultaneously. The process starts by using TF-IDF ("Term Frequency - Inverse Document Frequency") on the entire corpus to receive the most relevant keywords. These keywords will then be used to gather enough relevant notes from the corpus and, after dividing them into sentences, have a base classifier run over them and include a confidence score in the predictions. "High" and "low" quality data will be gathered from that data based on certain confidence score criteria, which will then be used to further fine-tune the base classifier. This will be done in parallel, once with with high quality, once with low quality data. The training sets will be divided into thirds and the base classifier trained one by one with each third, still parallel for high and low quality data. At the end, one model will be trained with all existing newly model-labeled data. Each intermediate model will be evaluated on a development set.

Through this process, sub-question 1 and 2 will be examined in a direct comparison, while sub-question 3 will be explored slowly throughout the whole process. I believe that these 3 sub-questions will help me investigate my main research question deeply, as each of them will give me a different insight into whether or not the semi-supervised learning method did indeed improve the model's performance in annotating unseen hospital notes.

1.1 Thesis Structure

My thesis is structured into 7 chapters, **Introduction**, **Related Work**, **Data and Annotation**, **Corpus Analysis**, **Experiments**, **Results and Error Analysis**, and **Discussion and Conclusion**. Chapter 2 will go into detail regarding the processes of semi-supervised learning, unsupervised learning, and pseudo-labeling, so that readers may get an understanding of the processes used in my experiments, as well as the conception of *MedRoBERTa.nl*. Chapter 3 will explain the labeled data availability and annotation process, with detailed explanations regarding previous work on this project. Chapter 4 is an analysis of the unlabeled corpus provided for this thesis, especially in regards to how I gathered and what I deemed "high" and "low" quality data. Chapter 5 is the detailed description of my experiments and how exactly they were carried out, while chapter 6 will be the reported results of these experiments, as well as an in depth error analysis of where the system failed on a holistic scale. Chapter 7 will be the discussion and conclusion of my thesis, meaning I will be explaining what the system's performance means for my research, and finally I will be summarizing my thesis and suggesting future work that could be done in order to further improve on my work.

Chapter 2

Related Work

This chapter will be describing the general processes that will be used in my thesis. It will be divided into 3 sub-sections: Section 2.1 will be describing what unsupervised learning, semi-supervised learning (SSL), and pseudo-labeling are. Section 2.2 will be briefly explaining the language model this entire project is based on, *MedRoBERTa.nl*, while section 2.3 will be explaining how this model was then fine-tuned on medical notes from COVID-19 patients to be used for ICF classification with levels. This is the model used in all of my experiments, as my thesis is not only a continuation of previous students' work on the project, but also an attempt to improve future works within the project.

2.1 Unsupervised Learning, Semi-Supervised Learning, and Pseudo-Labeling

The ideal machine learning process would, naturally, be performed on a plethora of labeled, varied data and an equally diverse, yet potentially smaller, test set. In supervised learning, it has been proven to be the most effective way of training a system and consistently providing the best results. However, that is the ideal scenario. In truth, there are many instances where that is simply not the case and the researchers are met with large amounts of unlabeled data and oftentimes either very little labeled data or even none at all, mostly because the process of human annotation is expensive, time-consuming, and complicated. That is where *unsupervised* or *semi-supervised learning* come in. While those two are ways of training a machine on very little to no labeled data, *pseudo-labeling* is a process that happens within semi-supervised learning (SSL).

2.1.1 Unsupervised Learning

The idea described above, with lots of labeled training data as well as labeled test data, is called *supervised learning*, where a machine is trained and tested on labeled data only. The opposite of that, so to speak, would be *unsupervised learning*. Here, a machine is trained on no labeled data whatsoever, which can be quite useful when no labeled data exists at all. Though it might seem counterproductive or even impossible to train a machine on no labeled input – as that would mean no feedback for the system to compare its predictions to – Ghahramani (2004) argue that unsupervised learning is about finding patterns in the data that go beyond structureless noise. It is through these patterns that the machine can construct representations of the input data and

learn from it. Hastie et al. (2017), however, also point out that with this strategy, it is much harder to judge the outputs of the system, as there is no loss function that can be applied to the expected and predicted output. Therefore, Hastie et al. (2017) say, one must resort to experimental approaches for judging the output's quality and validity, which has led to a myriad of methods of judging the output of unsupervised learning, as "effectiveness is a matter of opinion and cannot be verified directly" (Hastie et al., 2017, p. 487). The method used to verify the quality and validity of the output in this thesis will be the testing on the same test set as the other experiments will be tested on.

2.1.2 Semi-Supervised Learning

Semi-supervised learning is often described to lie in the middle between supervised and unsupervised learning. According to Hady and Schwenker (2013), SSL "refers to methods that attempt to take advantage of unlabeled data for supervised learning (semi-supervised classification)" (Hady and Schwenker, 2013, p. 217). What this means is that SSL strives to improve one of the processes by using information that is normally associated with the other one, such as using unlabeled data to improve a classification process (Van Engelen and Hoos, 2020). It is performed with large amounts of unlabeled data and small amounts of labeled data, which are both used for training.

There are numerous methods of semi-supervised learning, all with their own strengths and weaknesses based on the task they're being used for. But, according to Triguero et al. (2015), no matter the task, there are multiple common properties throughout all SSL techniques that define them all, starting with their *addition mechanism*. The first one they introduce is **incremental**, which is a method that step-by-step labels instances of the unlabeled dataset and, given that they pass certain criteria, adds the most confident of these newly labeled instances to the overall labeled dataset. Naturally, it is highly important in this method how the confidence score for each label is determined, as all future predictions depend on this threshold. Equally important in this method, according to Triguero et al. (2015), is the number of training examples added this way. This could either be defined as a parameter of the method, i.e. a constant value, that may or may not be independent from the classes of the project, or it could be defined as a value proportional to the number of classes in the labeled training set. Though this method trains faster than non-incremental methods, it also has the potential of adding false predictions to the labeled class labels, since there are many combinations of datasets and parameters that need to be considered (Triguero et al., 2015).

The second addition mechanism Triguero et al. (2015) mention is the **batch** mechanism. Here, instead of adding each instance one-by-one once they meet the criteria, instances are collected and later added to the labeled set as a batch. The advantage of this method is that these batches can "reprioritize the hypotheses obtained from labeled samples" (Triguero et al., 2015), but they also take much longer than incremental methods.

Lastly, Triguero et al. (2015) describe the **amending** method, which was originally introduced as a counter method of the incremental method's main drawback. In this method, the entire labeled set is chosen as the set that will be enlarged by the pseudo-labels and the algorithm can add or even remove instances based on the set criteria. This means the algorithm allows for amendment of already performed actions, which in turn means this algorithm focuses most on creating pseudo-labeled datasets with high

accuracy. The greatest drawback of this method, however, is the large computational power it requires compared to the incremental and batch approaches (Triguero et al., 2015).

Another important SSL property Triguero et al. (2015) mention in their paper that I needed to consider in my thesis was the stopping criteria, which describes the mechanism that signals the self-labeling process to stop. Triguero et al. (2015) mention three main processes for this; the simplest one being signaling to stop when every instance from the unlabeled dataset has received a label. The second one is selecting only a set amount of instances from the unlabeled dataset and labeling those, which tends to outperform the first method, but it has a predetermined number of iterations and cannot be adapted to the number of instances in the dataset. The third stopping method is to stop the process when the classifier's hypotheses do not change anymore, i.e. the error rate has leveled out. Though this method does not keep falsely labeled instances out of the increasing labeled set, it does limit the amount of unlabeled instances that are added to the increasing labeled set (Triguero et al., 2015).

2.1.3 Pseudo-Labeling

Pseudo-labeling is a process already touched upon in the previous section, where a classifier, through various techniques, labels unlabeled data itself (the so called "pseudo-labels"), adds those to the gold-labeled training set, and then uses that ever increasing labeled dataset to train itself iteratively. There are various methods to pseudo-labeling, such as pseudo-labeling with Hermite polynomial expansions, curriculum labeling, and curriculum pseudo-labeling, but due to time and computational constraints, I will only be working with the very classic approach Lee et al. (2013) explain in their work. Here, an instance is added to the labeled dataset if their label confidence score exceeds a certain predetermined threshold set by me (see chapter 4 for detailed information on said threshold). It is a rather simple approach that allows for interesting observations to be made, as instances around the threshold can be inspected and analyzed for further understanding of the classifier's performance. Further, there are practical factors that make it unfeasible for me to run an automated iterative SSL approach, which is why I will be doing that manually. First, the medical server RAM capacity is not big enough to repeatedly run large files without interruption, which would be the case here. Second, the data cannot leave the server due to privacy issues, so I am reliant on working on the server and do not have constant access to it. Third, I will be using a large language model, which would increase the automated iterations' run time by an impractical amount. And fourth, as this thesis is written within a time constraint of a few months, I simply do not have the time to explore the rather extensive process of automatically iterative SSL.

2.2 MedRoBERTa.nl

In an attempt to create a Dutch language model for the medical domain, Verkijk and Vossen (2021) conducted experiments to find out how to best go about this. In the end, they found that building a model from scratch through training on "Dutch hospital notes, sourced from EHRs [Electronic Hospital Records]" (Verkijk and Vossen, 2021) outperforms general Dutch language models (BERTje, RobBERT, Multilingual BERT), both in pre-training and fine-tuned.

To have a greater pool of comparison for their work, Verkijk and Vossen (2021) built two models, one was an extended version of the RobBERT model that received continued pretraining on domain-specific text, and one was a model built from scratch with a random initialization, also trained on domain-specific vocabulary. As mentioned above, the from-scratch model performed much better and was thus dubbed *MedRoBERTa.nl* and presented as the final model.

2.3 Kims (2021)'s Model

In collaboration with the A-PROOF team, working towards the same goal as mentioned earlier, Kim (2021) fine-tuned the *MedRoBERTa.nl* model to be able to classify the sentences in a hospital note as their corresponding ICF category, as well as their level, which was not something the original *MedRoBERTa.nl* model was trained for. For this, Kim (2021) fine-tuned the model on COVID-19 notes from secondary healthcare providers and curated a pipeline in which the hospital notes fed into it are anonymized, split into sentences, and returned with their ICF category and level. This was done in a two-step-process, in which Kim (2021) first fed the individual sentences into a category identification and classification model, then used 9 different regression models to determine each labeled sentence's ICF category *level*, one model for each category. Out of the more than 100 ICF categories, the 9 chosen due to their relevance in the COVID-19 research were: energy level (abbreviation ENR), attention functions (ATT), emotinal functions (STM), respiration functions (ADM), exercise tolerance functions (INS), weight maintenance functions (MBW), walking (FAC), eating (ETN), and work and employment (BER). Each of them came with an ICF code (e.g. *b1300*) and a functioning level scale from 0-4 or 0-5, where 0 indicates no functioning at all and 4/5 indicates full functioning. It is these 9 categories that my thesis will also be working with.

For Kim (2021)'s model, around 6000 notes were annotated according to the annotation guidelines described in chapter 3, and split into training, testing, and development sets. These were then used for both the classification model in the first step, as well as the regression models in the second step. In the first step, the model indicates whether a certain domain is present or absent in the current sentence through outputting either a 1 (present) or a 0 (absent) for each individual domain. The final output would look something like [0, 1, 0, 1, 1, 0, 0, 0, 0] if, for example, three different domains were present in one sentence. This sentence was then sent to each corresponding regression model (three in this example case) and given a level label. Once this process was done, all sentences of one note were assembled back into their original note and their sentence-level scores were accumulated into one large note-level score. This note-level score was important because healthcare providers are more interested in that score.

Overall, this classifier pipeline yielded quite good results in all but two categories, ATT and BER. With sentence-level F1 scores of 0.58 and 0.35 respectively, it is clear the model struggled quite a bit with these two categories. For both of them, as well as all other categories, the precision was higher than the recall, which means that the model did not often find sentences that should belong to either category, but when it did, it labeled them correctly. This is not surprising, as both categories had much less training data than the other ones, due to the low availability of those in the notes chosen for annotation. The overall high precision scores on all categories, however, show that the model has understood how to distinguish the different categories from one another,

but struggles with recognizing all categories in the input data (Galjaard, 2022). As a note of caution, Kim (2021) mentions in her report that there was an annotation issue regarding the INS category, as the definition for this category was revised in the middle of the project. But because of challenges related to resources, this thesis does not incorporate revised INS instances and the category will be treated as any other, though low performance scores in this class (both in this and Kim (2021)'s project) should be regarded with this information in mind.

Overall, Kim (2021)'s model performed quite well in all its classification tasks. However, when the previous students Badloe (2020) and Galjaard (2022) tried to fine-tune the model to their tasks, their outcomes were not as good as they had hoped. Badloe (2020) tried to perform domain adaptation on the model, moving the domain from COVID-19 data to lung and gastrointestinal cancer notes, but her baseline results remained the best ones, despite different learning rates and freezing of layers. Badloe (2020)'s baseline was using Kim (2021)'s classifier *without* adapting it to the target data and she herself says in her conclusion "the single most determining factor in model performance [is]: the ICF- distribution of the source training data" (Badloe, 2020). Galjaard (2022) wanted to explore something similar, though his domain adaption went from the secondary healthcare provider notes to primary healthcare provider notes, which – despite both being about COVID patients – were written in different language styles. Some of the final remarks were "The most obvious way in which to increase stability is procuring a larger data set, so the results can be evaluated in a proper manner" (Galjaard, 2022), which seems to mirror Badloe (2020)'s opinion regarding the improvement of the process. It becomes quite clear through this previous research that procuring more labeled data is highly important for the future success of this project.

Chapter 3

Data and Annotation

According to Kim (2021), the A-PROOF team provided us with around 8 million hospital notes from the two Amsterdam UMC locations, the Academic Medical Center (AMC) and the VU Medical Center (VUmc). Half of those notes were from both locations from 2017, 2 million of those notes were from 2018 from the AMC location only, and the last 2 million of those notes were from both locations again, from the first three quarters of 2020. Since the A-PROOF team had been particularly interested in studying the effects of COVID-19 on patients, the data was split in notes about COVID-19 patients (cov-2020) and notes about non-COVID-19 patients (non-cov-2020). Since annotation is a long, tedious, and expensive process, only a subset was selected to be annotated for gold labels. This chapter will go into detail regarding the annotation process (section 3.1) and the final annotated notes (section 3.2), as described by Kim (2021).

3.1 Annotation Process

As Kim (2021) describes in her report, the goal was to obtain 15,000 labeled notes. This annotated data was then to be used as the training, development, and test set, which is why she laid out criteria the notes should ideally have:

- Enough sentences with clear categories and levels ("positive examples")
- A balanced distribution of all 9 categories
- Diverse wording to capture all possible terminology that could be used to describe the categories

So as to not have to go through the million of notes by hand to select the notes that meet these criteria, a keyword-based search system was built. For this, lists of keywords for each category were created by professional team members using their expertise. Then, within this system, multiple filters could be determined, such as: the percentage of the notes that should contain the predetermined keywords (since searching only for notes with these keywords could risk overfitting), the specific categories that should be searched (to balance out underrepresented categories), the minimum amount of categories the note should contain (to make sure enough appropriate sentences are found), the proportion of COVID-19 patient notes, and the type of note it was, of which there were 60 in total. Figure 3.1 shows the parameters set for all keyword searches throughout Kim (2021)'s project.

Batch	% COVID	% Kwd	Kwd version	Matched doms	Min matched doms	Note types
w14 - w15	0.5	0.8	v2	all	4	all
w16 - w19	0.5	0.8	v3	all	4	all
w20 - w22(a)	0.3	0.7	v3	ATT, BER, MBW	1	*Consulten (niet-arts)*
w22(b) - w26	0.3	0.7	v4	all except ADM	3	all
w27 - w34	0.4	0.8	v4	all except ADM	3	all

Figure 3.1: The data settings for each batch, taken from Kim (2021)

After 4 weeks of annotation, the data was analyzed and it was discovered that the frequency of the obtained labels was greatly imbalanced as, with 41%, the ADM category was far too dominant, especially next to the 2-4% of the ATT, BER, and MBW categories. Further, with 49%, the ADM category was vastly over-represented in the COVID-19 data. Therefore, in order to increase the proportion of the ATT, BER, and MBW categories while also decreasing the ADM category in the labeled data, amendments were performed on the parameters. As can be seen in figure 3.1, the parameters were altered until the percentage of COVID-19 notes was 40%, the percentage of notes that contain the keywords was back to 80%, the matched categories were all but ADM, the minimum of matched categories per note were 3, and all note types were included. The keyword lists for categories ATT, BER, and MBW were also updated (hence the keyword version column). This process eventually led to around 6,000 annotated hospital notes.

3.2 Annotated Data

Of those 6,000 annotated notes, about 10% were disregarded, according to Kim (2021), as they contained information not relevant to the project. With around 3%, there were far less disregarded notes in the COVID-19 dataset than in the other ones, which had about 15%. The final 5,554 notes were made up of ca. 286,000 sentences, 5% of which contained at least one category label, which means that – matching the original goal – indeed around 15,000 sentences with category labels were gathered (Kim, 2021).

Figure 3.2 shows how many sentences per category were gathered in the end, which makes it quite clear that the ADM category was still greatly over-represented, while ATT, BER, and MBW were still greatly under-represented. The rest of the categories, however, were more or less balanced.

Kim (2021)'s analysis went even deeper than this, though, as she also reports on the statistics regarding the levels of each category. For the ADM category, she reports that although the overall distribution of ADM levels in all datasets is quite balanced, the COVID-19 dataset sticks out. Within the COVID-19 dataset, the distribution of ADM was quite different, as here there were a lot more instances of levels 0 and 1, while level 4 was not very well represented. For the ATT category, level 2 was consistently well represented in all datasets, for BER it was levels 4 and 0, for ENR 1 and 2, for FAC it was 4, and for STM it was 2. There were almost no instances of MBW level 0 in all datasets, and for INS, once again the COVID-19 dataset differs from the others, where levels 0 and 1 were over-represented and levels 4 and 5 under-represented.

As a final remark, Kim (2021) mentions that in her assessment of random notes versus keyword-selected notes the keyword-search method does not actually produce more

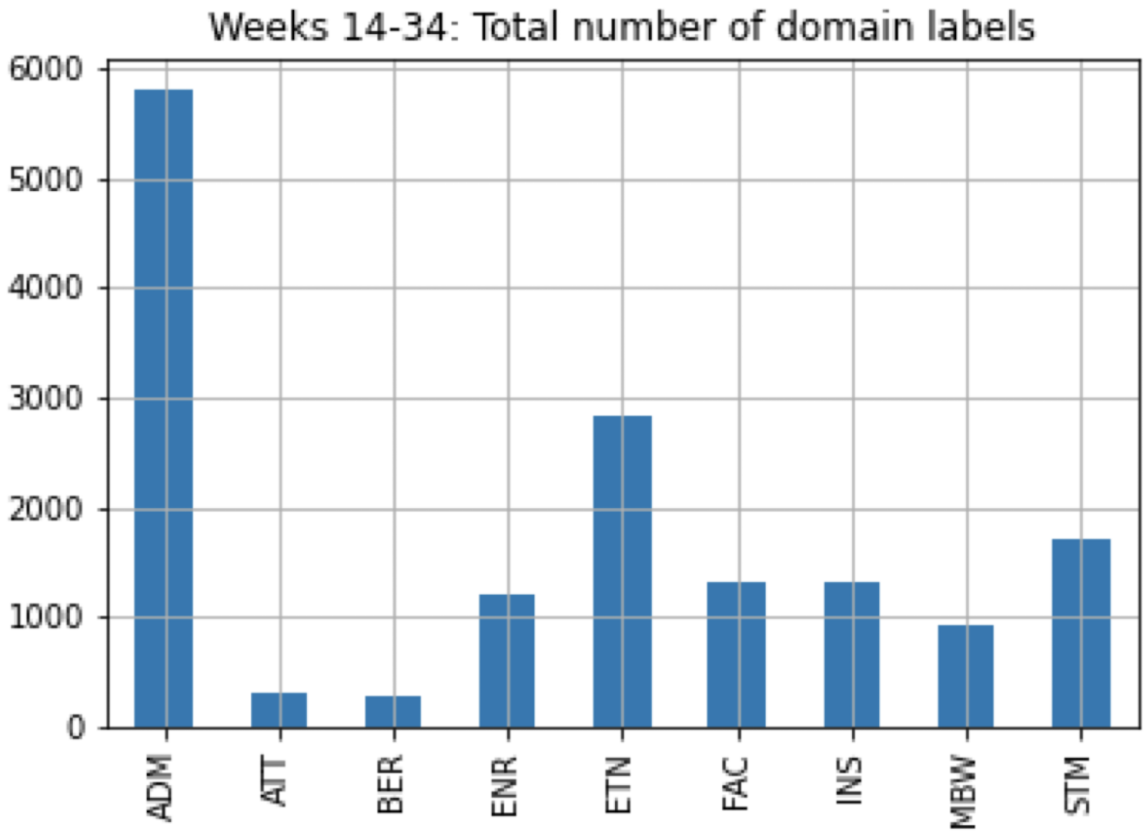

Figure 3.2: Final total number labeled sentences, taken from Kim (2021)

positive examples, as was the motivation behind this idea, though it does "somewhat" help with acquiring more sentences for the three under-represented categories. Here, the distribution of labels for these categories is narrowly higher than in the randomly selected notes.

Chapter 4

Corpus Analysis

In this chapter, a corpus analysis regarding the unlabeled data gathered for this project will be carried out. Since the experiments in this thesis are based on testing the quality and quantity of pseudo-labeled data, it is important to distinguish what was considered "high" and "low" quality data for them. Section 4.1 will be discussing how high and low quality was gathered and determined, while section 4.2 will be examining that data in detail in regards to category quantity and distribution. By having a good understanding of the data the classifier is trained and tested on, a greater understanding of how the used data effects the performance of the model can be gained. As this experiment was started to evaluate exactly how different quantities and qualities of data affect classifier outputs, this detailed examination of the training and testing data is absolutely essential.

Please see figure 4.1 for a visual representation of the process described in this chapter, though note that more detailed explanations of it all will be discussed in the following sections. The process began by running Term Frequency - Inverse Document Frequency (TF-IDF) over all available annotated hospital notes, which returned the 20 most relevant keywords per category. These keywords were then used in a pre-designed keyword matching process to gather around 40,000 unannotated hospital notes from the available corpus that included said keywords in them. These notes, through random selection, were then reduced to 405,000 sentences and run through Kim (2021)'s classifier to receive both a category classification and a confidence score per sentence. Not-O sentences were divided into 3 batches of high quality data, while the O sentences were assigned the category that received the highest confidence score during the classification and then divided into 3 batches of low quality data.

4.1 Data Quality

This section will be discussing how data was gathered from the unlabeled millions of hospital notes we had access to and how these were defined and divided into "high" or "low" quality data.

The first in step in the process of gathering high quality data was using TF-IDF ("Term Frequency - Inverse Document Frequency") on my document corpus, in which the TF-part divides the number of times a term appears in a document by the total number of terms in the document. The IDF-part divides the total number of documents by the number of documents with the term in it and takes the log of that result. The document corpus here were the annotated sentences for which I grouped sentences

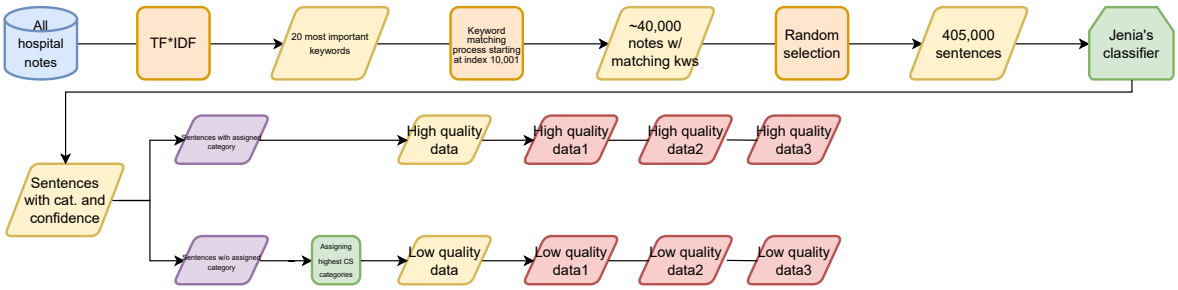

Figure 4.1: The beginning steps of the process

with the same ICF category as one document to measure the spread of terms across categories. Through this process, I could retrieve which words are the most important and prominent ones per category. I needed these to perform a keyword-based search through the unlabeled data to retrieve relevant sentences for the training of my model. In the end, I collected the 20 highest TF-IDF ranked words. All TF-IDF words per category can be found in table A.1 in the appendix. The reason I chose 20 words, which of course meant a lower TF-IDF score each time (i.e. less and less relevant words each time), was because in this step of the process, I wanted to maximize recall.

Since each keyword used in the process returned which *notes* included said keyword, but each note was made up of numerous sentences, it was clear that each of these notes would also include irrelevant sentences. Either irrelevant to the ICF category the note got assigned or irrelevant to my training process, as they were perhaps simply "O" sentences. For example, a note could say "Patient struggles with walking. Loses appetite when tired. Has an older brother." Here, due to the first sentence, the note could be assigned the FAC (walking) category, so the second sentence, regarding the patient's eating habits, is not relevant to the FAC classification. The last sentence does not describe anything about the patient's medical well-being, so it would be assigned the O category. However, as I was going to use the next step in the process to increase my data's precision, it did not matter that much that the data might include a lot of potentially irrelevant sentences. Furthermore, changing the process to collect only the *sentences* that had a matching keyword in them proved to be too computationally challenging and was not feasible under the time constraints of this thesis.

The next step in the process was to use the aforementioned keywords to gather the notes from the unannotated corpus that was available to us, which was done using Postma (2020)'s KeywordMatcher process. This tool uses a list of keywords (in this case the TF-IDF keywords retrieved earlier) to go through a list of files (in this case the unannotated corpus of hospital notes provided to us) and write to a new file which keywords were found in which file. That way an easy overview of which note got assigned to what keyword (and therefore ICF category) was compiled. Due to computational constraints, I gathered only around 40,000 notes in total, with a goal of 10,000 per section of available data (data from 2017, data from 2018, data from 2020, and data from 2020-Q4&2021-Q1), where I made sure to start collecting data *after* the first 10,000 notes, as these were used in the training of the previous students' projects. As a consequence of a character limit of 10,000 characters per note, the goal of 10,000 notes per year was not exactly reached, though it was still quite close: 2017 had 9958

files, 2018 had 9947 files, 2020 had 9933 files, and 20-21 had 9921 files. It was these final 39,759 files that the keyword matching process was then run over. However, since the code was not designed to match each note with only one category, there were a few duplicates, which is why the final note-per-category-per-year amount below does not add up to the files per year described above: 2017 had 403 duplicates, 2018 had 465, 2020 had 1448, and 20-21 had 2218.

Details on the note distribution after running the keyword search over them and collecting which of them belonged in which ICF category, based on their keyword match, are given in table 4.1.

	2017	2018	2020	20-21	**Total**
ADM	1313	1225	1326	1467	5331
ATT	1315	1295	1471	1555	5636
BER	514	524	529	578	2145
ENR	1039	1010	1129	1180	4358
ETN	1425	1420	1557	1591	5993
FAC	1967	1975	2270	2349	8561
INS	926	1033	1027	1158	4144
MBW	1132	1103	1236	1391	4862
STM	730	827	836	870	3263
Total	10,361	10,412	11,381	12,139	**44,293**

Table 4.1: ICF category note distribution per year using Postma (2020)'s Keyword-Matcher

Out of these 44,293 notes, 10,066 were assigned only a single category, 11,577 were assigned multiple categories, and the other 22,650 notes were the duplicates of the multiple-category notes. After separating these notes into sentences and assigning each sentence the same category as its note, there were 2,680,977 sentences in total available to me. Out of these, the category distribution was as follows:

- 458,830 FAC sentences, 17%
- 352,302 ATT sentences, 13%
- 349,546 MBW sentences, 13%
- 343,825 ADM sentences, 13%
- 343,081 ETN sentences, 13%
- 258,436 ENR sentences, 10%
- 241,756 INS sentences, 9%
- 180,231 STM sentences, 7%
- 152,970 BER sentences, 6%

Since over 2 million sentences would have taken far too long to get predictions on or fine-tune a classifier with, I reduced the number of total sentences to 405,000, as this was a number that has been proven to be high enough for good results, but low

enough to not be too computationally complex. These 405,000 sentences were chosen at random and equally split per category, i.e. 45,000 sentences per category. I furthermore ensured to only add sentences that were above 100 characters long, as I wanted to make certain I had long enough sentences that would provide actual learning examples for the model. However, this resulted in only roughly about 3000 instances for both high and low quality data, so I repeated this step and this time decreased the character length to more than 50, rather than 100. This resulted in the desired 45,000 sentences per category and 405,000 sentences in total.

Although these sentences were already divided into ICF categories through the keyword matching process, the point of this experiment was to ensure semi-supervised learning. i.e. training the classifier with data labeled by itself. That is why, once the keyword matching process was done, I ran these sentences through Kim (2021)'s yet unchanged classifier to see what categories it would assign them, though I added a confidence score to the results as well. I needed every sentence's score for the rest of my experiments. A confidence score is a score between 0 and 1 that states, per class, how "sure" the model is that the input instance belongs to this class, with 0 being sure that it is *not* that class and 1 being sure that it is. For comparison, after running the 405,000 sentences through the classifier, please see table A.2 in the appendix for the distribution of sentences per category/categories.

The most noticeable difference is the fact that Kim (2021)'s classifier was able to assign two or sometimes even three categories per sentence. Due to the attempt to avoid duplicate sentences, Postma (2020)'s KeywordMatcher could certainly not do that. Further, where Postma (2020)'s KeywordMatcher actually returned the most FAC sentences, Kim (2021)'s classifier classified most sentences as STM, which was one of the lowest ranking categories in the keyword matching process. Something similar is happening with the ATT category, where Postma (2020)'s KeywordMatcher ranked it as one of the second highest categories, while Kim (2021)'s classifier classified the least amount of sentences as ATT. The FAC category seems to have been assigned similarly often by both processes, while the BER category was similarly rarely assigned in both cases. The O category, as is the case with most classifiers, was naturally the highest ranking category assigned by Kim (2021)'s classifier. In fact, it was assigned so often that it would be the second highest ranking category in the keyword matching process results. It can be deduced from this comparison that a) simple keywords are not enough for a classifier to classify an input instance as a certain class – as then both processes would surely have had at least proportionally similar results – and b) human picked keywords, i.e. words we as humans would perhaps associate with certain topics, are not always actually representative of the subject they might stem from. The context and surrounding words seem to say a lot more about a sentence's topic than specific keywords within it.

4.2 Data Quantity

This section will be examining the previously gathered data's quantity and distribution regarding categories and confidence scores.

It is at this step in the process that I started dividing the data into "high" and "low" quality data. As mentioned above, the classifier's confidence score was vital to my experiments because I consider data that the model is quite sure about as high quality data. The reason for that is that I believe this means the data is written and

presented in a way that is clear and clean for the model to make an easy prediction, with little to no noise in it. The highest confidence score the classifier displayed for all classifications was 0.246, while the lowest one was 0.077, whereas the average confidence score was 0.11. This clearly shows that the scores overall tended to be in the lower half between the maximum and minimum score. The distribution of the maximum confidence scores per sentence were as follows:

- 0.11-0.12: 337,831
- 0.12-0.13: 11,748
- 0.13-0.14: 7808
- 0.14-0.15: 5956
- 0.15-0.16: 5044
- 0.16-0.17: 4681
- 0.17-0.18: 4619
- 0.18-0.19: 4445
- 0.19-0.20: 4136
- 0.20-0.21: 4455
- 0.21-0.22: 4534
- 0.22-0.23: 5027
- 0.23-0.24: 3846
- 0.24-0.25: 870

The enormous distribution in the 0.11-0.12 range represents the large distribution of O classifications, as a low confidence score for a class causes the classifier to not classify that instance as that class. When this happens with each class, the instance will be classified as 'none' and therefore O.

The distribution of maximum confidence scores in cases where the classifier assigned a class were as follows:

- 0.15-0.16: 28
- 0.16-0.17: 434
- 0.17-0.18: 4428
- 0.18-0.19: 4445
- 0.19-0.20: 4136
- 0.20-0.21: 4455
- 0.21-0.22: 4534

- 0.22-0.23: 5027
- 0.23-0.24: 3846
- 0.24-0.25: 870

It becomes quite clear to see here that the threshold for the classifier to change from 'none' to 'not-none' was somewhere above 0.15, though not always and/or exclusively, as can be seen in the next bullet list.

The distribution of maximum confidence scores in cases where the classifier did *not* assign a class were as follows:

- 0.11-0.12: 337,831
- 0.12-0.13: 11,748
- 0.13-0.14: 7808
- 0.14-0.15: 5956
- 0.15-0.16: 5016
- 0.16-0.17: 4247
- 0.17-0.18: 191

The fact that the scores from when the classifier did and did not assign a category overlapped in the 0.15-0.18 range shows that it was not the confidence score alone that caused the classifier to assign a class. For example, the Simple Transformers MultiLabelClassificationModel also has an argument called "threshold," which, according to their website, means "The threshold is the value at which a given label flips from 0 to 1 when predicting" (Rajapakse, 2020). The base classifier's threshold argument was set to 0.5, while my confidence score was calculated using the *SciPy* library's softmax function. This means that there is a chance that the classifier's internal threshold of 0.5 was not passed by the instance, although the *SciPy* confidence score was above 0.15. That would explain why some instances in the 0.11-0.16 range *were* classified and others in that range were *not*.

Initially, the plan was to work only with the data the classifier had indeed classified as not-none, however, as can be gathered from the second bullet list, that was not a lot of data overall. Especially considering the 405,000 sentences the data had come from. Therefore, to increase the amount of data and be able to properly train the classifier later, I decided to label *all* data classified as not-none as "high" quality data. For the "low" quality data, I decided to choose only the sentences that had been marked O, but still had one or more confidence scores between 0.12 and 0.15. The reason for this was that anything below 0.12 was the majority of the O class, which I deemed to be quite noisy or short sentences that would not actually be efficient and clear training data for the classifier (which is why I chose to ignore that data for the purpose of this thesis). Anything above 0.15 would be the opposite of that, which I did not want in my "low" quality data. In summary, for my "low" quality data I was looking for data that would be helpful for the training, but not so good that it would be better than the "high" quality data. Each sentence was then assigned the category of its highest matching confidence score, so if a sentence had confidence scores of [0.112, 0.113, 0.11,

0.13, 0.114, 0.07, 0.115, 0.111, 0.116] while still being classified as O, its new assigned category would be the fourth one (0.13), which would be ENR.

In the end, after dividing the data into these two sub-sets, I had 32,203 HQ data instances and 25,882 LQ data instances, totaling to 58,085 combined instances. These two sub-sets were then equally divided into three further sub-sets; simply their first, second, and last third (aptly named hq_data1, hq_data2, hq_data3, and lq_data1, lq_data2, lq_data3 for future reference). It was these thirds that I would eventually use to train the different models step-by-step.

For the full distribution of categories among the HQ data, please see table A.2 in the appendix without the O category. For the distribution among the first and second third of the HQ data, please see table A.4 and table A.5 in the appendix. For the full category distribution among the LQ data, please see table A.3 in the appendix; for the distribution among the first and second third of the LQ data, please see table A.6 and table A.7 in the appendix.

Chapter 5

Experiments

In this chapter, a detailed description of the experiments conducted throughout this thesis will be provided, as well as answers to the research questions "How much does the quality of the training data influence the model's performance?" and "How much does the quantity of the training data influence the model's performance?" As has been the focus already so far, the idea was to test whether a classifier performs better with high quality data or high quantity data, which was investigated through semi-supervised learning. By training a classifier with different corpora that range in quality and quantity, it will be easy to see what combination of both produces the highest performance – high quality and high quantity, high quality and low quantity, low quality and high quantity, or low quality and low quantity. My hypothesis is that this order will also be the ranking of highest to lowest performance on the final test set. As an introductory note to this chapter, it should be mentioned that while the semi-supervised learning methods described in chapter 2 were automated processes, due to time and computational restrictions during this thesis, my semi-supervised learning method was done manually. This means that instead of providing a model with all the data and having it run the whole training process on it automatically, I repeated the training and predicting of the models manually.

The chapter will be divided into four sections, where section 5.1 will be discussing the baseline of my experiments, section 5.2 will be explaining how the quality experiment of this thesis was conducted, and section 5.3 will be explaining how the quantity experiment of this thesis was conducted in turn. Section 5.4 will be explaining how the best model was chosen and present its performance and results on the test set.

5.1 Baseline

Before preparing experiments, it is beneficial to have a baseline performance against which to compare one's own experiments' results. In this case, the most useful approach for this would be testing how the base classifier used in all my experiments would perform on the development and test sets without any fine-tuning, so that I could directly observe whether increasing the data's quality or quantity improved that classifier's performance. It should be mentioned here, however, that Kim (2021)'s original machine learning pipeline did consist of two parts, both of which together eventually constructed her complete process, while I only made use of the first part of pipeline. Where this first part classified a sentence's ICF category (as mentioned throughout this paper), the second part of her pipeline classified said sentence's category *level*. I did not, however,

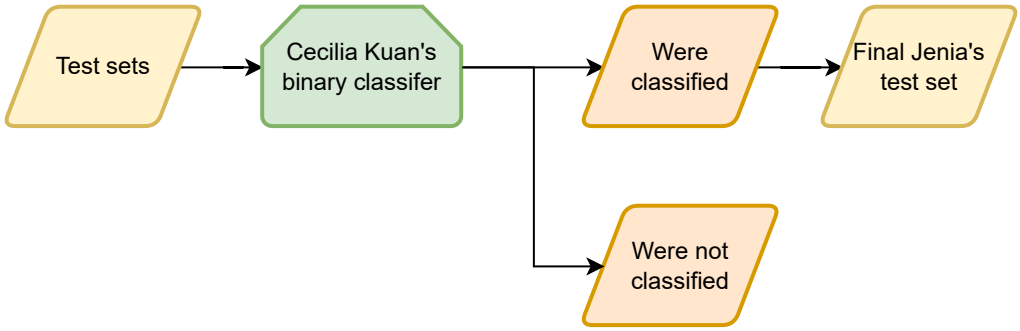

Figure 5.1: Creation of the development and test sets

focus on a sentence's level within the scope of this thesis, so that aspect of Kim (2021)'s classifier remained untouched by me.

5.1.1 Test Sets

Regarding the test and development sets for both the baseline and the rest of my experiments, I must point to Kim (2021) and Galjaard (2022) and Badloe (2020) once again, as I used their test sets and data for my test and development sets. However, as was discovered by Kuan (2023) during her thesis work, any fine-tuned version of Kim (2021)'s model struggled with identifying positive examples in the input instances *at all*. It is for that reason that Kuan (2023) first built a binary classifier trained on Kim (2021)'s training data to determine which input instances would be classified at all, simply denoted by either 0 or 1. Therefore, for my own test and development sets, I used this classifier to first run my desired datasets through there and determine which of the sentences within them would be useful for my experiments at all. The ones assigned a 0, i.e. would not be classified by Kim (2021)'s model, were not used in my experiments, as they were not deemed useful for the experiments. For a distribution regarding ICF categories within both the test and development set, please see table A.8 and table A.9 in the appendix. For a visual representation of this process, please see figure 5.1. The development set was Kim (2021)'s original test set run through the binary classifier, while the test set was made up of all of Galjaard (2022) and Badloe (2020)'s datasets combined and then run through the binary classifier. In the end, the development set had 1148 instances in it and the test set 1965. The development set was used for evaluation after each fine-tuning of a model, while the test set was only used for the final best model.

Regarding the distribution among the test sets, it is clear that in both the development and the test set, the ATT and BER categories are highly underrepresented, while both sets are also highly imbalanced. This will be further discussed in chapter 6.

5.1.2 Results

Results from the base classifier on the development set can be seen in figure 5.2 and results by it on the test set can be seen in figure 5.3.

Comparing these two results, it becomes clear that, except in the O category, the recall scores for both sets are identical. What this means is that for both sets, the

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.838843	0.673228	1.000000	0.428571	0.714286	0.538462	0.479769	0.381818	0.557143	0.699248
Recall	0.366426	0.939560	0.777778	0.857143	0.925926	0.833333	0.976471	0.583333	0.975000	0.958763
F1 score	0.510050	0.784404	0.875000	0.571429	0.806452	0.654206	0.643411	0.461538	0.709091	0.808696

Figure 5.2: Baseline performance on the development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.917373	0.513514	0.875000	0.193548	0.403226	0.231788	0.365639	0.194444	0.221591	0.505435
Recall	0.315828	0.939560	0.777778	0.857143	0.925926	0.833333	0.976471	0.583333	0.975000	0.958763
F1 score	0.469886	0.664078	0.823529	0.315789	0.561798	0.362694	0.532051	0.291667	0.361111	0.661922

Figure 5.3: Baseline performance on the test set, *support - O: 470, ADM: 319, ATT: 11, BER: 16, ENR: 113, ETN: 265, FAC: 171, INS: 96, MBW: 145, STM: 134*

classifier found the same proportion of true positives out of all positives it should have found. Given that all recall scores except for two (O and INS) are at least above 0.7, the classifier did indeed manage to find a large amount of that proportion within the two sets. Such high recall scores indicate that the classifier may have found most instances of each class, though may have also labeled a lot of other ones with the wrong classes, i.e. there will be more false positives in this dataset. INS was overall the worst performing category, though, with an F1-score of 0.46, which could be explained by the annotation changes that occurred during Kim (2021)'s project, as mentioned in chapter 2.

Regarding the development set (figure 5.2), the precision scores range from poor (BER, FAC, and INS), to mediocre (ETN and MBW), all the way to fairly good (O, ATT, ENR, STM), with ATT having a precision score of 1, meaning the classifier may not have found all ATT instances, but whatever it did classify as ATT was always correct. A quick look at figure A.2 in the appendix reveals that one ATT instance was classified as O, while the other was labeled FAC. Since the ATT category stands for "attention," while the FAC category represents sentences related to "walking," it does not seem intuitive for a human brain as to how these two got confused for one another, though perhaps attention is needed to perform walking.

The precision scores in the test set (figure 5.3) were not as high as in the development set. Here, they range from extremely poor (BER, ENR, ETN, FAC, INS, and MBW) to mediocre (ADM and STM), with only the O category having a very good precision score above 0.9. The low precision scores in the six categories mean that these categories were mostly mislabeled, leading to more false positives. Given that the development set's precision scores were so much higher, it is no surprise that its F1-scores are much higher as well. Though these scores are by no means poor, ranging predominantly from 0.5 to 0.8, they could still certainly be improved, especially in the BER and INS categories.

Overall, the base classifier performs somewhat unremarkably on both sets, though definitely worse on the test set. The reason for this is most likely the different kinds of data represented in the test and development set. The development set, where the baseline performed better on, was largely made up of the same kind of data the classifier was trained on, i.e. a lot of COVID-19 data. The test set, on the other hand, included far more non-COVID-19 data, taken from Galjaard (2022) and Badloe (2020)'s data.

This shows and explains the differences in the two evaluations. However, one should not forget that the training data of the baseline classifier included hundreds to thousands of instances per categories, while the development and test sets included far less, especially in the ATT and BER category. What this means is that, since the support is low in the test sets, the result are rather inconclusive and cannot be fully trusted, especially in the low represented classes such as ATT and BER. Further testing and comparing of other fine-tuned models' performances should provide more clarity.

5.2 Quality – High Quality vs. Low Quality

As can be seen in figure 5.4, the same process was done and repeated on both the high quality and the low quality dataset that I curated in chapter 4. This was done so that a direct comparison would be easy to draw from the results, as the procedures and the base classifier were the exact same each time.

5.2.1 *ModelHQ1* and *ModelLQ1*

The process began by taking Kim (2021)'s classifier and fine-tuning it with the first third of the HQ and LQ datasets, which resulted in the models *ModelHQ1* and *ModelLQ1*. These first two models were then evaluated on the development set (Kim (2021)'s original test set with only the positive examples from Kuan (2023)'s binary classifier), which lead to the results seen in figure 5.5 and figure 5.6.

It should be noted here that these confusion matrices only focus on the singular ICF categories, since that was the scope of this thesis within its time and computational restraints. Further, except for a few outliers, the singular categories were the most represented ones both in the training and the test sets, which is why I believe it is far more relevant to focus on these in regards to the evaluation.

Now, even with quick glances at these results, it is clear that the F1 scores, i.e. the harmonic mean of precision and recall, of the *ModelHQ1* model are predominantly higher than the the F1-scores of the *ModelLQ1* model. However, why that is not the only metric to focus on and which one I used to decide on my best model will be discussed in section 5.4.

Comparing these two results to the baseline results on the development set (figure 5.2), one can quickly see that the results have not improved that much. In fact, most of the precision and F1 scores have decreased. Even the poorest precision scores from the baseline experiment (BER and INS) got worse for the HQ model, although the INS recall and F1 scores increased slightly for the LQ model. This could be the result of the higher INS distribution among the LQ training set, as this would have trained the classifier better to correctly recognize INS sentences. All recall scores (except in the O category) for the HQ model did, however, increase from the baseline performance, while for the LQ model, only the BER, ETN, INS, and STM recall scores improved, the rest got worse. It seems the HQ model's lower F1 scores stem from the lower precision scores then, rather than the recall scores, as it does quite extraordinarily in that metric.

Overall, just as with the baseline, the recall scores tend to be higher than the precision scores for both models, meaning they also classify more false positives. Between the two models, the HQ one majoritively has higher F1 scores overall, except for the O, BER, FAC, and MBW categories. With such a high amount of FAC instances in its training data, it is not surprising that the HQ model's FAC precision score is that

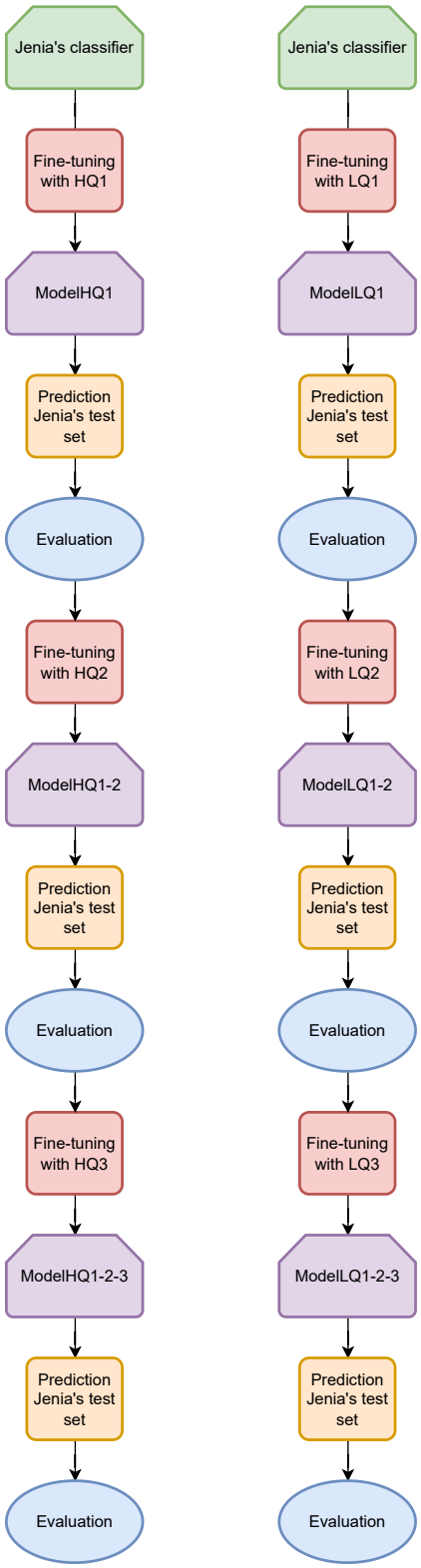

Figure 5.4: High quality vs. low quality data experiments

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.791667	0.625455	0.888889	0.375000	0.621951	0.419689	0.397129	0.352113	0.513158	0.648276
Recall	0.102888	0.945055	0.888889	0.857143	0.944444	0.964286	0.976471	0.694444	0.975000	0.969072
F1 score	0.182109	0.752735	0.888889	0.521739	0.750000	0.584838	0.564626	0.467290	0.672414	0.776860

Figure 5.5: ModelHQ1 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.657895	0.638132	0.800000	0.538462	0.672727	0.416667	0.426230	0.348485	0.565217	0.635762
Recall	0.180505	0.901099	0.888889	1.000000	0.685185	0.952381	0.917647	0.638889	0.975000	0.989691
F1 score	0.283286	0.747153	0.842105	0.700000	0.678899	0.579710	0.582090	0.450980	0.715596	0.774194

Figure 5.6: ModelLQ1 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

poor (0.397), as the model clearly overfitted to this category during training. Its low precision score means that the model often falsely classified instances as that class. Two interesting examples for an FAC false positive I would like to point out can be seen in table 5.1.

Original sentence	*Translation*	Gold	Predicted
Gesport omgekleed naar de uitgang gelopen	*I got dressed and walked to the exit*	-	FAC
Gewoon lopen gaat goed, maar langzaam	*Just walking is fine, but slow*	-	FAC

Table 5.1: Interesting false positive FAC examples from *ModelHQ1*

I wished to point out these two examples because although the human annotators labeled these two sentences as having no category at all, looking at them myself, I felt as though they do indeed hold valuable information about a patient's walking abilities and should therefore be labeled as FAC. This is not meant as criticism of the annotators, but rather to point out that sometimes models will perform poorly on paper, if one only looks at confusion matrices, while in reality a model may have labeled something with a class a human would agree with, but the gold labels simply did not match. This should always be kept in mind when evaluating any model's performance scores. For another example of my point, please see table A.10 in the appendix.

It is true that both models' highest precision and F1 scores are in the ATT category, a category that has only between 100 and 300 instances per training set. This could suggest that this amount of training instances is a more ideal amount for good training without overfitting. Especially since the precision and recall score for the ATT category in the HQ model are identical and still very good for the LQ model. However, this could also be a comment on the different qualities of the training data of the models. The LQ model had 219 training instances, while the HQ model had 189, similar enough amounts in machine learning to not warrant such differences, especially compared to the baseline, yet the HQ model consistently had higher F1 scores than the LQ model, i.e. more balanced scores.

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.743243	0.643382	0.888889	0.352941	0.571429	0.415385	0.415385	0.320513	0.520000	0.630872
Recall	0.099278	0.961538	0.888889	0.857143	0.888889	0.964286	0.952941	0.694444	0.975000	0.969072
F1 score	0.175159	0.770925	0.888889	0.500000	0.695652	0.580645	0.578571	0.438596	0.678261	0.764228

Figure 5.7: ModelHQ2 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.703125	0.631783	0.727273	0.466667	0.686567	0.417526	0.403061	0.370968	0.542857	0.653061
Recall	0.162455	0.895604	0.888889	1.000000	0.851852	0.964286	0.929412	0.638889	0.950000	0.989691
F1 score	0.263930	0.740909	0.800000	0.636364	0.760331	0.582734	0.562278	0.469388	0.690909	0.786885

Figure 5.8: ModelLQ2 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

5.2.2 *ModelHQ2* and *ModelLQ2*

The next step was to fine-tune *ModelHQ1* and *ModelLQ1* with the second third of the HQ and LQ training data respectively. This process was identical to the one mentioned in subsection 5.2.1, only with a different base classifier this time. The results for both models can be seen in figure 5.7 and figure 5.8.

In this section, I will only be comparing each newly fine-tuned model pair to each and other and to the baseline. A comparison between all fine-tuned models will be discussed in section 5.4.

A look at all F1 scores between *ModelHQ2*, *ModelLQ2*, and the baseline reveals that the baseline once again has the highest scores there, except in categories ATT, BER, and INS. *ModelHQ2* has the highest ATT F1 score of all three models, while *ModelLQ2* dominates the BER and INS categories. *ModelLQ2*'s high precision scores in these categories seems to be the reason for that in the INS category, while all of its scores are higher than any of the other two's scores in the BER category. However, the baseline's higher F1 scores seem to stem from its overall higher precision scores in almost all categories.

All three models have two categories where they out-perform the other models in the recall score, while four times there are ties between two models for the recall score. Except for the O category, all three models always have a higher recall than precision score, which means that, just as before, the models predict more false positives for each category. With such balanced scores throughout each category and metric, neither one of the two fine-tuned models always or predominantly performs better than the other, which seems to be suggesting that the quality of the data did not have much of an impact on the models' performance.

The highest performing category for the precision and F1 score of each model is ATT, which is surprising as it is one of the least represented categories in fine-tuned models' training data. The highest performing category for the recall score is FAC for the baseline, STM for the HQ model, and BER for the LQ model. STM is a highly represented category in the HQ model's training data, so its high performance makes sense, but BER is a rather poorly represented class in the training data for the LQ model, though it seemed enough to have taught the model what a correct BER sentence looks like. Regarding the precision score of the ATT category, this means that

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.756098	0.637037	0.727273	0.352941	0.644737	0.423280	0.411168	0.313953	0.527027	0.643836
Recall	0.111913	0.945055	0.888889	0.857143	0.907407	0.952381	0.952941	0.750000	0.975000	0.969072
F1 score	0.194969	0.761062	0.800000	0.500000	0.753846	0.586081	0.574468	0.442623	0.684211	0.773663

Figure 5.9: ModelHQ3 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

for this class, all models predicted more false negatives, meaning whenever an instance was indeed of the ATT category, the models labeled that correctly, though they missed a large portion of all available ATT instances. An example of such a false negative from the HQ model can be seen in table 5.2.

Original sentence	*Translation*	Gold	Predicted
Bij onderzoek verhoogd afleidbaar, afasie met gestoorde fluency en woordvindstoonris, geen begripsstoornis, en instabiel lopen, geen andere focale neurologische uitval	*During research increased distractible aphasia with impaired fluency and [wordfindingtoonris], no comprehension disorder, and unstable walking, no other focal neurological failure*	ATT	FAC

Table 5.2: False negative ATT example from *ModelHQ2*

While the human annotators labeled this sentence as ATT (presumably because of the "increased distractible aphasia" and perhaps the "focal neurological failure"), the model classified it as FAC, the walking category. My hypothesis here is that the first half of the sentence involved a lot of content the model had perhaps not encountered (enough) before, or not with enough clear cut class labels, so it was unsure as what to classify that. However, the phrase *en instabiel lopen* ("unstable walking") included the word *lopen* ("walking"), which has been a fairly common word in the FAC category training instances. This finding suggests that the model not only sometimes still relies heavily on singular keywords for classification, it also struggles with unfamiliar words or context and tends to ignore those.

In summary, given the high recall and low precision scores of both fine-tuned models, it seems the models have learned to recognize certain keywords or phrases as belonging to a specific class (high recall), though they failed to learn that not every time these words are mentioned the instance does indeed talk about a patient's functioning within that category (low precision). Training with more and higher quality data might eradicate these problems, especially since the quality of the training data between these two models did not make a big difference in their level of performance.

5.2.3 *ModelHQ3* and *ModelLQ3*

The third step in the process was once again fine-tuning the two already fine-tuned models with the last third of the HQ and LQ datasets. This process was identical to the one mentioned in subsection 5.2.2, only with a different base classifier this time. The results for both models can be seen in figure 5.9 and figure 5.10.

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.637306	0.631373	0.888889	0.500000	0.666667	0.415385	0.411765	0.411765	0.569231	0.643836
Recall	0.222022	0.884615	0.888889	0.857143	0.851852	0.964286	0.741176	0.583333	0.925000	0.969072
F1 score	0.329317	0.736842	0.888889	0.631579	0.747967	0.580645	0.529412	0.482759	0.704762	0.773663

Figure 5.10: ModelLQ3 results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

At first glance, one can see immediately that again the recall scores of both models are much higher than the precision scores, which still indicates more false positives within the model performance. Though the recall scores of both models are indeed quite formidable, ranging mostly between 0.7 and 1.0, the precision scores are still quite poor, ranging predominantly between 0.3 and 0.6. Just like during the last two steps, high recall and low precision means the classifier returns *most* of the correct results, though not as many as there are. In other words, were one to use these models on unfamiliar data without gold labels, though there would be a large amount of class A instances returned, one could not always be entirely sure that all of them are indeed class A.

Compared to the baseline, *ModelHQ3* does have either higher or very close recall scores (higher scores in the ADM, ATT, ETN, INS, and STM categories), while *ModelLQ3* only has a significantly lower recall score (-0.23) in the FAC category. The other scores are also either higher or quite close. In regards to the precision scores, *ModelLQ3* actually has a higher score than the baseline in the BER, INS, and MBW categories. This would indicate an increase in performance compared to the baseline for this model, as its recall scores are now close or higher than the baseline, and its precision scores higher than it 3 times, which was not the case in the previous step. *ModelHQ3* only seemed to have improved in its recall scores, all of its precision scores are still lower than the baseline.

ModelLQ3's precision scores are higher than *ModelHQ3*'s in 6 categories (ATT, BER, ENR, FAC, INS, and MBW), while *ModelHQ3*'s recall scores are higher than its counterpart's in 5 categories (ADM, ENR, FAC, INS, and MBW). They both tie in ATT and STM. Although *ModelLQ3*'s precision scores are not extraordinary, having higher precision scores but lower recall scores than its counterpart means that this model has more false negatives and less false positives than the HQ model. However, compared to themselves, the models still both have more false positives than false negatives in their predictions, as mentioned above.

Despite its rather impressive recall scores (6 of them higher than 0.9), *ModelHQ3* still displays two precision scores between 0.31 and 0.36 (INS and BER respectively). Neither of the other two models have any scores that low, the lowest score among them being 0.411 in the FAC and INS categories in the LQ model. *ModelHQ3* also more often than not has the lowest F1 score out of the three models, thanks to its low precision scores.

All in all, although the baseline still curated the highest F1 scores overall, *ModelHQ3* shows quite impressive recall scores, while *ModelLQ3* impresses with precision scores that majoritively range higher than its HQ counterpart.

Looking at the improvement of all scores in the LQ model, it goes against my initial hypothesis to see such high scores on a model trained with what I deemed to be "low" quality data. Given that this data was procured from originally O-labeled

sentences that were simply assigned their highest confidence score class, it is interesting to see just how beneficial this data was for training after all. Not only are these scores quite impressive on their own, were this to be the final model of an experiment, they are also rather exemplary compared to the baseline, especially in the ATT and INS categories. The ATT class had only a little less than 600 instances in the training data, while INS had around 3500 in total. This raises an interesting point regarding data quantity, as both categories did quite well, despite their very different training quantities. Perhaps within these two categories, regardless of quantity and despite the fact that my process labeled these sentences "low" quality, the training data was still so diverse and clean that the classifier learned quite well how to distinguish these classes. This makes me believe that it was indeed the training data that caused such high performance scores in *ModelLQ3*. However, given that the ATT category only had 9 instances and INS 36 instances in the development set, this imbalance between training and test set makes the results rather inconclusive for these minority classes, as touched upon in subsection 5.1.2.

Regarding *ModelHQ3*; were this to be my final model on another experiment, I believe I would continue training and/or tweaking the training data, with final results such as these, especially in the BER and INS category. As much as one always has to make a choice between precision and recall, I believe precision scores of less than 0.4 in more than 1 category would not let me present this model as a robust one. BER and INS had between 1300 and 2300 instances (i.e. ∼430 and ∼770 instances per batch) in the HQ training data, which is well between the numbers for the LQ model, yet the HQ model performed so much worse in these categories, especially in the precision metric. This leads me to believe that perhaps the training data, despite my "high" quality labeling, was not as clean and/or diverse as it could have been and therefore did not increase the classifier's ability to distinguish these sentences well. This hypothesis is supported by the fact that low precision scores in these categories means more false positives in the model's predictions, as mentioned before, which would suggest overfitting, though that seems unlikely, given that the LQ model had much more INS instances in its training data and still performed less poorly on it.

5.3 Quantity – All Data

This section will be discussing the experiment conducted to answer my sub-research question of "How much does the quantity of the training data influence the model's performance?" After training two different models with different kinds of quality of data, I then trained one of the models, *ModelHQ3* to be precise, with the rest of the LQ data, in order to evaluate a model's performance after a high *quantity* of data. Please see figure 5.11 for a visual representation of my process.

As can be seen in the figure, the process was basically the same as in subsection 5.2.1 and the following two sections, only with a different base classifier yet again. The only difference this time, however, is that I did not evaluate each intermediate model. The reason for that was that I was indeed trying to focus only on data quantity, so it was more important to evaluate the final model that was trained with all the available training data, rather than a model that was trained on a section of the data. I did save each intermediate model, though, as they could still be useful and provide important and/or interesting findings in the future.

Please see figure 5.12 for the results of this model on the development set. It is

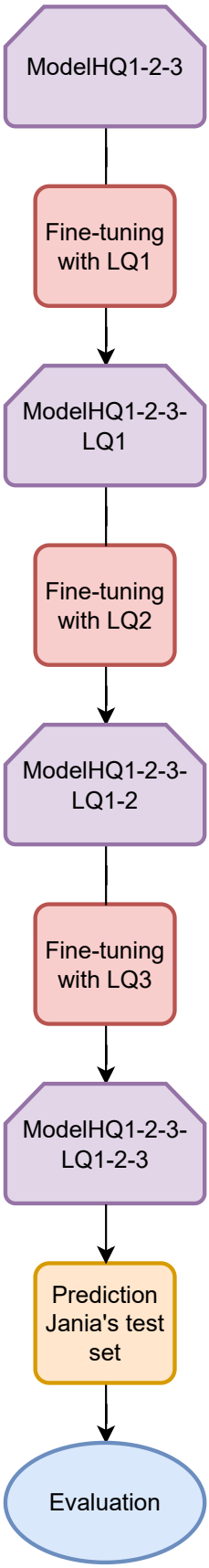

Figure 5.11: Training *ModelHQ3* (here labeled *ModelHQ1-2-3*) with the LQ data

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.615000	0.634921	1.000000	0.428571	0.677966	0.416667	0.393939	0.392157	0.571429	0.645833
Recall	0.222022	0.879121	0.888889	0.857143	0.740741	0.952381	0.764706	0.555556	0.900000	0.958763
F1 score	0.326260	0.737327	0.941176	0.571429	0.707965	0.579710	0.520000	0.459770	0.699029	0.771784

Figure 5.12: *ModelHQ1-2-3LQ1-2-3*'s results on development set, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

clear here that the model's overall performance was acceptable, with definite room for improvement, but no subpar performance either. Especially the recall scores are, as with all models evaluated so far, quite high, ranging mostly between 0.7 and 0.96, with only two outliers at 0.55 (INS) and 0.22 (O). The precision scores are lower, with four scores around 0.4 (BER, ETN, FAC, and INS), and most scores ranging between 0.57 and 0.67. The only exception is once again the ATT category, with a precision score of 1. With such precision and recall scores, it is no surprise that *ModelLQHQ3*'s F1 scores range predominantly between 0.52 and 0.77, with only one score significantly higher than that (0.94, ATT) and one score significantly lower than that (0.32, O).

Comparing the model to the baseline, even if they are majoritively lower, the model's precision scores are quite often very close to the baseline's, with only ETN and FAC having differences of around 0.1 points. INS and MBW are the only two categories where *ModelLQHQ3*'s precision scores are higher than the baseline's. Interestingly, both models' scores in the BER category are identical for every metric. Regarding the recall scores, *ModelLQHQ3* only surpasses the baseline's scores in the ATT and ETN categories, though here by more than 0.1 points. Overall, however, the model's recall scores are quite a lot lower than the baseline's.

ModelLQHQ3 only surpasses the baseline's F1 score in the ATT category, though most of the other scores of this metric are quite similar to those of the baseline, with only two categories (ENR and FAC) having difference of 0.1 or higher. Overall, the model performs worse than the baseline, though not by much.

The high ATT precision score shows that out of the 9 instances in the development set, the model found a few of them and despite labeling all of those correctly, it did miss the rest of the ATT sentences. However, its high recall score (and a look at its full confusion matrix, not pictured here) also shows that the model only missed 1 ATT instance, which it falsely classified as FAC. This sentence happened to be the same one as in table 5.2, which suggests that this model also relied quite heavily on the word *lopen* and did not know what to do with the unfamiliar words/context in the sentence. Regarding other false positives in the FAC category (as it had such a low precision, together with INS), there seemed to be a pattern of two categories these sentences would fall into: Either an example as mentioned in table 5.1, where a case for an FAC label could be made, or sentences such as *Onderzoek bij vader van meneer loopt* ("Investigation of sir's father is ongoing"), that are not related to a patient's walking ability, but involve a version of the word *lopen*.

There is a similar pattern for INS false positives, where the model only ever falsely classified O-instances as INS, all of which included words such as "active," "sport," or "bicycle." This shows clear evidence that the classifier has learned to associate certain words with certain categories. Though I would like to once again make the case that the model was not always wrong in these assumptions, as some of the instances were

phrases like "Sport: 3x a week," which I believe does indeed relay information about a patient's exercise functioning.

Diving deeper into the false positives of BER and ETN, the two other categories with low precision scores, evidence for similar patterns as described above become clear as well, especially with regards to a reliance on certain keywords. Sentences for which a case for these categories could be made are rarer, however.

Looking at these results, of a model trained on 58,085 total instances, 32,203 of which were "high" quality instances, and 25,882 of which were "low" quality instances, the results are quite representative of such training data. Given that Kim (2021)'s classifier was trained on around 200,000 instances, it is no surprise that this one (and the previous models) have not performed as well as the baseline. However, as mentioned above, the results of this model were not inherently unacceptable, even if improvements could still be made.

5.4 Best Model

This section will present a comparison of *all* models created so far against one another, instead of just compared to the baseline. This is done so that I can best describe how and why I chose the final best model, which I eventually evaluated on the test set. It is important to mention here that I decided to find the model that I deem to have the highest *recall* performance, rather than trying to focus on precision or F1. The reason for this is that is that a high recall score, as mentioned above, returns more false positives, while a high precision scores returns more false negatives. Considering what the A-PROOF team wishes to uses this project for, i.e. from observing the long-term effects of diseases or new drugs, to predicting recovery trajectories in patients and the like, I believe it is far more dangerous to have a false negative than a false positive. Naturally, in a medical context, neither should ideally ever happen. However, were a patient to be told they will be healthy again within x amount of time, so their treatment is stopped after that time and they are sent home, but later it is discovered that the data was wrong and the patient, despite perhaps feeling better, was by no means ready to be discharged from the hospital, I believe the consequences for the patient could be much worse than the other way around. Similar points could be made for the long-term effects of drugs or diseases; if data suggests a patient can be discharged from a doctor's or hospital's close observation, immediate care in an emergency could come too late. On the other hand, if a patient is expected to stay in an intensive or emergency care unit for x amount of time because the model suggested it, but then they eventually recover much quicker than predicted, health emergency could certainly be avoided. It is for these reasons that I decided to focus on which model displays the best recall performance, although my argument of course assumes that the classifier predicted a low level for the category, as this is an indication of a patient's poor and/or dangerous functioning in this category.

It should of course be mentioned that any and all machine learning predictions in medical (and most other contexts) should never be taken lightly. A careful medical institute should always conduct vigorous tests regarding their patients and never simply rely on what a computer suggests. Human judgement should always also be consulted.

5.4.1 Comparisons

Please see figure A.4 in the appendix for an easier overview of all result tables of all models. Once again, please note that in these comparisons I will mostly be focusing on the models' recall performances, as I have discussed their overall performances and their performances compared to the baseline in the previous sections of this chapter.

After having made my decision regarding which metric I find most important for my project, I calculated each model's average, median, and highest and lowest recall score, which can be seen in table 5.3.

Model	Average Recall	Median Recall	Highest Recall	Lowest Recall
ModelHQ1	0.832	0.945	0.976	0.103
ModelHQ2	0.825	0.921	0.975	0.099
ModelHQ3	0.831	0.926	0.975	0.112
ModelLQ1	0.813	0.909	1.0	0.181
ModelLQ2	0.827	0.913	1.0	0.162
ModelLQ3	0.789	0.871	0.969	0.222
ModelLQHQ3	0.772	0.868	0.959	0.222

Table 5.3: Average, median, and highest and lowest recalls of all models

I decided to consider each model's median recall as well because I wanted to see in what general range the model was performing. However, since a median is not a perfect representation of that (as, for example, the median in [1, 2, 30, 530, 6730] would be 30, despite not being a good representation of the list), I decided to also include the average recall and the highest and lowest recall. It should be noted here that the lowest recall for every single model without exception was from the O category. As one can see in table 5.3, the model with both the highest average, median, and general highest recall is *ModelHQ1*, though *ModelHQ3* is quite close, even surpassing the other in the lowest recall score. Another interesting trend that can be observed from this table is that for the HQ models, the performance *decreases* for the second model (at least in terms of average and median recall), but then *increases* again for the third model, though not higher than the first model. The LQ models, on the other hand, first *increase* in performance, then *decrease* even below the performance of the first model. Therefore, *ModelHQ1* is the HQ models' best performing one and *ModelLQ2* the best one for the LQ models. Interestingly enough, *ModelLQHQ3*, the model trained on the most data, was the worst performing one out of all the models, both in average, median, and highest overall recall.

Results such as these, especially in the recall category, with its tendency for more false positives, suggests a problem of overfitting within the training data. These results indicate that, if in doubt, the model is more likely to assign *a* class to the instance than none. This could also be a direct consequence of the missing negative examples in the training data, as the models may not have been trained enough to recognize what a none-sentence looks like. This is also supported by the extremely low O recall scores throughout all models, which rarely range higher than 0.2, some even as bad as 0.099 (*ModelHQ2*). Where all models had higher recall than precision scores in the other classes, the O class always showed higher precision over recall. This means that the models only found an extremely small amount of the true O sentences, despite it being the most represented class in development set. One should not forget, however,

that these models did not receive *no* negative examples, as their base classifier, Kim (2021)'s classifier, was indeed trained on those as well during its creation. But it seems as though the negative examples from this training of the base classifier were not enough to properly teach my models what a none-sentence looks like, especially since my newly added training data contained zero negative examples due to the division of high and low quality data (as described in chapter 4).

Another interesting fact to point out is that the highest recall score of 1.0 can be found in *ModelLQ1* and *ModelLQ2* in the BER category. A look at both of their full confusion matrices (not shown here) reveals that the models predicted all 7 of the BER instances correctly, which explains the perfect recall score and the rather poor precision score. I believe it is this perfect score that made the models' overall average recall score appear so high, while the rest of their recall scores are rather ordinary.

All models, without fail, have a recall score of 0.889 for the ATT category and all labeled 8 out of the 9 instances correctly, but then collectively made the same mistake as mentioned in table 5.2. Reasons and explanation for this were discussed earlier.

Category	**Highest Recall Score**
O	*ModelLQ3*, *ModelLQHQ3*
ADM	*ModelHQ2*
BER	*ModelLQ1*
ENR	*ModelHQ1*
ETN	*ModelHQ1*, *ModelHQ2*, *ModelLQ2*,*ModelLQ3*
FAC	*ModelHQ1*
INS	*ModelHQ3*
STM	*ModelLQ1*, *ModelLQ2*

Table 5.4: Highest recall score per category

As can be seen in table 5.4, *ModelHQ1* is also the model that has the highest number of highest recall scores per category, where it alone dominates the ENR and the FAC category and shares first place with *ModelHQ2*, *ModelLQ2*, and *ModelLQ3* in the ETN category. *ModelHQ1* has 730 ENR, 1001 ETN, and 1206 FAC training instances in its training set. For the ENR category, the model managed to label more true positives correctly than not (51 true positives out of 54 total instances, 28 false positives in class O), though in the ETN and FAC categories, the model mislabeled more O instances as the respective class (ETN: 81 true positives, 111 false positive O instances, FAC: 83 true positives, 115 false positive O instances). Given that ENR had the least amount of training instances, it is not surprising that it showed less signs of overfitting, unlike the other two categories, which both had over 1000 training instances and caused the classifier to assign ETN or FAC when in doubt. This did generate higher recall scores for the two latter categories, though also much poorer precision scores (both around 0.4), while ENR may have a slightly lower recall score, though a much better precision score (0.62). This leads to far more balanced scores all around, as well as a much better F1 score (0.75, while the other two have 0.58 and 0.56 respectively). Scores such as these would be highly desirable for any classifier, as a balance of recall and precision is often the ideal performance, which is why I believe that the amount of ENR training instances for *ModelHQ1* seems to be the optimal amount for good performance and less overfitting.

Regarding ETN in the other models, it has a precision score of around 0.41 in all

models except *ModelHQ3*, where it is slightly higher (∼+0.01), and a recall score of around 0.95/0.96 in all models. This category had an extremely high amount of training instances in every model (starting at 1000 already), which appears to have been enough to cause overfitting, as represented by the low precision scores in all models. It seems STM, however, with similarly numerous training instances, was not overfitted for as much during training, as its precision score may be lower than its recall score, though nowhere near as low as ETN's. Interestingly enough, *ModelLQ1* and *ModelLQ2* had the highest recall scores in the STM category, a category in which *ModelLQ1* had 2081 training instances, and *ModelLQ2* had 3921 training instances. This is obviously much higher than *ModelHQ1*'s ENR training instances, as discussed above, though the two models perform exceptionally well not only in the recall category, but in the precision category as well (both between 0.63 and 0.65). A look at all three models' more detailed confusion matrices (not depicted here) shows that *ModelLQ1* and *ModelLQ2* had 96 true positive STM classifications, and *ModelHQ1* had 94, so very close. *ModelHQ1* and *ModelLQ2* both have 55 false positive instances for the O category, while *ModelLQ1* has 51. However, where *ModelLQ1* and *ModelLQ2* have only one false negative instances for the STM category, *ModelHQ1* has 3, which would explain *ModelLQ1* and *ModelLQ2*'s better recall performance in that category. Perhaps the STM training data was a lot clearer overall and/or aligned well with the STM instances in the test data, since it seems much less overfitting occurred during training for this category. It seems that too much training data for any class in the training set leads to overfitting for most models, though it is important to remember that high training instances amounts for poorly represented classes in the test set tend to make the results unreliable. As always, balanced training, development, and testing sets are truly important for a good assessment of the classifier's performance.

Throughout all models, the INS category continuously has the worst recall score of all categories, usually ranging somewhere between 0.55 and 0.69, with a small outlier of 0.75 in *ModelHQ3*. With equally poor precision scores across all models, it is clearly by far the worst performing category everywhere. Having rather high amounts of training instances in all models, it becomes clear that such an amount led to severe overfitting. A look at some of the models' more detailed confusion matrices also shows that the models tended to label an O sentence as INS twice as often as they labeled true INS instances as such.

Four out of the seven models have an MBW recall score of 0.975, despite having rather different precision scores across all models in that category (although they all still range from 0.51 to 0.57). The other three models also display quite formidable MBW recall scores, ranging from 0.9 to 0.95. MBW has consistently less (even *much* less, in the case of the LQ models) training data than INS, which is once again evidence that imbalanced representations of classes during training simply leads to overfitting in the higher represented classes, as can be seen by the much poorer INS performance scores than MBW across all models.

After these comparisons and close observations, it is clear that *ModelLQHQ3* performed the worst out of all the fine-tuned models. Its precision scores are more often than not substandard and its recall scores only cross the 0.9 mark 3 times. It consistently has the worst recall scores out of all models or lies at the lower boundary of the range, with a small outlier in the FAC category, where it was only second worst. However, since it was fine-tuned with the most data, which therefore also differed the most from the original data it was trained on, I do not believe overfitting here to be

the main problem. Rather, I believe that so much new, dissimilar data made the model un-learn certain patterns or contexts it had learned about the classes during previous training. This would also explain its worse performance compared to the baseline.

In conclusion, after learning about how and why each model performs the way it does and looking more closely at why that could be, I decided that *ModelHQ1* is the best performing one out of all my fine-tuned models and was the one I used on the final test set. Even though it still did not perform as well as the baseline in terms of precision scores, the model did display the highest recall score. However, since a balanced model is important, I also decided to look at the models' average precision scores. The model with the highest average precision score was *ModelLQHQ3* with a score of 0.57, while *ModelLQ3* was close behind, also with 0.57. *ModelHQ1* had an average precision score of 0.56, so I decided that if this model is still so close to the highest average precision score of all models, it would still qualify as a well balanced model.

As a final answer to the research questions "How much does the quality of the training data influence the model's performance?" and "How much does the quantity of the training data influence the model's performance?", it has become clear from this investigation that the quality of the data does not seem to influence a model's performance much. Both HQ and LQ models were consistently trained on similar amounts of data, but the LQ models did not perform worse overall, it was only the recall scores that were poorer in comparison to the HQ models. The precision scores, however, were predominantly better than the HQ models'. On the other hand, the data's quantity did seem to influence the models' performances, especially in terms of comparing *ModelLQHQ3* to the other models, as adding more dissimilar fine-tuning data the model appears to have made it un-learn associations it had learned about the categories in its initial training. This is shown by its poor performance scores and is not surprising in regards to the fact that the data the base classifier was trained on and the data from the development set came from the same project, while the fine-tuning data was a different kind of medical data. Therefore, the extent to which the quantity of the training data influences the models' performance depends on the *type* of data the models are fine-tuned with, as well as what kind of data they are tested on.

Chapter 6

Results and Error Analysis

This chapter will be presenting and discussing the results of the best model decided on in chapter 5, *ModelHQ1*, on the final test set, as well as conducting a more detailed error analysis over these results. This test set, as mentioned above, was created through the same process as in figure 5.1, except this time it was all of Galjaard (2022) and Badloe (2020)'s data put together to create the final test set. For its ICF category distribution, please see table A.9 in the appendix. This chapter will summarize the answer to the previously asked research question "What specific categories do not benefit from semi-supervised learning?" Section 6.1 will be presenting and analyzing the model's results on the test set, including a comparison to the baseline and a comparison to the performance on the development set. Potential reasons and explanations will be provided in section 6.2, on the other hand, as well as detailed looks at misclassification examples, and other interesting outliers or trends among the results.

6.1 Results

This section will be discussing *ModelHQ1*'s results on the test set. Please see figure 6.1 for its full performance.

Right away, it is clear that the overall F1 scores on the test set are far superior than the ones on the development set. However, for all except the O category, the reason for this seems to be the higher precision score, rather than the recall score. Only 4 of the categories (ADM, BER, ENR, and INS) had both of their metrics increase, the other ones showed an increase in precision and a decrease in recall. But most of the recall scores did not decrease by that much, except for ATT, which decreased by around 0.07 points, and MBW, which decreased by around 0.13 points. Overall, though, the F1 scores for all ICF categories are quite formidable, ranging predominantly between 0.7 and 0.85, with only two outliers (BER and INS) with scores of around 0.6, and the O class with a score of 0.27. *ModelHQ1*'s lowest score overall on the test set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.690058	0.849162	0.900000	0.441176	0.734266	0.593897	0.580071	0.500000	0.622449	0.656566
Recall	0.169784	0.952978	0.818182	0.937500	0.929204	0.954717	0.953216	0.770833	0.841379	0.970149
F1 score	0.272517	0.898080	0.857143	0.600000	0.820312	0.732272	0.721239	0.606557	0.715543	0.783133

Figure 6.1: Results of *ModelHQ1* on the test set, *support - O: 470, ADM: 319, ATT: 11, BER: 16, ENR: 113, ETN: 265, FAC: 171, INS: 96, MBW: 145, STM: 134*

(if the O class is not counted) is 0.44, for the BER precision score, which is indeed an improvement over the development set, where the lowest score was 0.35 (if the O class is not counted) in the INS precision score. On this set as well, the model has higher recall than precision scores in all categories except for O and ATT, where that is reversed. Categories the model previously did not do well on in terms of precision (BER, ETN, FAC, INS, and MBW) did indeed improve by quite a lot in the test set performance. In regards to recall, the model's highest score went from 0.976 in FAC to only 0.97 in STM, whereas its lowest score (if the O class is not counted) increased from 0.69 in INS to 0.77 in INS, which shows a truly small regression in the highest score and a rather significant improvement in the lowest score. The model's performance even increased in all metrics for the O category, even by 1.0 points in terms of precision, leading to an increase of 0.9 points in the F1 score. Were one to rank the precision scores in both the test and development set performances from highest to lowest, the first half would always include ATT, O, STM, ADM, and ENR, even if their order within that half would not be the same each time. MBW, ETN, FAC, INS, and BER are always the worse performing categories, with their order being almost identical for both. This shows that the model definitely has categories where it generally performs better and categories where it generally performs worse (especially as this is also the case in the comparison to the baseline performance).

Compared to the baseline, *ModelHQ1* did significantly better in almost all categories in all scores, except for the O class and the recall score for FAC and MBW. The recall score of the baseline is more than 1.0 points higher than the one of *ModelHQ1* in the MBW category, but only 0.2 points higher for the FAC category, meaning that, including the O class, the baseline has only two scores across the entire table that are significantly better than *ModelHQ1*'s scores. Given that the baseline's performance, especially in the precision metric, was already so poor, it is not surprising that another model did better in that category, though the baseline's recall scores were quite formidable, so higher scores there are indeed an improvement. With the baseline's F1 scores often ranging only between 0.2 and 0.53, *ModelHQ1*'s F1 scores of between 0.6 and 0.898 (if the O class is not counted) are already much higher.

In summary, *ModelHQ1* did much better on the test set than it did on the development set and than the baseline did on the test set. Its general scores are also not subpar at all, with a more balanced performance overall, but still formidable recall scores, which is what I did deem the more important metric for this experiment. Except for a few categories, though, even the model's precision scores are acceptable, especially since a balance between recall and precision will always be necessary and with an increase in one score, one usually has to accept a decrease in the other. Given that a good classifier should be trained with such a balance in mind, this model would certainly not disappoint. Therefore, looking at results such as these, as well as all the ones from subsection 5.4.1, I believe the answer to the question "What specific categories do not benefit from semi-supervised learning?" is that there are no categories that do not benefit from semi-supervised learning. Though there indeed certain categories that tended to have lower scores than others, as mentioned above, there were no scores that never changed or appeard to be stuck in a local minimum. I therefore believe that with more training and experiments, the performance scores in all categories could indeed be improved and the model balanced out more for a more realistic performance.

	pred:	pred:ADM	pred:ATT	pred:BER	pred:ENR	pred:ETN	pred:FAC	pred:INS	pred:MBW	pred:STM
true:	118	51	1	18	35	164	105	69	69	65
true:ADM	11	304	0	0	0	0	1	2	0	1
true:ATT	1	0	9	0	0	0	0	0	0	1
true:BER	1	0	0	15	0	0	0	0	0	0
true:ENR	5	0	0	0	105	1	1	1	0	0
true:ETN	6	1	0	0	0	253	2	0	2	1
true:FAC	7	0	0	0	0	0	163	1	0	0
true:INS	6	2	0	1	2	1	9	74	1	0
true:MBW	15	0	0	0	0	7	0	1	122	0
true:STM	1	0	0	0	1	0	0	0	2	130

Figure 6.2: *ModelHQ1*'s confusion matrix on the test set, *support - O: 470, ADM: 319, ATT: 11, BER: 16, ENR: 113, ETN: 265, FAC: 171, INS: 96, MBW: 145, STM: 134*

6.2 Error Analysis

This section will provide a detailed error analysis of *ModelHQ1*'s performance on the test set, which will include error types, distribution, significance, sources, and suggestions on how to avoid such errors in the future. Please see figure 6.2 for a detailed confusion matrix of its performance and please note that "pred:" and "true:" stand for the O class. Further, it should be noted here that a lot of the sentences in the test (and development) data were often the same sentence twice, such as "Hij heeft moeite met lopen. Hij heeft moeite met lopen," though they were displayed only once here, for space and readability reasons. For further information about this, please see Badloe (2020).

As is immediately clear from this table, the most common type of error, as mentioned in multiple sections above, is false positives. The largest numbers can be found in the diagonal, i.e. the true positives, and in the first row, which shows that the model classified none-sentences quite often as one of the other classes. Since most none-sentences had already been filtered out by Kuan (2023)'s binary classifier, the ones that did remain in the test set were most likely sentences that would indeed be difficult to classify as O. There were far fewer confusions among the classes themselves, as the few and low numbers across the columns suggest. Only within the O category itself were there a few more false negatives, especially in the ADM and MBW category. This would make that the second most common type of mistake, although its numbers are still very little compared to how many false positives the model assigned.

6.2.1 False Positives

I would like to point out a few false positives for the four classes with the two lowest precision classes, as low precision means that the model overclassified certain classes when they should not have been assigned to that sentence, which is how false positives are counted. The four classes discussed in this section will be BER, ETN, FAC, and INS, as all their precision scores are all below 0.6. Taking a closer look at the mistakes made in these categories will reveal trends and patterns the model tends to follow when misclassifying an unseen instance.

False Positives BER

The false positive BER examples can be found in table 6.1. As argued before, a possible case for a BER gold label could be made for the first sentence, as it describes that the patient could only work around 2 hours, was tired, and did not have a lot of strength in their hands, but perhaps these descriptions are too vague for a concrete labeling. However, the first and the second sentence included the word *werk* ("work") and "school," which could be why the classifier assigned it the BER label. Although the second sentence seems to be more describing a patient's background in the work & employment category (which would include school for children), not anything about their functioning in this regard, the model labeled this sentence as BER, most likely due to the word "school" in the sentence. The third sentence, labeled INS, as it talks about the patient's exercise tolerance functioning, was also labeled BER by the model. This was the only BER-labeled sentence that was originally belonging to a different ICF category, rather than the O class. Since it also contains a version the word *werk*, it seems likely this is the reason the model labeled it BER instead of INS. All three of these examples once again show a strong reliance on keywords by the model, as well as proof of overfitting.

Original sentence	*Translation*	Gold	Predicted
moe, zwaar werk vandaag gehad, ook iets langer dan 2 uur gewerkt, weinig kracht in handen. gaat	*tired, had a hard job today, also worked a little more than 2 hours, little strength in hands. go*	-	BER
Gaat naar 3e klas HAVO, gaat goed op school Vindt het leuk om naar buiten te gaan met vriendinnen	*Goes to 3rd grade HAVO, does well at school Enjoys going outside with friends*	-	BER
Lichamelijk onderzoek Algemene indruk: wat bleek, wakker en alert Karnofsky - score: 70 - in staat tot zelfverzorging. niet tot werkzaamheden WHO - classificatie: 2 - in staat voor zichzelf te zorgen, niet om te werken.	*Physical examination General impression: somewhat pale, awake and alert Karnofsky - score: 70 - capable of self-care. not able to work WHO classification: 2 - able to care for themselves, not able to work.*	INS	BER

Table 6.1: False positive BER examples from *ModelHQ1* on test set

False Positives ETN

The false positive ETN examples can be found in table 6.2. As can be seen in figure 6.2, MBW was the category with the most false positive ETN mislabelings (except for O). This is not surprising, since ETN describes a patient's eating functioning and MBW their weight maintenance functioning, which are already heavily related. Three out of the four examples here include the word *voeding* ("nutrition," or in combination with *sonde*, "feeding"), while the second example includes the word *eten* ("to eat"), so

despite the MBW category describing similar topics as the ETN category, it seems as though the model was easily swayed by these keywords. This is further proven by the fact that the O example does not talk about a patient's functioning at all, only about a feeding tube shortage. Furthermore, it is quite interesting that the ENR example even includes the word *energie* ("energy"), yet the model seems to have given the word *eten* more attention. Perhaps this was due to other patterns the model had learnt during previous training, especially since this does look like a rather ambiguous sentence, that could potentially fit in either category.

Original sentence	*Translation*	Gold	Predicted
er is een tekort aan enterale sondevoedingsslangen, graag om de 48 uur vervangen en niet 1x per 24 uur.	*there is a shortage of enteral tube feeding tubes, please replace every 48 hours and not once every 24 hours.*	-	ETN
Eten kost veel energie	*Eating takes a lot of energy*	ENR	ETN
Het gaat verder goed met patiente, conditie is verbetert, heeft geen sondevoeding meer nodig	*The patient is doing well, her condition has improved and she no longer needs tube feeding*	INS	ETN
Voedingstoestand (snaq score): 4 Diëtiste nodig ja, heeft contact met diëtiste in het OLVG	*Nutritional status (snaq score): 4 Dietitian needed yes, in contact with dietician at the OLVG*	MBW	ETN

Table 6.2: False positive ETN examples from *ModelHQ1* on test set

False Positives FAC

The false positive FAC examples can be found in table 6.3. Noticeable here again, as touched upon already in previous chapters, is that three out of the five examples have the word *lopen* in it, which seems to be the reason the classifier labeled these as FAC. The INS example, although indeed describing a patient's exercise tolerance and not including a version of *lopen*, does describe how patient mobilized herself on a chair, so it is not implausible that this sentence could also be labeled FAC. The ADM example, consisting of 3 words only, shows no signs of previous patterns or keywords that could explain why the model labeled this instance as FAC, especially since it includes the word *ademen* ("to breathe") and, given the model's previous patterns, should have alerted the model of the sentence's belonging to the ADM category, yet it did not cause that. It is a strange instance indeed.

False Positives INS

The false positive INS examples can be found in table 6.4. Though all of these are false positive examples. i.e. examples, where the model *falsely* labeled an instance INS, all five of these examples do indeed describe something related to a patient's exercise tolerance functions. There are mentions of walking, exertion, balance exercises, and exercise bikes; all of which could be considered related to INS sentences. However, the

Original sentence	*Translation*	Gold	Predicted
PPF: D Analyse: - Algemeen Looptraining met rollator -Conclusie uit subjectieve en objectieve gegevens	*PPF: D Analysis: - General Walking training with a walker - Conclusion from subjective and objective data*	-	FAC
Piepje bij ademen	*Beep when breathing*	ADM	FAC
-Dhr uitgedaagd om met de handen los de knieen hoog op te tillen tijdens het lopen.	*-Mr challenged to lift his knees high with his hands while walking.*	ENR	FAC
dhr wilde niet met rollator naar toilet lopen DIG: redelijke intake, heeft bijna hele avondmaaltijd opgegeten.	*Mr. did not want to walk to the toilet with a walker. DIG: reasonable intake, ate almost the entire evening meal.*	ETN	FAC
In de middag heeft mw gemobiliseerd op de stoel en maakt gebruik van de fiets trappen van de fysiotherapeut.	*In the afternoon, Mrs. mobilized on the chair and used the physiotherapist's bicycle pedals.*	INS	FAC

Table 6.3: False positive FAC examples from *ModelHQ1* on test set

ADM example describes when the patient has *dyspnoe* ("dyspnea"), which is clearly related to the patient's breathing levels, yet it is also a rare, uncommon word. The model may have been unsure about it and instead focused on the word *inspanning* ("exertion"), which categorized the instance as INS. The ENR and MBW examples both seem indeed heavily related to the INS category and could feasibly have been gold labeled as such as well, so the model's confusion here is understandable, but perhaps the words *balans* ("balance") and *actief* ("active") were what eventually swayed the model to INS. Regarding the O example, although the model seems to have firmly associated *lopen* with FAC, it may be that the word *wandeling* ("walk", noun) made the classifier believe that the INS category is more appropriate in this case, though.

6.2.2 False Negatives

As mentioned above, the ADM and MBW categories are the two categories with the most false negatives, i.e. instances where the classifier labeled a sentence as *not* ADM or MBW, when in fact it was one of these classes. As these mostly happened within the O category and the other cases were sufficiently discussed above, this section will only focus on false ADM and MBW instances within the O class.

False Negatives ADM

The false negative AMD examples can be found in table 6.5. Interesting to note here already is the mention of the word *lopen* in the first example, and yet the classifier labeled this instance not as FAC, but as O. The sentence also includes the word *adem* ("breath," noun), but as discussed above, it does not seem to be a trigger word for the

Original sentence	*Translation*	Gold	Predicted
Geen adjuvante chemotherapie A/ Komt enkel buiten voor een wandeling en boodschap.	*No adjuvant chemotherapy A/ Only goes outside for a walk and errands.*	-	INS
Enkel dyspnoe bij zware inspanning	*Only dyspnea during heavy exertion*	ADM	INS
zich fit O covid revalidatie P HUR, fietsen 80-120 watt, balansoefeningen	*feeling fit O covid rehabilitation P HUR, cycling 80-120 watts, balance exercises*	ENR	INS
Loopt wat, bukken gaat niet zo goed, wordt dan duizelig	*Walks a bit, bends down not so well, then becomes dizzy*	FAC	INS
Pre - operatief stabiel gewicht, actief leefpatroon (dagelijks 5 km wandelen, hometrainer)	*Pre-operative stable weight, active lifestyle (5 km walk daily, exercise bike)*	MBW	INS

Table 6.4: False positive INS examples from *ModelHQ1* on test set

model to label that instance as ADM. The word was either not prominent enough in the training data to cause such a correlation – although that is highly unlikely, as it is basically the title word of the category – or the model simply used other patterns and context to distinguish ADM sentences from other classes. Given the model's high precision and recall score in this category, that one seems the more likely explanation.

Original sentence	*Translation*	Gold	Predicted
ongeveer 100 meter lopen op viak terrein moet ik na een paar minuten stoppen om op adem te komen	*After walking about 100 meters on rough terrain, I have to stop after a few minutes to catch my breath*	ADM	-
Kan niet meer platliggen vanwege de pijn, in combinatie met de benauwdheid.	*Can no longer lie flat because of the pain, in combination with the shortness of breath.*	ADM	-
Hoestte en voelde zich kortademig na inspanning.	*Coughed and felt short of breath after exertion.*	ADM	-

Table 6.5: False negative ADM examples from *ModelHQ1* on test set

False Negatives MBW

The false negative MBW examples can be found in table 6.6. Similar as with the false negative ADM examples, all three of these examples include a version of the word *gewicht* ("weight"), which is once again a title word of the category, yet none of these three sentences was labeled as MBW. Based on this, it also seems unlikely that *gewicht*

was not prominent enough in the training data, but unlike ADM, the MBW category does not have such outstanding precision and recall scores, only 6.2 and 8.4 respectively. Given that two of the three examples include numerical characters, there is a chance that these confused the model too much to make a decision about this sentence, though this is also unlikely, as a lot of the MBW instances include numbers and were labeled correctly. The last sentences includes both *voeding* and *gewicht*, yet is also neither labeled as ETN nor MBW, which is rather hard to explain. All three of these examples (as well as other, not depicted ones) show all the signs that they should have been assigned the correct category, or in some cases perhaps the ETN category, yet none of them are. Further in depth research could be conducted to find an explanation for this.

Original sentence	*Translation*	Gold	Predicted
mevrouw heeft een goede voedingstoestand, risico op ondervoeding obv stabiel gewicht.	*The lady has a good nutritional status, risk of malnutrition based on stable weight.*	MBW	-
GPE verbeterd van 63 naar 85. Huidige voedingstoestand lijkt redelijk obv stabiel gewicht.	*GPE improved from 63 to 85. Current nutritional status seems reasonable based on stable weight.*	MBW	-
Slechte voedingstoestand, totaal gewichtsverlies van 25% waarvan 20% postoperatief.	*Poor nutritional status, total weight loss of 25% of which 20% postoperatively.*	MBW	-

Table 6.6: False negative MBW examples from *ModelHQ1* on test set

6.2.3 Conclusion

Overall, the performance of *ModelHQ1* on the test set is quite good. It is not only good compared to the baseline model's performance on the test set, it is also good on its own, with F1 scores rarely dipping below 0.7 and recall scores predominantly in the 0.9-1.0 range. Even its precision scores, which are not quite as high at its recall scores, are average and although the ENR, ETN, FAC, and INS categories still show room for improvement, the model still performs adequately. Compared to the baseline, the model also performs much better, though one should not overlook the fact that the baseline model performed extraordinarily poorly on the test set. Usually, when testing classifiers on development and test sets, they tend to perform better on the development set, given that more tweaks for optimal performance can be done based on the results. In this case, however, the model performed better on the test set. Since both sets are noticeably small, though, especially compared to the data the model was trained on, these results should still be regarded with caution, as small test sets return unreliable and somewhat inconclusive results. A larger, more balanced test set would provide more accurate results.

Section 6.2 shows that most errors the classes suffered from where false positives, especially when it came to classifying O sentences. As these had already passed Kuan (2023)'s binary classifier and were therefore only the most difficult cases left, this does

make sense. More often than not, the model labeled these as not-O and seemed to do so based mostly on certain keywords that often (though not always) appeared to make the classifier decide on one of the classes, such as *lopen* for the FAC and *werk* for the BER class. Since these are indeed quite indicative words for these categories, there is always the possibility that it was annotation error, rather than a classifier error, as these cases passed the binary classifier for a reason. However, the classifier also does not seem to be able to handle rare or uncommon words, such as *dyspnoe* or *woordvindstoornis*, which it tends to ignore and simply focus on words it seems to be more familiar with when it comes to deciding what class the sentence should be. However, the model was not always outright incorrect, as I believe that a minor fraction of the false positives resulted from to the presence of annotation errors, which is why computing a strict IAA-score is always advisable.

The false negatives the model showed in its performance data appeared mostly in the ADM and MBW classes, both of which had strong potential keywords in their examples that the classifier still did not pick up on. ADM had the most training instances in the model (over 3000), while MBW only had a bit more than 600, which could explain the slightly worse recall and precision score for the latter class. ADM also had more than twice as many testing instances than MBW, once again being the most represented class, yet it was these two classes that the classifier struggled the most with identifying correctly and rather labeled them as O. As mentioned above, the small test set will yield unreliable results.

Chapter 7

Discussion and Conclusion

This chapter will encompass a discussion and conclusion of the project's findings. section 7.1 will delve into the discussion of the experiments, while in section 7.2, I will address the project's limitations and offer recommendations for future research. Finally, section 7.3 will mark the conclusion of this project.

7.1 Discussion

After experiments and research, three main points solidified: 1) "High quality" data does not automatically improve a classifier's performance, 2) High quantities of data not only do not automatically improve performance, they seem to make the model unlearn already learned patterns, and 3) Training and evaluation data should be balanced in terms of type and quantity for accurate results. While this project set out to research the difference between "high" and "low" quality data versus high and low quantity of data, it seems that was not quite what was discovered during this task.

7.1.1 Data Quality

As seen in chapter 6, difference in the quality of data mostly influenced precision versus recall scores. "High" quality data seemed to have increased the models' recall scores, while "low" quality data seemed to have increased their precision scores. This is surprising in and of itself, as one would expect this the other way around, since normally clear and straight-forward data teaches the model quite precise examples of what a certain category should look like. This would then lead the model to be more meticulous when deciding whether an instance of a certain class or not, instead of assigning the category despite a rather low confidence score. One could argue, of course, that in my case, since I decided to base my best model on the model with the best recall scores, the high quality models did indeed perform better and therefore high quality data does indeed improve performance, but depending on what one's research task and focus is, one could decide on either high or low quality data and still create well performing classifiers for the task. That is why I do not wish to claim that higher quality data = higher performance scores. Further, one should also remember that "high" quality data is quite arbitrary, hence the quotation marks, and that is was decided based on one classifier's confidence score when labeling this data. That is of course neither the only, nor the best method to go about this, as the real quality of an instance should be decided by a human annotator during the data labeling. Only

a human with the right expertise knowledge could decide whether a sentence is of accurate quality to be labeled “high” or “low.”

7.1.2 Data Quantity and Similar Sets

Also visible in chapter 6, the quantity of the data stands in strong correlation with the models’ performance. *ModelLQHQ3*, the model with the most training data, was one of the worst performing models in terms of recall, which I believe to be caused by the addition of too much dissimilar fine-tuning data, both in terms of learned patterns and alignment with the test set. The base classifier may have been trained on large amounts of secondary care level COVID-19 data, which is also prominent in the development set, but the data I added for fine-tuning and the test set included other types of data and the model was either not able to transfer what it had learned in pre-training to the new type of data, or the new fine-tuning data altered its learned patterns and associations too much. This would also explain why *ModelHQ1*, one of the two models with the least amount of training data, performed so well in the recall metric, as it was simply still rather similar to the baseline in terms of what it had learned. Another quantity issue was the fact that the O category consistently performed so poorly on all models, most likely because the new training data did not include any negative samples and the models only had what their base classifier was trained on. However, the O sentences that were included in the test and development sets were the ones that had passed the filtering of Kuan (2023)’s binary classifier, so they were the more difficult cases to classify. All the mentioned issues led to overfitting for all ICF categories or distorted the results, which is why I believe that either less training data, more evaluation data, or more similar types of training and testing data would have made for more accurate results and better performing models.

7.2 Future Work

To improve the future work and research of my project, I would recommend balancing out the fine-tuning data to prevent overfitting for certain classes. This could be done by enhancing the set with more annotated data, though that seems like a lot of work when the data is already available. That is why, after running the data through Kim (2021)’s classifier the first time to receive the confidence scores, the process could be repeated until each category has a balanced representation within the training set. Further, since the amount of fine-tuning data I provided my models with was not nearly as much as their base classifier had received during training and the added, different data simply confused my models, more training data should be added in the future, so that the models can be trained on equal amounts of different types of data and learn all possible patterns. This is further supported by the fact that, out of the HQ models, *ModelHQ1* performed the best, *ModelHQ2* the worst, and *ModelHQ3* only slightly worse than the first. This shows that adding only a little new data to the training was not enough to confuse the classifier, though adding a medium amount brought in just enough new and different data that the classifier was starting to un-learn certain patterns. Once again adding more data gave the classifier more data to also learn the patterns of the newly added data. *ModelLQHQ3*’s performance shows that adding too much low quality data simply confuses it again, which is why adding more of the desired kinds of training data is necessary to further improve performance. Since my

semi-supervised approach to training a classifier did not fail or produce overall abysmal results, this should be easily done by retrieving more of the millions of unannotated hospital notes and using Kim (2021)'s classifier to have them labeled before using them for training. This would avoid elaborate and costly human annotation. Moreover, the test and development sets should be made up of equal amounts of the types of data used in training, as well as be large enough not to present inconclusive results.

Furthermore, I could have used the data with a confidence score lower than my set low-quality-threshold as negative examples and added it to the training set of both the HQ and the LQ models. One could also, of course, play around with the threshold of Kim (2021)'s classifier, of when an instance switches from 0 to 1. Again, it was set to 0.5 during my experiments, but experimenting with that and trying out different thresholds here could potentially lead to the desired balance between recall and precision.

Lastly, as mentioned in chapter 6, some of the sentences in the test and development sets were duplicated, which is an error that occurred during Badloe (2020)'s project. At this point in the project, it is difficult to determine the specific impact and magnitude of influence on the results these duplications had, but it is important to remember this error, were this experiment ever be reproduced. For the sake of better comparability across the whole A-PROOF project, sentences should be single sentences in any future work. It should also be mentioned that while Kim (2021), Galjaard (2022), and Badloe (2020) conducted their experiments on both a sentence level *and* a note level within the data, this academic year us students focused solely on a sentence level performance. We had quite ambitious goals for our theses and knew that further experiments for note level performances could be easily derived by future students from our current research.

7.3 Conclusion

In subsection 7.3.1, this section will summarize the project's initial goals and its final results by referring back to the research questions and sub-questions introduced in chapter 1. Subsection 7.3.2 will present the project's conclusion.

7.3.1 Summary

This thesis aimed to answer the question **"Does using semi-supervised learning to train a model improve the model's performance in automatically annotating unlabeled hospital notes?"** In order to do so, three sub-questions were proposed that the project focused on answering during the process:

1. How much does the quality of the training data influence the model's performance?

2. How much does the quantity of the training data influence the model's performance?

3. What specific categories do not benefit from semi-supervised learning?

By having a classifier label a large amount of unlabeled sentences, dividing these sentences into "high" and "low" quality sentences based on the classifier's confidence score for each instance, and then training said classifier on different amounts of high

and low quality data, different models were created that could then each be evaluated on a development set until a best model was found. This model was then evaluated on the test set. Each intermediate model was compared to the baseline, which was the base classifier's performance on the development set, until the best model was also compared to the baseline's performance on the test set.

7.3.2 Conclusion

After all experiments were done and analyzed, the answer to the first question became clear: The quality of the training data does not influence the model's performance by a lot. Precision and recall seemed to be influenced in the sense that LQ models tended to have higher precision and HQ models higher recall, but neither warrants and automatic title of "better performing model." Both models of each training phase still tended to perform better in some and worse in other categories, also compared to the baseline. This is equally true for how much data the models were trained with, which answers the second sub-question, with the only exception that here higher recall scores meant much worse precision scores due to the constant overfitting of the model. With higher amounts of training data and the same, rather small development and test set, a real analysis is hard to conduct, as the results will always be made unreliable by the imbalanced ratio. Regarding the last sub-question, there seemed to be no classes that got stuck in a local minimum or never improved scores, it all depended on the previously mentioned factors. All of my findings make me optimistic that this project could easily be improved by future A-PROOF interns and then truly result in more labeled data with much less time and effort from the human annotators. Semi-supervised learning is a fascinating and promising classifier training technique that holds great potential, if done right, so I hope the A-PROOF team can make use of it for the continuation of their project in the future and continue working towards a great tool.

Appendix A

Appendix

ICF code	Domain	Abbrev.	Functioning levels scale
b1300	Energy level	ENR	0-4
b140	Attention functions	ATT	0-4
b152	Emotional functions	STM	0-4
b440	Respiration functions	ADM	0-4
b455	Exercise tolerance functions	INS	0-5
b530	Weight maintenance functions	MBW	0-4
d450	Walking	FAC	0-5
d550	Eating	ETN	0-4
d840-d859	Work and employment	BER	0-4

Figure A.1: Overview of the ICF domains in the project, taken from Kim (2021)

	pred:	pred:ADM	pred:ATT	pred:BER	pred:ENR	pred:ETN	pred:FAC	pred:INS	pred:MBW	pred:STM
true:	203	80	0	8	19	60	81	33	30	40
true:ADM	8	171	0	0	1	0	1	1	0	0
true:ATT	1	0	7	0	0	0	1	0	0	0
true:BER	1	0	0	6	0	0	0	0	0	0
true:ENR	3	0	0	0	50	0	0	0	1	0
true:ETN	12	1	0	0	0	70	1	0	0	0
true:FAC	1	1	0	0	0	0	83	0	0	0
true:INS	8	1	0	0	0	0	6	21	0	0
true:MBW	1	0	0	0	0	0	0	0	39	0
true:STM	4	0	0	0	0	0	0	0	0	93

Figure A.2: Baseline performance on the development set, part 2, *support - O: 484, ADM: 182, ATT: 9, BER: 7, ENR: 54, ETN: 84, FAC: 85, INS: 36, MBW: 40, STM: 97*

	pred:	pred:ADM	pred:ATT	pred:BER	pred:ENR	pred:ETN	pred:FAC	pred:INS	pred:MBW	pred:STM
true:	433	159	1	25	73	232	135	86	136	91
true:ADM	8	171	0	0	1	0	1	1	0	0
true:ATT	1	0	7	0	0	0	1	0	0	0
true:BER	1	0	0	6	0	0	0	0	0	0
true:ENR	3	0	0	0	50	0	0	0	1	0
true:ETN	12	1	0	0	0	70	1	0	0	0
true:FAC	1	1	0	0	0	0	83	0	0	0
true:INS	8	1	0	0	0	0	6	21	0	0
true:MBW	1	0	0	0	0	0	0	0	39	0
true:STM	4	0	0	0	0	0	0	0	0	93

Figure A.3: Baseline performance on the test set, part 2, *support - O: 470, ADM: 319, ATT: 11, BER: 16, ENR: 113, ETN: 265, FAC: 171, INS: 96, MBW: 145, STM: 134*

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.791667	0.625455	0.888889	0.375000	0.621951	0.419689	0.397129	0.352113	0.513158	0.648276
Recall	0.102888	0.945055	0.888889	0.857143	0.944444	0.964286	0.976471	0.694444	0.975000	0.969072
F1 score	0.182109	0.752735	0.888889	0.521739	0.750000	0.584838	0.564626	0.467290	0.672414	0.776860

ModelHQ1 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.657895	0.638132	0.800000	0.538462	0.672727	0.416667	0.426230	0.348485	0.565217	0.635762
Recall	0.180505	0.901099	0.888889	1.000000	0.685185	0.952381	0.917647	0.638889	0.975000	0.989691
F1 score	0.283286	0.747153	0.842105	0.700000	0.678899	0.579710	0.582090	0.450980	0.715596	0.774194

ModelLQ1 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.743243	0.643382	0.888889	0.352941	0.571429	0.415385	0.415385	0.320513	0.520000	0.630872
Recall	0.099278	0.961538	0.888889	0.857143	0.888889	0.964286	0.952941	0.694444	0.975000	0.969072
F1 score	0.175159	0.770925	0.888889	0.500000	0.695652	0.580645	0.578571	0.438596	0.678261	0.764228

ModelHQ2 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.703125	0.631783	0.727273	0.466667	0.686567	0.417526	0.403061	0.370968	0.542857	0.653061
Recall	0.162455	0.895604	0.888889	1.000000	0.851852	0.964286	0.929412	0.638889	0.950000	0.989691
F1 score	0.263930	0.740909	0.800000	0.636364	0.760331	0.582734	0.562278	0.469388	0.690909	0.786885

ModelLQ2 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.756098	0.637037	0.727273	0.352941	0.644737	0.423280	0.411168	0.313953	0.527027	0.643836
Recall	0.111913	0.945055	0.888889	0.857143	0.907407	0.952381	0.952941	0.750000	0.975000	0.969072
F1 score	0.194969	0.761062	0.800000	0.500000	0.753846	0.586081	0.574468	0.442623	0.684211	0.773663

ModelHQ3 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.637306	0.631373	0.888889	0.500000	0.666667	0.415385	0.411765	0.411765	0.569231	0.643836
Recall	0.222022	0.884615	0.888889	0.857143	0.851852	0.964286	0.741176	0.583333	0.925000	0.969072
F1 score	0.329317	0.736842	0.888889	0.631579	0.747967	0.580645	0.529412	0.482759	0.704762	0.773663

ModelLQ3 results on evaluation set

	O	ADM	ATT	BER	ENR	ETN	FAC	INS	MBW	STM
Precision	0.615000	0.634921	1.000000	0.428571	0.677966	0.416667	0.393939	0.392157	0.571429	0.645833
Recall	0.222022	0.879121	0.888889	0.857143	0.740741	0.952381	0.764706	0.555556	0.900000	0.958763
F1 score	0.326260	0.737327	0.941176	0.571429	0.707965	0.579710	0.520000	0.459770	0.699029	0.771784

ModelLQHQ3 results on evaluation set

Figure A.4: Results of all models on the development set

Category	Words
ADM	o2, spo2, peep, respiratoire, beademing, ademhalingsfrequentie, fio2, temperatuur, zuurstof, polsfrequentie, saturatie, sat, dyspneu, lengte, bloeddruk, temperatuurbron, resp, neusbril, insufficiëntie, 96, 94, insufficientie, ventilatie, 36, bpm, liter
ATT	aandacht, trekken, concentratie, behouden, helder, bewustzijn, concentratieverlies, concentreren, afgeleid, ogen, geheugen, alert, concentratieproblemen, reageert, angsten, aanspreken, lezen, concentratiestoornissen, adequaat, vergeetachtigheid, cognitie, kenmerk, spreekt, inattentie, hoofd, geheugenproblemen
BER	gewerkt, uitgevoerde, aanpak, psk, fietsen, trampoline, school, dribbelen, heuvel, infratechniek, tik, balanseren, hur, stage, uren, vorm, volhouden, beroep, sociale, waarde, werkte, seconden, th, grond, ziektewet, halve
ENR	moeheid, vermoeiend, slaperig, graad, uitgevoerde, aanpak, fietsen, zwak, vermoeidheidsklachten, fit, suf, uitgeput, hur, vermoeider, gewichtsverlies, verzwakt, vermoeide, stoel, wakker, czs, slap, balansoefeningen, watt, fitter, ochtend, indruk
ETN	sondevoeding, voeding, sv, dig, protein, nutridrink, sonde, compact, ml, eet, nutrison, drinkvoeding, tpv, duodenumsonde, gegeten, protino, voedingstoestand, geschatte, maagsonde, arla, eiwitbehoefte, dieet, orale, water, flesje, kcal
FAC	fac, gelopen, meter, toilet, stoel, trap, traplopen, mob, afdeling, supervisie, rond, veilig, infuuspaal, personen, loophulpmiddel, persoon, krukken, transfers, liep, ondersteuning, ongestoord, gang, looprek, wankel, rondje, buiten
INS	stoel, fietsen, inspanningstolerantie, wandelen, gezeten, fietst, sport, boodschappen, km, huishouden, mob, wandelt, trap, gelopen, buiten, bedrand, meter, traplopen, toilet, hond, fitness, fiets, gedoucht, trappen, mobiliseren, douchen
MBW	afgevallen, gewichtsverlies, voedingstoestand, snaq, obv, tov, ondervoeding, gebruikelijk, eindscore, voeding, inflammatie, agv, afvallen, graad, anorexie, gerelateerde, gewichtsverandering, bmi, acute, kilo, wb, toegenomen, nachtzweten, gew, onbedoeld, bewust
STM	angst, stemming, angstig, bang, somber, affect, emotioneel, boos, blij, depressieve, depressie, gespannen, angsten, verdrietig, dood, doodswens, stress, modulerend, paniek, normofoor, depressief, vrolijk, onrustig, gedachten, gevoelens, sombere

Table A.1: TF-IDF words per ICF category

Category/Categories	Amount
O	372,797
STM	8169
ADM	7094
FAC	4019
ETN	3561
INS	2268
ENR	2125
MBW	2116
BER	1342
ATT	458
FAC & INS	280
ETN & MBW	232
ADM & INS	156
ENR & INS	108
ADM & ENR	88
FAC & STM	29
ADM & ETN	25
ADM & STM	16
ENR & MBW	14
MBW & STM	11
ENR & ETN	11
ADM & INS & FAC	11
ATT & ENR	11
ADM & MBW	10
ENR & MBW & INS	7
ADM & FAC	7
INS & MBW	6
ENR & STM	5
BER & INS	4
ADM & ENR & INS	4
ETN & STM	3
ENR & INS & FAC	3
BER & INS & STM	3
ADM & ETN & MBW	3
ATT & STM	2
MBW & FAC	1
BER & STM	1

Table A.2: ICF category sentence distribution using Kim (2021)'s classifier & high quality dataset category sentence distribution

Category/Categories	Amount
STM	6410
ETN	5011
ADM	4701
INS	3584
BER	1835
FAC	1732
ATT	582
FAC & INS	487
MBW	416
ENR	224
BER & INS	143
ADM & INS	105
ETN & MBW	82
ENR & INS	78
BER & STM	48
ENR & STM	47
ATT & STM	40
INS & STM	38
ADM & STM	29
BER & ENR	19
ETN & STM	18
FAC & STM	17
INS & MBW	14
ADM & ETN	13
ENR & MBW	11
BER & ENR & INS	11
ATT & FAC	8
ATT & BNR	8
ADM & ENR	8
BER & FAC	7
ENR & FAC & INS	7
ENR & ETN	6
FAC & INS & STM	5
BER & INS & STM	5
MBW & STM	4
ADM & BER & INS	4
ATT & ETN	3
ADM & MBW	3
ADM & FAC & INS	3
BER & FAC & INS	3
ETN & INS	2
ETN & FAC	2
BER & MBW	2
ADM & FAC	2
ENR & FAC & INS & STM	2
ENR & ETN & INS & MBW	2
ATT & ENR	1

Table A.3: ICF category sentence distribution among the low quality dataset

Category/Categories	Amount
ADM	3002
STM	2381
FAC	1206
ETN	1001
ENR	730
INS	716
MBW	629
BER	554
ATT	189
FAC & INS	92
ADM & INS	53
ETN & MBW	49
ENR & INS	36
ADM & ENR	35
ADM & ETN	12
ADM & STM	8
ATT & ENR	6
MBW & STM	5
FAC & STM	4
ENR & INS & MBW	4
INS & MBW	3
ENR & ETN	3
ADM & FAC	3
ADM & FAC & INS	3
ADM & ENR & INS	2
ENR & STM	1
ENR & MBW	1
BER & STM	1
BER & INS	1
ATT & STM	1
ENR & FAC & INS	1
BER & INS & STM	1
ADM & ETN & MBW	1

Table A.4: ICF category sentence distribution among the first high quality training set

Category/Categories	Amount
ADM	2346
STM	2116
ETN	1676
FAC	1519
ENR	802
INS	750
MBW	667
BER	361
ATT	117
FAC & INS	103
ETN & MBW	96
ADM & INS	55
ADM & ENR	36
ENR & INS	31
FAC & STM	9
ADM & ETN	7
ENR & MBW	6
ADM & MBW	6
ADM & STM	5
ENR & ETN	4
ADM & FAC & INS	4
MBW & STM	3
ADM & FAC	3
ENR & STM	3
ATT & ENR	2
ENR & INS & MBW	2
ADM & ETN & MBW	2
ETN & STM	1
ENR & FAC & INS	1
ADM & ENR INS	1

Table A.5: ICF category sentence distribution among the second high quality training set

Category/Categories	Amount
STM	2081
ADM	1807
ETN	1529
INS	1072
BER	794
FAC	500
ATT	219
FAC & INS	150
MBW	131
ENR	64
BER & INS	55
ADM & INS	37
ENR & INS	33
ETN & MBW	25
BER & STM	17
ENR & STM	15
INS & STM	11
ATT & STM	11
ADM & STM	11
BER & ENR & INS	7
INS & MBW	6
ADM & ETN	6
BER & FAC	5
BER & ENR	5
FAC & STM	4
ETN & STM	4
ATT & BER	4
ADM & ENR	4
ENR & ETN	3
ATT & FAC	3
ENR & MBW	2
ATT & ETN	2
BER & INS & STM	2
ADM & BER & INS	2
ENR & FAC & INS & STM	2
MBW & STM	1
ETN & FAC	1
ENR & FAC & INS	1
ADM & FAC & INS	1

Table A.6: ICF category sentence distribution among the first low quality training set

Category/Categories	Amount
ETN	2149
STM	1840
ADM	1769
INS	1075
FAC	567
BER	484
ATT	157
FAC & INS	136
MBW	127
ENR	91
ETN & MBW	32
BER & INS	32
ADM & INS	29
ENR & INS	22
ENR & STM	19
ATT & STM	12
BER & STM	11
INS & STM	9
BER & ENR	9
ETN & STM	8
ADM & STM	8
FAC & STM	6
ENR & MBW	6
INS & MBQ	3
ADM & ENR	3
ENR & FAC & INS	3
ETN & INS	2
ENR & ETN	2
ADM & FAC	2
ADM & ETN	2
BER & FAC & INS	2
MBW & STM	1
ETN & FAC	1
BER & FAC	1
ATT & FAC	1
ATT & BER	1
ADM & MBW	1
FAC & INS & STM	1
BER & ENR & INS	1
ADM & FAC & INS	1
ENR & ETN & INS & MBW	1

Table A.7: ICF category sentence distribution among the second low quality training set

Category/Categories	Amount
O	484
ADM	182
STM	97
FAC	85
ETN	84
ENR	54
MBW	40
INS	36
FAC & INS	14
ENR & INS	11
ATT	9
BER	7
ETN & MBW	6
ADM & INS	6
ADM & ENR	5
ADM & ETN	4
BER & INS	3
ADM & FAC & INS	3
FAC & STM	2
ADM & MBW	2
ADM & FAC	2
ADM & ENR & INS	2
ENR & FAC & INS	2
INS & STM	1
ENR & FAC	1
BER & ENR	1
ATT & INS	1
FAC & INS & STM	1
ENR & INS & STM	1
ATT & ETN & INS	1
ADM & ATT & ENR	1

Table A.8: ICF category sentence distribution among the development set

Category/Categories	Amount
O	470
ADM	319
ETN	265
FAC	171
MBW	145
STM	134
ENR	113
INS	96
ETN & MBW	39
FAC & INS	30
ENR & INS	25
ADM & INS	17
BER	16
ADM & ENR	16
ATT	11
ADM & FAC & INS	10
ENR & FAC & INS	7
ADM & FAC	7
MBW & STM	6
ENR & MBW	6
ADM & MBW	6
ADM & ETN	6
BER & STM	5
BER & INS	5
INS & STM	5
ENR & ETN	4
ETN & INS	3
BER & ETN	3
ATT & ENR	3
FAC & STM	2
BER & ENR	2
ATT & FAC	2
ADM & ENR & INS	2
INS & MBW	1
ETN & STM	1
ENR & STM	1
BER & FAC	1
ATT & BER	1
ADM & STM	1
ATT & ETN & INS	1
ATT & BER & ENR	1
ENR & INS & STM	1
ADM & INS & MBW	1
ADM & ENR & MBW	1
ADM & ENR & FAC & INS	1
ADM & ATT & ENR & INS	1
BER & ENR & INS & STM	1

Table A.9: ICF category sentence distribution among the test set

Original sentence	*Translation*	Gold	Predicted
[patient]: gaat goed op school, geen probleem, ook niet thuis, ze kan zich concentreren	*[patient]: doing well at school, no problem, not even at home, she can concentrate*	-	BER

Table A.10: Further BER false positive examples from *ModelHQ1*

Bibliography

S. Badloe. 2020.

E. H. Galjaard. 2022.

Z. Ghahramani. *Unsupervised Learning*, pages 72–112. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-28650-9. doi: 10.1007/978-3-540-28650-9_5. URL https://doi.org/10.1007/978-3-540-28650-9_5.

M. F. A. Hady and F. Schwenker. *Semi-supervised Learning*, pages 215–239. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013. ISBN 978-3-642-36657-4. doi: 10.1007/978-3-642-36657-4_7. URL https://doi.org/10.1007/978-3-642-36657-4_7.

T. Hastie, J. Friedman, and R. Tisbshirani. *The elements of Statistical Learning: Data Mining, Inference, and prediction.* Springer Series in Statistics. Springer, 1 edition, 2017.

J. Kim. *Automated Assignment of ICF Functioning Levels to Clinical Notes in Dutch.* Vrije Universiteit Amsterdam, 2021.

C. Kuan. 2023.

D.-H. Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

M. Postma. GitHub - cltl/KeywordMatcher — github.com. https://github.com/cltl/KeywordMatcher, 2020. [Accessed 18-Jul-2023].

T. Rajapakse. Classification Models — simpletransformers.ai. https://simpletransformers.ai/docs/classification-models/#multilabelclassificationmodel, 2020. [Accessed 05-10-2023].

I. Triguero, S. García, and F. Herrera. Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study. *Knowledge and Information systems*, 42:245–284, 2015.

J. E. Van Engelen and H. H. Hoos. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440, 2020.

S. Verkijk and P. Vossen. Medroberta. nl: a language model for dutch electronic health records. *Computational Linguistics in the Netherlands Journal*, 11:141–159, 2021.