UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

NLP-Based Hospital Diagnosis Reporting Aid

Gwendoline Nijssen
Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :
Prof. dr. Frank NEVEN

COPROMOTOR :
dr. Brecht VANDEVOORT

2024
2025

De transnationale Universiteit Limburg is een uniek samenwerkingsverband van twee universiteiten in twee landen: de Universiteit Hasselt en Maastricht University.

UHASSELT
KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

NLP-Based Hospital Diagnosis Reporting Aid

Gwendoline Nijssen

Scriptie ingediend tot het behalen van de graad van master in de informatica

PROMOTOR :
Prof. dr. Frank NEVEN

COPROMOTOR :
dr. Brecht VANDEVOORT

Acknowledgement

I would like to express my sincere gratitude to all those who have supported and guided me throughout my academic journey and the completion of this Master's thesis. First and foremost, I would like to thank Prof. Dr. Frank Neven for his guidance, feedback, and supervision over the past year. His support has been essential in shaping both the direction and the quality of this research. I am also deeply grateful to Prof. Dr. Inigo Bermejo Delgado for sharing his expertise in Machine Learning and Deep Learning. His insights significantly contributed to this work. I also appreciate both his prof. Neven's assistance with the administrative procedures, particularly in obtaining the necessary ethical approvals and in facilitating access to the supercomputer clusters of the VSC, which made this study possible. My sincere thanks go to Brecht Vandevoort, whose prior experience was a great resource throughout my research. His support and his practical advice were greatly appreciated. A special thank you goes to Ruben Hoffman, who facilitated the collaboration with the Jessa Hospital. His help in securing ethical approvals and clarifying medical aspects of the research played a key role in the successful execution of this thesis. Last but certainly not least, I would like to thank my family and friends for their unwavering patience, understanding, and support throughout my studies. Your encouragement has been a constant source of motivation and have made this possible.

Abstract

Introduction

This thesis researches the use of Natural Language Processing (NLP) techniques to support hospital diagnosis reporting and quality of care assessment for patients with Atrial Fibrillation (AF), following guidelines set by the European Society of Cardiology (ESC). The ESC has established specific guidelines for AF patient care, including requirements for calculating stroke risk score, referred to as CHA_2DS_2–VASc scores, but monitoring adherence to these guidelines across thousands of patient records presents a substantial logistical challenge. This research assesses the feasibility of using various modern NLP approaches to automate hospital diagnosis reporting.

Research Design and Methodology

The study employed a two-phase retrospective design, first developing methodologies using the English MIMIC-IV dataset containing over 40,000 patient records, then applying these approaches to Dutch hospital discharge notes from Jessa Hospital comprising 12,516 records.

The initial phase utilized the MIMIC-IV dataset, a comprehensive collection of over 40,000 English patient records from Beth Israel Deaconess Medical Center in Boston. We created two specialized subsets: `MIMIC-IV-Ext-Cardio`, containing discharge notes exclusively from cardiology departments, and `MIMIC-IV-Ext-Diagnoses`, encompassing records from all available hospital departments. This approach allowed us to evaluate model performance both within the target domain (cardiology) and across broader clinical contexts. We extracted the `MIMIC-IV-Ext-Cardio` dataset by combining columns from both `MIMIV-IV` and `MIMIC-IV-Note` (Johnson et al., 2023a) (Johnson et al., 2024) (Johnson et al., 2023b) to retrieve useful discharge notes specific to the cardiology department, similar to our data in the Jessa Hospital which was specific to the cardiology department as well. This two-phase approach was chosen to accommodate the timeline of administrative and ethical approvals required for accessing the Jessa Hospital dataset.

The second phase involved collaboration with Jessa Hospital in Hasselt, Belgium, providing access to 12,516 Dutch cardiac discharge notes from patients admitted to the cardiology department between 2016 and 2023. This dataset presented unique challenges, including significant class imbalance (88% AF-negative versus 12% AF-positive cases) and the need to process Dutch medical terminology.

We explored various NLP techniques for two primary tasks: automated AF classification and extraction of CHA_2DS_2–VASc scores.

AF Classification

For AF classification, we implemented an XGBoost classifier combined with Term Frequency-Inverse Document Frequency (TF-IDF) vectorization. This approach treats each discharge note as a collection of weighted terms, where TF-IDF scores reflect both the frequency of terms within individual documents and their discriminative power across the entire corpus. XGBoost,

Task	Dataset	Acc.	F1 for AF	Macro F1	Weighted F1
AF Classification	MIMIC-IV-Ext-Cardio	95%	0.90	0.93	0.95
AF Classification	MIMIC- IV-Ext-No- Cardio	86%	0.74	0.82	0.86
AF Classification	Jessa-AF	93%	0.71	0.84	0.93
AF Classification	Jessa-AF (larger n-grams)	94%	0.78	0.87	0.95
CHA_2DS_2–VASc Extraction	Jessa-CHADSVASc-10	95%	-	0.82	0.95

Table 1: Results of performance for various models across different tasks and datasets

a gradient boosting algorithm, builds an ensemble of decision trees that sequentially correct errors made by previous models. Our pre-processing pipeline included text cleaning, stop-word removal, and morphological processing (lemmatization for English, stemming for Dutch). We employed stratified k-fold cross-validation with hyperparameter optimization using scikit-learn's HalvingRandomSearchCV (Pedregosa et al., 2011) to identify optimal model configurations. The balanced training approach maintained equal proportions of AF-positive and AF-negative samples during training while preserving realistic class distributions in test sets.

CHA_2DS_2–VASc Score Extraction

For CHA_2DS_2–VASc score extraction, we fine-tuned MedRoBERTa.nl, a Dutch medical language model pre-trained on approximately 2.4 million hospital notes from Amsterdam University Medical Centers (Verkijk and Vossen, 2021). This model employs the RoBERTa architecture (Liu et al., 2019) with 12 transformer layers, 768 hidden dimensions, and 12 attention heads, specifically optimized for Dutch clinical text with a vocabulary of 52,000 subword tokens.

Our fine-tuning approach added a linear classification head to map the 768-dimensional token representations to our target classes (scores 0-7, 'no score', and 'greater than 2'). We implemented a pre-processing strategy that prioritized clinically relevant sections of discharge notes, particularly the '*BESLUIT EN BESPREKING*' (Conclusion and Discussion) section, where approximately 80% of explicit CHA_2DS_2–VASc score mentions occurred.

Results

As can be seen in table 1, the XGBoost classifier demonstrated strong performance across multiple evaluation contexts. On the `MIMIC-IV-Ext-Cardio` dataset, the model achieved 95% overall accuracy with an AF-specific F1-score of 0.90, indicating robust performance within the target cardiology domain. However, when applied to the broader `MIMIC-IV-Ext-No-Cardio` dataset containing records from all hospital departments, performance declined to 86% accuracy with an AF F1-score of 0.74, highlighting the importance of domain-specific training data.

On the Dutch `Jessa-AF` dataset, the model achieved 93% overall accuracy but showed reduced AF-specific performance with an F1-score of 0.71. This performance gap between English and Dutch datasets can be attributed to several factors, including the smaller size of the Dutch training corpus, language-specific challenges in clinical terminology, and the presence of class imbalance that was more pronounced in the Jessa dataset.

The enhanced n-gram approach (incorporating unigrams, bigrams, and trigrams) showed meaningful improvements for Dutch text processing, increasing the AF F1-score from 0.71 to 0.78. This improvement demonstrates the importance of capturing contextual patterns, particularly negation phrases common in clinical documentation such as '*geen voorkamerfibrillatie*' (no atrial fibrillation) and '*uitgesloten*' (excluded).

The fine-tuned MedRoBERTa.nl model achieved good performance on the score extraction task, with 95% overall accuracy and 0.82 macro F1-score on the `Jessa-CHADSVASc-10` dataset. The model demonstrated particularly strong performance in identifying cases where no explicit CHA_2DS_2–VASc score was mentioned (F1-score: 0.967), which represents a crucial capability for quality assessment purposes. Exact score classes (0-7) achieved F1-scores consistently above 0.89, with several classes reaching near-perfect performance. The primary challenge emerged with inequality expressions (e.g. 'CHA2DS2-VASc > 2'), where limited training examples resulted in reduced performance. This limitation reflects the relatively small dataset size and the low occurrence of these expressions in Dutch clinical documentation practices.

Our analysis of ESC quality indicators revealed significant gaps in clinical documentation practices. Among 1,489 AF-positive cases in the Jessa dataset, only 44.5% had explicit CHA_2DS_2–VASc scores documented in their discharge notes, while 55.5% lacked any score mention. Among patients with documented scores, 26.7% had clinically significant scores (>2) requiring anticoagulation consideration according to ESC guidelines (Hindricks et al., 2020). This finding underscores the potential value of automated extraction tools for quality assessment, as manual review of nearly 1,500 cases would represent a substantial time investment for clinical staff. The documented score distribution showed realistic clinical patterns, with scores 2-4 being most common and representing 62.1% of all documented scores.

Discussion

This thesis demonstrates that both traditional machine learning and modern transformer approaches can achieve clinically relevant performance for automated clinical text processing. The XGBoost + TF-IDF combination proved particularly effective for binary classification tasks, achieving the hypothesized >90% accuracy threshold while maintaining a level over interpretability through feature importance analysis. The success of this relatively simple approach aligns with recent findings suggesting that traditional machine learning methods can remain competitive with more sophisticated deep learning techniques for well-defined classification tasks (Falter et al., 2024).

The transformer-based approach showed good performance for the more complex score extraction task, indicating the importance of domain-specific pre-training. No baseline was established using non-domain specific language models, but the feasibility of our approach is proven and provides useful insight for further research. The 95% accuracy achieved by the fine-tuned MedRoBERTa.nl model substantially exceeded our 85% accuracy hypothesis, demonstrating the effectiveness of language-specific and domain-specific model development. However, in clinical context it is more important to evaluate results using F1-score, to minimize both false postives and false negatives. Given a macro F1-score of 0.82 for the CHA_2DS_2–VASc extraction model, we can consider the class imbalance and provide a more weighted metric by looking at the weighted F1-score (0.95 in this case).

Feasibility of Automated QI Analysis

The quality indicator analysis provides concrete evidence of documentation gaps that could be addressed by using automated NLP techniques. The finding that fewer than half of AF patients had documented CHA_2DS_2-VASc scores suggests opportunities for quality improvement using models similar to the ones discussed in this thesis. The models' ability to provide confidence scores for their predictions creates opportunities for human-in-the-loop systems, where uncertain cases can be flagged for manual review while high-confidence predictions proceed automatically. This approach could significantly reduce the time required for quality assessment while maintaining clinical oversight for ambiguous cases.

Limitations

Several important limitations emerged during the study. The 512-token sequence length limitation inherent to BERT-based architectures presented challenges for longer clinical documents, though our pre-processing approach of prioritizing relevant sections proved effective for most cases. A sliding-window approach should be researched and tested to see if improvements can be made to existing models.

The class imbalance in the Jessa dataset, while clinically realistic, required careful consideration in model training and evaluation. Language-specific challenges proved more significant than initially anticipated, with many pre-trained clinical embeddings being trained on English text, requiring a different pre-processing approach. Despite using enhanced n-gram approaches and domain-specific pre-processing, the performance gap between English and Dutch datasets suggests there is room for improvement. The scarcity of high-quality Dutch medical language models limited our options, though MedRoBERTa.nl proved to be an excellent choice for our extraction task.

The relatively small size of the Jessa dataset, particularly for rare score categories in the CHA_2DS_2-VASc extraction task, limited our ability to train robust models for all classes.

Future Research

This work establishes baseline performance metrics that can guide future research in several directions. Extending the binary AF classification to multi-label classification for specific AF types (e.g. paroxysmal, persistent, permanent) would provide more clinically relevant information and closer alignment with ICD-10 coding and provide a larger scope for automated reporting.

The development of more sophisticated embedding approaches that better capture medical semantics while maintaining the benefits of gradient boosting represents an interesting technical challenge. Our FastText experiments showed limited success, but more advanced approaches might yield better results. For the score extraction task, developing models capable of inferring CHA_2DS_2-VAScscores from risk factor mentions rather than explicit score statements would provide great insight into domain-specific pre-trained model capabilities for Dutch medical text.

Finally, extending the quality indicator analysis to include anticoagulant therapy assessment would complete the picture of ESC guideline adherence and provide a more comprehensive quality monitoring system than the one tested in this thesis.

Conclusion

The research in this thesis successfully demonstrates the feasibility of using modern NLP techniques to support hospital diagnosis reporting and quality assessment for AF patients. The combination of traditional machine learning for classification tasks and transformer-based models for extraction tasks achieved good performance (compared to manual labeling) while highlighting important considerations for practical implementation.

The establishment of baseline performance metrics, development of domain-specific pre-processing pipelines, and demonstration of real-world applicability provide a foundation for future research and potential implementation in automated reporting tools. While challenges remain, particularly in cross-language text processing, handling of rare classes and large class imbalances, the results suggest that automated quality monitoring tools can provide valuable support for healthcare quality improvement initiatives.

Samenvatting

Inleiding

Dit proefschrift onderzoekt het gebruik van Natural Language Processing (NLP) technieken ter ondersteuning van ziekenhuisdiagnoserapportage en kwaliteit van zorg voor patiënten met Voorkamerfibrillatie (VKF), volgens richtlijnen vastgesteld door de European Society of Cardiology (ESC). De ESC heeft specifieke richtlijnen opgesteld voor AF patiëntenzorg, inclusief vereisten voor het berekenen van een beroerterisicoscore, aangeduid als CHA_2DS_2–VASc scores, maar het monitoren van naleving van deze richtlijnen over duizenden patiëntendossiers vormt een aanzienlijke logistieke uitdaging. Dit onderzoek beoordeelt de haalbaarheid van het gebruik van verschillende moderne NLP toepassingen om ziekenhuisdiagnoserapportage te automatiseren.

Onderzoeksontwerp en Methodologie

De studie hanteerde een tweefasig retrospectief ontwerp, waarbij eerst methodologieën werden ontwikkeld met behulp van de Engelse MIMIC-IV dataset die meer dan 40.000 ontslagdocumenten bevat. Vervolgens werden deze bevindingen toegepast op Nederlandstalige ziekenhuisontslagbrieven van het Jessa Ziekenhuis bestaande uit 12.516 ontslagbrieven. De eerste fase gebruikte de MIMIC-IV dataset, een uitgebreide verzameling van meer dan 40.000 Engelse patiëntendossiers van Beth Israel Deaconess Medical Center in Boston (Johnson et al., 2023a) (Johnson et al., 2024) (Johnson et al., 2023b). We creëerden twee gespecialiseerde subsets: MIMIC-IV-Ext-Cardio, die ontslagbrieven uitsluitend van de afdeling cardiologie bevatte, en MIMIC-IV-Ext-Diagnoses, die onstlagbrieven bevatte van alle andere beschikbare ziekenhuisafdelingen. Deze benadering stelde ons in staat om modelprestaties te evalueren zowel binnen het doeldomein (cardiologie) als over bredere klinische contexten. We stelde de MIMIC-IV-Ext-Cardio dataset samen door kolommen te combineren van zowel MIMIC-IV als MIMIC-IV-Note (Johnson et al., 2023a) (Johnson et al., 2024) (Johnson et al., 2023b) om nuttige ontslagbrieven specifiek voor de cardiologie afdeling op te halen, vergelijkbaar met onze data in het Jessa Ziekenhuis die ook enkel documenten uit de cardiologie afdeling bevatte. Deze tweefasige benadering werd gekozen om rekening te houden met de tijdlijn van administratieve en ethische goedkeuringen vereist voor toegang tot de Jessa Ziekenhuis dataset.

De tweede fase omvatte een samenwerking met Jessa Ziekenhuis in Hasselt, België, dat toegang verschafte tot 12.516 Nederlandstalige ontslagbrieven van patiënten opgenomen op de afdeling cardiologie tussen 2016 en 2023. Deze dataset presenteerde unieke uitdagingen, inclusief significante imbalans van klasses (88% AF-negatieve versus 12% AF-positieve gevallen) en de noodzaak om Nederlandstalige medische terminologie te verwerken. We onderzochten verschillende NLP technieken voor twee primaire taken: geautomatiseerde AF classificatie en extractie van CHA_2DS_2–VASc scores.

AF Classificatie

Voor AF classificatie implementeerden we een XGBoost classifier gecombineerd met Term Frequency-Inverse Document Frequency (TF-IDF) vectorisatie. Deze benadering behandelt

Taak	Dataset	Acc.	F1 voor AF	Macro F1	Gewogen F1
AF Classificatie	MIMIC-IV-Ext-Cardio	95%	0.90	0.93	0.95
AF Classificatie	MIMIC- IV-Ext-No- Cardio	86%	0.74	0.82	0.86
AF Classificatie	Jessa-AF	93%	0.71	0.84	0.93
AF Classificatie	Jessa-AF (larger n-grams)	94%	0.78	0.87	0.95
CHA_2DS_2–VASc Extractie	Jessa-CHADSVASc-10	95%	-	0.82	0.95

Table 2: Resultaten van de prestatie van de modellen over de verschillende taken en datasets heen

elke ontslagbrief als een verzameling gewogen termen, waarbij TF-IDF scores zowel de frequentie van termen binnen individuele documenten als hun onderscheidend vermogen over het gehele corpus weergeven. XGBoost, een gradient boosting algoritme, bouwt een samenvoegsel van beslissingsbomen die opeenvolgend fouten corrigeren die gemaakt werden door vorige modellen. Onze pre-processing pipeline omvatte data cleaning, verwijdering van stopwoorden, en morfologische verwerking (lemmatization voor Engels, stemming voor Nederlands). We gebruikten gestratificeerde k-fold cross-validation met hyperparameter optimalisatie met behulp van scikit-learn's HalvingRandomSearchCV (Pedregosa et al., 2011) om optimale modelconfiguraties te identificeren. De gebalanceerde trainingsaanpak handhaafde gelijke verhoudingen van AF-positieve en AF-negatieve samples tijdens training terwijl realistische klasseverdelingen in testsets behouden bleven.

CHA_2DS_2–VASc Score Extractie

Voor CHA_2DS_2–VASc score extractie verfijnden we MedRoBERTa.nl, een Nederlands taalmodel voorgetraind op ongeveer 2,4 miljoen ziekenhuisbrieven van Amsterdam University Medical Centers (Verkijk and Vossen, 2021). Dit model gebruikt de RoBERTa architectuur (Liu et al., 2019) met 12 transformer lagen, 768 verborgen dimensies, en 12 attention heads, specifiek geoptimaliseerd voor Nederlandstalige klinische tekst met een vocabulaire van 52.000 subword tokens. Onze fine-tuning benadering voegde een lineaire classificatiekop toe om de 768-dimensionale token representaties te mappen naar onze doelklassen (scores 0-7, 'geen score', en 'groter dan 2'). We implementeerden een pre-processing strategie die prioriteit gaf aan klinisch relevante secties van ontslagbrieven, met name de 'BESLUIT EN BESPREKING' sectie, waar ongeveer 80% van expliciete CHA_2DS_2–VASc score vermeldingen voorkwamen.

Resultaten

Zoals te zien is in 2, toonde de XGBoost classifier sterke prestaties over meerdere evaluatiecontexten. Op de MIMIC-IV-Ext-Cardio dataset behaalde het model 95% algehele nauwkeurigheid (accuracy) met een AF-specifieke F1-score van 0.90, wat robuuste prestaties binnen het doeldomein aangeeft. Echter, wanneer toegepast op de bredere MIMIC-IV-Ext-No-Cardio dataset die dossiers van alle ziekenhuisafdelingen bevatte, daalde de prestatie naar 86% nauwkeurigheid (accuracy) met een AF F1-score van 0.74, wat het belang van domein-specifieke trainingsdata benadrukt. Op de Nederlandse Jessa-AF dataset behaalde het model 93% algehele nauwkeurigheid (accuracy) maar toonde verminderde AF-specifieke prestatie met een F1-score van 0.71. Deze afstand tussen prestasties van Engelse en Nederlandse datasets kan worden toegeschreven aan verschillende factoren, inclusief de kleinere omvang van het Nederlandstalige trainingscorpus, taalspecifieke uitdagingen in klinische terminologie, en de aanwezigheid van klasse-onbalans die meer uitgesproken was in de Jessa dataset. De verbeterde n-gram benadering (met inbegrip van unigrammen, bigrammen, en trigrammen) toonde betekenisvolle verbeteringen voor Nederlandse tekstverwerking, waarbij de AF F1-score steeg van 0.71 naar 0.78 door het toevoegen

van bi- and trigrams in onze aanpak. Deze verbetering toont het belang aan van het vastleggen van contextuele patronen, met name ontkenningen en negaties die gebruikelijk zijn in klinische documentatie zoals 'geen voorkamerfibrillatie' en 'uitgesloten'.

Het verfijnde MedRoBERTa.nl model behaalde goede prestatie op de score extractietaak, met 95% algehele nauwkeurigheid en 0.82 macro F1-score op de Jessa-CHADSVASc-10 dataset. Het model toonde bijzonder sterke prestatie in het identificeren van gevallen waar geen expliciete CHA_2DS_2–VASc score werd vermeld (F1-score: 0.967), wat een cruciale mogelijkheid vertegenwoordigt voor kwaliteitsbeoordelingsdoeleinden. Exacte scoreklassen (0-7) behaalden F1-scores consequent boven 0.89, met verschillende klassen die bijna perfecte prestatie bereikten. De primaire uitdaging ontstond bij ongelijkheidsuitdrukkingen (bijv., 'CHA2DS2-VASc > 2'), waar beperkte trainingsvoorbeelden resulteerden in verminderde prestatie. Deze beperking weerspiegelt de relatief kleine omvang van de dataset en het lage voorkomen van deze uitdrukkingen in Nederlandstalige klinische documentatiepraktijken. Onze analyse van ESC kwaliteitsindicatoren onthulde significante gaten in klinische documentatiepraktijken. Onder de 1.489 AF-positieve gevallen in de Jessa dataset hadden slechts 44,5% expliciete CHA_2DS_2–VASc scores gedocumenteerd in hun ontslagbrieven, terwijl 55,5% elke scorevermelding miste. Onder patiënten met gedocumenteerde scores hadden 26,7% klinisch significante scores (>2) die anticoagulatie overweging vereisten volgens ESC richtlijnen (Hindricks et al., 2020). Deze bevinding onderstreept de potentiële waarde van geautomatiseerde extractietools voor kwaliteitsbeoordeling, aangezien handmatige beoordeling van bijna 1.500 gevallen een aanzienlijke tijdsinvestering voor klinisch personeel zou vertegenwoordigen. De gedocumenteerde scoreverdeling toonde realistische klinische patronen, met scores 2-4 die het meest gebruikelijk waren en 62,1% van alle gedocumenteerde scores vertegenwoordigden.

Discussie

Dit proefschrift toont aan dat zowel traditionele machine learning als moderne transformer benaderingen klinisch relevante prestaties kunnen behalen voor geautomatiseerde klinische tekstverwerking. De XGBoost + TF-IDF combinatie bleek bijzonder effectief voor binaire classificatietaken, waarbij de gehypothetiseerde >90% nauwkeurigheidsdrempel werd behaald terwijl een niveau van interpreteerbaarheid werd gehandhaafd door feature importance analyse. Het succes van deze relatief eenvoudige benadering sluit aan bij recente bevindingen die suggereren dat traditionele machine learning methoden competitief kunnen blijven met meer geavanceerde deep learning technieken voor goed gedefinieerde classificatietaken (Falter et al., 2024).

De transformer-gebaseerde benadering toonde goede prestatie voor de meer complexe score extractietaak, wat het belang van domein-specifieke pre-training aangeeft. Geen baseline werd vastgesteld met behulp van niet-domein specifieke taalmodellen, maar de haalbaarheid van onze benadering is bewezen en biedt nuttig inzicht voor verder onderzoek. De 95% nauwkeurigheid behaald door het verfijnde MedRoBERTa.nl model overtrof onze 85% nauwkeurigheidshypothese aanzienlijk, wat de effectiviteit van taalspecifieke en domein-specifieke modelontwikkeling toont. Echter, in klinische context is het belangrijker om resultaten te evalueren met behulp van F1-score, om zowel valse positieven als valse negatieven te minimaliseren. Gegeven een macro F1-score van 0.82 voor het CHA_2DS_2–VASc extractiemodel, kunnen we de klasseonbalans overwegen en een meer gewogen metriek bieden door te kijken naar de gewogen F1-score (0.95 in dit geval).

Haalbaarheid van Geautomatiseerde QI Analyse

De QI analyse biedt concreet bewijs van gaten in de rapportering die kunnen worden aangepakt door geautomatiseerde NLP technieken te gebruiken. De bevinding dat minder dan de helft van AF patiënten gedocumenteerde CHA_2DS_2–VASc scores hadden, suggereert mogelijkheden voor kwaliteitsverbetering met behulp van modellen vergelijkbaar met degenen besproken in dit proefschrift. De mogelijkheid van de modellen om betrouwbaarheidsscores voor hun voorspellingen te bieden creëert mogelijkheden voor human-in-the-loop systemen, waar onzekere gevallen

kunnen worden gemarkeerd voor handmatige beoordeling terwijl hoge-betrouwbaarheid voorspellingen automatisch kunnen doorgaan. Deze benadering zou de tijd vereist voor kwaliteitsbeoordeling aanzienlijk kunnen verminderen, terwijl klinisch toezicht voor ambigue gevallen wordt gehandhaafd.

Beperkingen

Verschillende belangrijke beperkingen kwamen naar voren tijdens de studie. De 512-token sequentielengte inherent aan BERT-gebaseerde architecturen presenteerde uitdagingen voor langere klinische documenten, hoewel onze pre-processing benadering, die prioriteit gaf aan relevante secties, effectief bleek voor de meeste gevallen. Een sliding-window benadering zou moeten worden onderzocht en getest om te zien of verbeteringen kunnen worden gemaakt aan bestaande modellen. De klasse-onbalans in de Jessa dataset, hoewel klinisch realistisch, vereiste zorgvuldige overweging in modeltraining en evaluatie. Taalspecifieke uitdagingen bleken significanter dan aanvankelijk geanticipeerd, met veel voorgetrainde klinische embeddings getraind op Engelse tekst, wat een andere pre-processing benadering vereiste. Ondanks het gebruik van verbeterde n-gram benaderingen en domein-specifieke pre-processing, suggereert het verschil in prestatie tussen Engelstalige en Nederlandstalige datasets dat er ruimte voor verbetering is. De schaarste van hoogkwalitatieve Nederlandse medische taalmodellen beperkte onze opties, hoewel MedRoBERTa.nl een uitstekende keuze bleek voor onze extractietaak. De relatief kleine omvang van de Jessa dataset, met name voor zeldzame scorecategorieën in de CHA_2DS_2–VASc extractietaak, beperkte ons vermogen om robuuste modellen voor alle klassen te trainen.

Toekomstig Onderzoek

Dit werk stelt baseline prestatiemetrieken vast die toekomstig onderzoek in verschillende richtingen kunnen begeleiden. Uitbreiding van de binaire AF classificatie naar multi-label classificatie voor specifieke AF types (bv. paroxysmaal, persistent, permanent) zou meer klinisch relevante informatie bieden en nauwere afstemming met ICD-10 codering en een grotere reikwijdte voor geautomatiseerde rapportage bieden.

De ontwikkeling van meer geavanceerde embeddingbenaderingen die medische semantiek beter vastleggen terwijl de voordelen van gradient boosting behouden blijven vertegenwoordigt een interessante technische uitdaging. Onze FastText-experimenten toonden beperkt succes, maar meer geavanceerde benaderingen zouden betere resultaten kunnen opleveren. Voor de score extractietaak zou het ontwikkelen van modellen die in staat zijn CHA_2DS_2–VASc scores af te leiden uit vermeldingen van risicofacoren in plaats van expliciete score verklaringen, groot inzicht bieden in mogelijkheden voor het verwerken van Nederlandse medische tekst met voorgetrainde modellen. Ten slotte zou uitbreiding van de kwaliteitsindicator (QI) analyse, zoals het nagaan van toediening van bloedverdunnende medicatie in het geval val een significante CHA_2DS_2–VASc een meer uitgebreid kwaliteitsbewakingssysteem bieden dan degene getest in dit proefschrift.

Conclusie

Het onderzoek in dit proefschrift toont succesvol de haalbaarheid aan van het gebruik van moderne NLP technieken ter ondersteuning van ziekenhuisdiagnoserapportage en kwaliteitsbeoordeling voor AF patiënten. De combinatie van traditionele machine learning voor classificatietaken en transformer-gebaseerde modellen voor extractietaken behaalde goede prestaties (vergeleken met handmatige labeling) terwijl belangrijke overwegingen voor praktische implementatie werden benadrukt. De vaststelling van baseline prestatiemetrieken, ontwikkeling van domein-specifieke pre-processing pipelines, en demonstratie van echte-wereld toepasbaarheid bieden een fundament voor toekomstig onderzoek en potentiële implementatie in geautomatiseerde rapportagetools. Hoewel uitdagingen blijven bestaan, met name in tekstverwerking met Nederlanse tekst, behandeling van zeldzame klassen en grote klasse-onbalansen, sugger-

eren de resultaten dat geautomatiseerde kwaliteitsbewakingstools waardevolle ondersteuning kunnen bieden voor het rapporteren van kwaliteitsindicatoren voor patiënten met AF.

Contents

Glossary

AF Atrial Fibrillation. 15–19, 21–24, 27, 30, 31, 33, 34, 36, 38–41, 55–60, 62, 63, 65–67, 70–73

BERT Bidirectional Encoder Representations from Transformers. 44, 45, 51, 68

CHA_2DS_2–VASc Is a score calculated to measure a patient's CVA or stroke risk. 2–4, 16–19, 21, 23–28, 42, 43, 45, 46, 49, 51, 53, 55–57, 61–65, 67–73

CNN Convolutional Neural Network. 43, 53

CVA cerebrovascular accidents. 14, 16

ESC European Society of Cardiology. 17, 18, 62, 63, 71–73

ICD-10 10th Revision of the International Statistical Classification of Diseases and Related Health Problems. 15, 17, 22

LLM Large Language Model. 18, 68, 72, 73

MLM Masked Language Modeling. 45, 51

MZG Minimum Hospital Data Set. 15, 17

NLP Natural Language Processing. 2, 4, 18, 29, 43, 61, 68, 71–73

OOV Out-of-Vocabulary. 34, 35, 67

RNN Recurrent Neural Network. 43, 53

RoBERTa Robustly optimized BERT approach. 43–45, 68

TF-IDF Term Frequency-Inverse Document Frequency. 18, 21, 28, 33–38, 41, 57, 66, 67

VSC Flemish Supercomputer Center (Vlaams Supercomputer Centrum). 31, 32

WHO World Health Organization. 15

Chapter 1

Introduction

1.1 Background

1.1.1 Minimum Hospital Data Set (MZG)

Hospitals in Belgium are required to register various types of data, classified under the Minimum Hospital Data Set (*Minimale Ziekenhuisgegevens*, MZG). The MZG data often contains reports and statistics that need to be reported to the government for further analysis, but are also used to enforce general quality guidelines.

One of those requirements includes correctly coding, i.e. mapping, diagnoses and treatments to a relevant and universally understandable code. The World Health Organization (WHO) defines the 10th Revision of the International Statistical Classification of Diseases and Related Health Problems (ICD-10) to facilitate the communication of health problems in various types of records, such as hospital records and death records (World Health Organisation, 2019). Belgium enforces the use of the ICD-10-CM coding system, which is widely used to correctly communicate and identify medical information. In this thesis, the term ICD-10 will be used to refer to ICD-10-CM. However, it is important to note that certain countries may have slight variations in their coding systems.

1.1.2 Hospital Discharge Notes

When patients are admitted to and later discharged from a hospital, a discharge note will be created. Discharge notes are official medical documents that contain information about a patient's hospital stay, diagnoses, treatments, and crucial data on patient quality of care.

These documents often follow a structured format, with different sections for the patient's personal information, medical history, diagnoses, treatments, procedures that were performed during the hospital stay, medication, and additional medical notes.

Hospital extract certain metrics and data from discharge notes, and will further analyze and structure them for the MZG reporting (FOD Volksgezondheid, Veiligheid van de Voedselketen en Leefmilieu, 2016).

1.1.3 Atrial Fibrillation

In this thesis, we exclusively focused on patients with the condition known as Atrial Fibrillation (AF). AF is the most common arrhythmia or heart rhythm disorder, in which the two upper chambers of the heart, called the atriums, contract too quickly and irregularly.

In this thesis, Atrial Fibrillation (AF) is used as an umbrella term for the different types of Atrial Fibrillation, as well as Atrial Flutter. Atrial Fibrillation is classified into several types,

Risk factor	Description	Score
Congestive heart failure	Higher risk of stroke for patients with congestive HF, including patients with Clinical HF, or those who have objective evidence of moderate to severe Left Ventricular dysfunction, or Hypertrophic Cardiomyopathy (also affecting the left ventricle of the heart).	1
Hypertension	History of hypertension, including patients on antihypertensive therapy, as this may indicate higher risk for stroke.	1
Age $\geq$ 75 years	Age is a powerful driver of stroke risk. Age-related risk is a continuum, but for reasons of simplicity and practicality, 1 point is given for age 65 - 74 years and 2 points for age $\geq$75 years.	2
Diabetes mellitus	Diabetes mellitus is a well-established risk factor for stroke.	1
Stroke	Previous stroke, systemic embolism, or TIA confers a particularly high risk of ischemic stroke, hence weighted 2 points.	2
Vascular disease	Vascular disease (PAD or myocardial infarction), Angiographically significant CAD, and complex aortic plaque on the descending aorta as an indicator of significant vascular disease, are all predictors of or risk factors for ischemic stroke.	1
Age 65–74 years	See above.	1
Sex **c**ategory (female)	Stroke risk is higher in females.	1
	Maximum total score	9

Table 1.1: Risk factors for computation of CHA_2DS_2–VASc (Hindricks et al., 2020)

based on the duration of episodes a patient experiences. These types include paroxysmal, persistent, long-term, and permanent Atrial Fibrillation. Atrial flutter is also classified as an arrhythmia, caused by the atriums contracting more quickly than expected. While Atrial Fibrillation is defined to be more chaotic than Atrial Flutter, both arrhythmias have similar symptoms, causes, and treatment options.

Discharge notes for patients with AF, whether AF was a pre-existing condition or newly diagnosed during hospitalization, typically contain more information about the diagnosis and the risks involved for the patient, as well as treatment decisions made during the hospital stay. For example, patients are often given anticoagulation therapy, as untreated, AF can increase the risk of cerebrovascular accidents (CVA), also known as strokes.

The documentation of patient care during their hospital stay is certainly important, and it facilitates the assessment of the adherence to specific clinical guidelines and quality of care standards.

1.1.4 CHA_2DS_2–VASc Score

To determine the risk of stroke for patients with AF, a CHA_2DS_2–VASc score is often calculated. The criteria, or risk factors, used to determine the score are presented in table 1.1.

Many patients who are admitted to a hospital with a history of AF or a new diagnosis of AF will have a CHA_2DS_2–VASc score calculated for them, which can be found in their discharge

Type	ICD-10 code
Paroxysmal Atrial Fibrillation	I48.0
Persistent Atrial Fibrillation	I48.1x
Chronic Atrial Fibrillation	I48.2x
Typical Atrial Flutter	I48.3
Atypical Atrial Flutter	I48.4
Unspecified Atrial Fibrillation	I48.91
Unspecified Atrial Flutter	I48.92

Table 1.2: The types of AF with their corresponding ICD-10 code

note. The CHA_2DS_2–VASc score is a good predictive value for the risk of stroke for patients diagnoses with AF (Gažová et al., 2019). Based on the resulting score of the patient, various treatment options are considered. For instance, patients with a CHA_2DS_2–VASc score of two or more are expected to receive anticoagulant therapy following the quality indicators for patients with AF as decided by the European Society of Cardiology (ESC).

1.2 Research Objectives

1.2.1 Quality Indicators for Atrial Fibrillation

One of the research objectives of this thesis is to calculate the percentage of adherence to current quality indicators for patients diagnosed with Atrial Fibrillation (AF), as determined by the ESC guidelines.

We have chosen the following quality indicators in particular to focus on in this thesis:

- Proportion of AF patients with a reported CHA_2DS_2–VASc score
- Proportion of AF patients with a significant CHA_2DS_2–VASc score (two or higher) that received anticoagulant therapy

Given these research objectives, we need a way to correctly identify patients with AF given a set of discharge notes.

1.2.2 Automated ICD-10 Coding

One of the components of MZG data is the ICD-coding of patient diagnoses through discharge letters. For example, when a patient is admitted to a hospital and a diagnosis for 'Atrial fibrillation (Unspecified)' is established, the hospital has to attach the ICD-10 code for the diagnosis to the discharge note, which in this case would be the code I48.91.

However, extracting MZG data is still often done manually, and hospitals hire dedicated coders whose job is to correctly identify and code records for the hospital. Records often have to be re-evaluated due to human error. The government will only allow for a certain number of revisions before a hospital is fined, thus making mistakes costly. This is where automated ICD-10 can introduce a great improvement in the efficiency and accuracy of ICD-10 coding.

Our second primary research objective focuses on correctly identifying patients with Atrial Fibrillation (AF). To determine the proportion of patients with AF, we first need to classify the reports into two classes: those of patients with AF, and those of patients without AF.

Diagnoses for any type of AF in hospital discharge notes will later be linked to one of the ICD-10 codes, found in table 1.2. It is therefore useful to correctly identify patients with AF through the correct classification of the ICD-10 codes for Atrial Fibrillation (AF).

1.3 Research Questions and Hypotheses

1.3.1 Primary Research Questions

We define the following primary research questions:

1. Can modern Natural Language Processing (NLP) techniques help Belgian hospitals to measure and survey ESC quality indicators through discharge letters, specifically for patients with Atrial Fibrillation (AF)? (RQ1)
2. Can modern Natural Language Processing (NLP) techniques help in automating the identification of patients with Atrial Fibrillation (AF) through discharge notes? (RQ2)
3. Can modern Natural Language Processing (NLP) techniques, specifically Large Language Model (LLM)s, provide help in extracting relevant CHA_2DS_2–VASc scores from discharge notes? (RQ3)

1.3.2 Primary Hypotheses

H1: Atrial Fibrillation Classification
Machine learning models using Term Frequency-Inverse Document Frequency (TF-IDF) vectorization combined with XGBoost can achieve clinically acceptable accuracy (>90%) for automated identification of atrial fibrillation cases from hospital discharge notes, both in English (MIMIC-IV) and Dutch (Jessa Hospital) datasets.

This hypotheses is primarily based off of results from previous studies, indicating a baseline approach using XGBoost in combination with TF-IDF can achieve great performance out of the box. The study by Falter et al. (2024) demonstrated that XGBoost applied to TF-IDF matrices achieved an accuracy of 94% for atrial fibrillation classification on the MIMIC-III dataset. Importantly, this traditional machine learning approach outperformed more sophisticated deep learning techniques, including BioBERT (accuracy: 84%), suggesting that for clinical text classification tasks, the interpretability and robustness of XGBoost may be more valuable than the theoretical sophistication of transformer-based models (Falter et al., 2024).

The superiority of XGBoost over deep learning approaches in this domain can be attributed to several factors. Clinical discharge notes often contain domain-specific vocabulary and abbreviations that may not be well-represented in the clinical corpora used to train models like BioBERT or ClinicalBERT. The 90% accuracy threshold specified in this hypothesis is based off of the previous study by Falter et al. (2024)

However, systematic reviews reveal significant limitations in current evidence for performance of automated ICD-10 coding or similar applications. The comprehensive systematic review by Kaur et al. (2023) of 42 studies (2010-2021) following PRISMA guidelines found that top-performing models achieved only micro-F1 scores of 0.477-0.551 on MIMIC datasets, noting that 'efforts are still required to improve ICD code prediction accuracy' and highlighting the need for large-scale clinical corpora.

H2: CHA_2DS_2–VASc Score Extraction
Fine-tuned transformer models, specifically MedRoBERTa.nl trained on Dutch medical text, can accurately extract CHA_2DS_2–VASc scores from Dutch hospital discharge notes with >85% accuracy, enabling automated quality indicator assessment.

Given our approach using linear classification on top of a pre-trained model, we hypothesize a model with similar or higher accuracy (85%) than the binary classification model mentioned in Falter et al. (2024) could be possible for our task.

H3: Quality Indicator Assessment
Modern NLP techniques can successfully automate the measurement of ESC quality indicators for atrial fibrillation patients, specifically:

- The proportion of AF patients with reported CHA_2DS_2–VASc scores
- The proportion of AF patients with clinically significant scores ($\geq$2)

1.3.3 Secondary Hypotheses

H4: Cross-domain Generalization
Atrial Fibrillation (AF) classification models trained on cardiology department data will show reduced performance when applied to general hospital discharge notes from other departments, highlighting the importance of domain-specific training.

H5: Language-specific Models Domain-specific pre-trained models (MedRoBERTa.nl for Dutch medical text) will outperform general language models for clinical information extraction tasks in Dutch medical text.

Chapter 2

Methodology

In this chapter, we discuss the various methodologies we used to explore and create solutions for our research questions.

2.1 Research Design

The thesis was structured as a retrospective study and was split into two phases to ensure continuous research progress. The first phase focused on exploring and developing the methodologies using a public dataset of hospital discharge notes. We chose to use the MIMIC-IV dataset for this, which we will discuss in more detail later. The second phase consisted of applying the previously researched methods to a dataset of Dutch hospital discharge notes from the Jessa Hospital. This two-phase approach was mostly chosen to accommodate pending administrative and ethical approvals required for accessing the Jessa Hospital dataset.

Using the `MIMIC-IV-Ext-Cardio` dataset, we researched and developed our methodology in the first phase of this thesis. We implemented and compared multiple approaches for the automated classification of Atrial Fibrillation (AF), including traditional machine learning with Term Frequency-Inverse Document Frequency (TF-IDF) features, embedding-based approaches with FastText, and defined baseline performance benchmarks. This phase allowed us to try out various ideas and hyperparameter configurations without the constraints of limited availability of data.

The second phase consisted mostly of the application of the previously researched methods for AF classification, while diving further into methods for CHA_2DS_2–VASc score extraction. We applied our best-performing `MIMIC-IV-Ext-Cardio` methodologies to the `Jessa-AF` dataset of Dutch hospital discharge notes, then researched and implemented a CHA_2DS_2–VASc score extraction pipeline using a medical large language model, and evaluated the performance of our models on the different predefined and excluded subsets of the original `Jessa-AF` dataset. Throughout this chapter, we will refer to these experiments according to the dataset that was used.

2.2 Datasets

2.2.1 MIMIC-IV Dataset

To explore various methodologies for processing and analyzing hospital discharge notes while we waited on approval to start phase two of the thesis, we utilized the MIMIC-IV dataset (Johnson et al., 2024), created by Johnson et al. (2023b). The dataset was accessed through PhysioNet (Goldberger et al., 2002). In accordance with the dataset's access requirements, the CITI Program course on Human Research for Data or Specimens Only Research was completed before

retrieving and analyzing the data. In accordance with the dataset's access requirements, the CITI Program course on Human Research for Data or Specimens Only Research was completed before retrieving and analyzing the data. This course provided guidelines and regulations around working with human and medical data in a research context. It assisted in keeping us informed about potential breaches of regulations and provided instructions on what to do in the event that any data was stolen or leaked.

We combined the original MIMIC-IV dataset with the MIMIC-IV-Note dataset (Johnson et al., 2023a), which was retrieved through PhysioNet (Goldberger et al., 2002) as well.

MIMIC-IV (Medical Information Mart for Intensive Care, version IV) is a large, freely available database comprising disidentified health data associated with over 40,000 patients admitted to the Beth Israel Deaconess Medical Center (BIDMC) in Boston, Massachusetts between 2008 and 2019 (Johnson et al., 2024). A wide range of clinical data is included in the dataset, such as laboratory test results, medication information, and, most importantly for our study, diagnoses in the form of ICD-10 codes (Johnson et al., 2024). The MIMIC-IV-Note dataset forms an extension to the basic MIMIC-IV dataset, with clinical notes that include discharge summaries.

Different subsets were extracted from the original MIMIC-IV dataset and combined with the MIMIC-IV-Note dataset. Throughout this thesis, we refer to two of our extracted subsets as

- `MIMIC-IV-Ext-Diagnoses`, and
- `MIMIC-IV-Ext-Cardio`

following the recommended citation guidelines for derivative works as outlined by Physionet (Johnson et al., 2024).

`MIMIC-IV-Ext-Diagnoses` contains discharge notes retrieved from the original MIMIC-IV-Note dataset hospitalizations. This means we did not make any extra effort to specifically include ICU stays, which MIMIC-IV also provides data for, as the regular hospitalizations are more similar to the data we would be receiving in phase two. We then annotated the dataset with a label for Atrial Fibrillation (AF), indicating which discharge notes were related to patients with (1) or without (0) AF. Whether a patient was (previously) diagnosed with AF was determined by the ICD-10 codes that were linked to each hospital discharge note, which could be found in the original MIMIC-IV dataset. Discharge notes were the only type of note that were included in the final dataset.

`MIMIC-IV-Ext-Cardio` contains discharge notes for any patient who was both present in MIMIC-IV and MIMIC-IV-Note, similar to the previous subset. This dataset of discharge notes was also annotated with a label for Atrial Fibrillation (AF), indicating which discharge notes were related to patients with or without AF. However, for this dataset, we only included records that were related to patients that were admitted to the Cardiology department. This information was obtained from the specified services that were linked to each hospitalization. More specifically, only records tagged with 'CMED' (Cardiac Medical - for non-surgical cardiac related admissions) and 'CSURG' (Cardiac Surgery - for surgical cardiac admissions) were included Johnson et al. (2024).

2.2.2 Jessa Hospital Dataset

The dataset we received from the Jessa Hospital in Hasselt, Belgium, contained hospital discharge notes for a pre-determined set of patients. The population of the dataset included all cardiac patients (patients admitted to the cardiology department) with hospitalization between 01/01/2016 and 31/12/2023, with an available hospitalization report, and over the age of 18. This original dataset had already been annotated with a binary label indicating the presence or lack of an Atrial Fibrillation (AF) diagnosis for this patient.

The Jessa Hospital dataset has also been annotated in multiple ways to accommodate our research questions, and various subsets of the original Jessa Hospital dataset were created for different research questions. We categorize the datasets as follows:

- `Jessa-AF`, the subset of the original Jessa Hospital dataset, without any unusable discharge notes (e.g., empty records, or records only containing a headline). This dataset contains columns for the discharge note and a binary label in case the note has been coded with any given ICD code for AF
- `Jessa-CHADSVASc`, a subset of the Jessa-AF dataset in which only AF-positive cases are included, containing the actual discharge note and various possible annotated information. Several purposes for which this dataset was annotated will be covered in more detail later in this thesis.
- `Jessa-CHADSVASc-3`, a subset of the Jessa-CHADSVASc dataset, which was restricted to records that contained a CHA_2DS_2–VASc score that was present in the dataset at least three times. This was implemented to eliminate records containing scores that were extremely rare in the dataset.
- `Jessa-CHADSVASc-10`, a subset of the Jessa-CHADSVASc dataset, which was restricted to records that contained a CHA_2DS_2–VASc score that was present in the dataset at least ten times. This was implemented to eliminate records containing scores that were extremely rare in the dataset.

`Jessa-CHADSVASc` and all of its subsets were created by using various regular expressions and manual labeling techniques to annotate the available data and retrieve the appropriate CHA_2DS_2–VASc scores.

Data Annotation and Labeling
The annotation for the `Jessa-CHADSVASc` dataset was done using both automated (Python script with regular expressions) and manual approaches to ensure high-quality labels. The records were then manually checked multiple times to ensure the correctness of the labels, as this would have a big influence on the training and performance of the resulting models.

The manual annotation process proposed some challenges that are inherent to text processing in general. As mentioned previously, medical professionals often document CHA_2DS_2–VASc scores using various notations, including abbreviations or with spelling mistakes. Some records even contained contradictory score information, such as mentioning both 'CHA2DS2-VASc = 3' and 'CHADS-VASC score 4' within the same document. These inconsistencies reflect real-world clinical documentation practices but may propose some challenges for creating a performant model.

Various other annotated datasets were also created in case more complex models would be created in this thesis, which included annotations for the start index and end index of a score within a text, as well as masked text where different parts of the CHA_2DS_2–VASc score and text was masked to be able to retrieve it using clinical reasoning, or so it could have been used to fine-tune the used model on specific, such as Named Entity Recognition (NER) or Question-Answering. However, these datasets, and these approaches, were not used in the final experiments of this thesis and will therefore be not discussed in detail.

2.3 Data Characteristics and Limitations

The datasets we are using have various interesting characteristics and propose some challenges that are essential to understand before discussing our methodologies and interpreting our results in this thesis. These characteristics and the constraints that impacted our methodological choices are discussed in this section.

2.3.1 MIMIC-IV Datasets

The `MIMIC-IV-Ext-Cardio` dataset consists of more than 40.000 English patient records sourced from Beth Israel Deaconess Medical Center (Johnson et al., 2023a), while the `Jessa-AF` dataset contains 12,516 Dutch cardiac discharge notes from hospital admissions between 2016 and 2023, after filtering out empty notes.

2.3.2 Jessa Hospital Datasets

The `Jessa-AF` dataset consisted of 12,516 complete Dutch discharge notes, with a significant class imbalance of 88% AF-negative cases (11,008 records) versus 12% AF-positive cases (1,508 records). The `MIMIC-IV-Ext-Cardio` dataset has a smaller class imbalance, with 40.73% of cases (17,300 records) being AF-positive. The class imbalance in the `Jessa-AF` dataset, even though it is rather large, is mostly representative of real-world clinical populations. Studies of hospitalized cardiac patients often reported AF prevalence rates of around 15% for different groups of patients, such as the study done by Ergün et al. (2021) and Constante et al. (2024). Particularly, Constante et al. researched the prevalence of different heart diseases, including AF, in pediatric patients, in which the study concluded 16.7% of patients presented with AF (Constante et al., 2024). Working together with the Jessa Hospital, we learned that the class imbalance of the retrieved dataset was on the bigger side, but nonetheless a good test to see how well any of our methods would work.

The `Jessa-CHADSVASc` dataset consisted of 1,496 AF-positive records with 17 unique labels describing the different scores, while the `Jessa-CHADSVASc-3` dataset consisted of 1,489 records with 12 labels after removing 7 records from untrainable classes (less than 3 cases per class). Both of these datasets have a significant class imbalance, dominated by cases where no explicit CHA_2DS_2–VASc score was mentioned (more than half of all records). Only around 674 records (around 45%) from `Jessa-CHADSVASc` contained actual CHA_2DS_2–VASc scores. A total of 7 additional records were filtered out for the `Jessa-CHADSVASc-10` dataset, compared to `Jessa-CHADSVASc-3`, as these 7 records were labeled with a very rare class. However, this does not influence the overall distributions for AF-positive and AF-negative records too much. The records in `Jessa-CHADSVASc-10` only have 10 different classes in their training, validation and test sets, compared to 12 in `Jessa-CHADSVASc-3` and 17 in `Jessa-CHADSVASc`.

The original possible labels for `Jessa-CHADSVASc` we established are shown in table 2.1, including the occurrence of records with this label within the dataset.

We notice that for scores containing inequality signs, their prevalence in the dataset is a lot lower or simply nonexistent. The label that stands out from said subset of records is label 13, indicating a CHA_2DS_2–VASc score bigger than 2. This makes sense since any CHA_2DS_2–VASc score equal to or bigger than 2 indicates clinical significance and therefore might be mentioned more often.

Looking at table 2.1, we can deduce the labels for both `Jessa-CHADSVASc-3` and `Jessa-CHADSVASc-10`.

Some example notations of the scores in the discharge notes were

- *CHA^2Ds^2- VASc van 2*
- *ChasVasc 4/9*
- *CHA2DS2-VASc (=2)*
- *CHA^2Ds^2-VASc score is 7/9*
- *CHA2DS2-VASc Score - 1*
- *CHA2DSV2VASc =0*

Numeric label	Label description	Prevalence
0	Score of 0	58 records (3.9%)
1	Score of 1	74 records (4.9%)
2	Score of 2	133 records (8.9%)
3	Score of 3	149 records (10.0%)
4	Score of 4	129 records (8.6%)
5	Score of 5	71 records (4.7%)
6	Score of 6	21 records (1.4%)
7	Score of 7	10 records (0.7%)
8	Score of 8	3 records (0.2%)
9	Score of 9	3 records (0.2%)
10	No score	827 records (55.3%)
11	Score > 0	0 records
12	Score > 1	2 records (0.1%)
13	Score > 2	11 records (0.7%)
14	Score > 3	0 records
15	Score > 4	1 record (0.1%)
16	Score > 5	2 records (0.1%)
17	Score > 6	0 records
18	Score > 7	0 records
19	Score > 8	0 records
20	Score < 1	0 records
21	Score < 2	1 record (0.1%)
22	Score < 3	1 record (0.1%)
23	Score < 4	0 records
24	Score < 5	0 records
25	Score < 6	0 records
26	Score < 7	0 records
27	Score < 8	0 records
28	Score < 9	0 records

Table 2.1: Labels for the `Jessa-CHADSVASc` dataset, including their prevalence in the dataset

- *ChadsVasc groter dan 2*
- *CHADSVasc (4/9)*
- *CHADS - vasc 3*

Since we had fewer records than in our previous experiments with the `MIMIC-IV-Ext-Cardio` dataset, we anticipated it would take more work and research to get good results right away. If any possible notation of a CHA_2DS_2–VASc score does not have a label and is not mentioned in table 2.1, you may assume it was not present in our dataset (e.g. a score written as *'hoge chadsvasc score'* (*'high chadsvasc score'*) was not found in our dataset).

We anticipated that the biggest data-related limitations in this thesis would be

- Class imbalance
- Absence (or very low prevalence) of certain labels
- Difficulty finding models that properly interpret Dutch medical language
- Preprocessing bias may lead to explicit score extraction over 'clinical reasoning'

This is why we anticipated that choosing certain evaluation metrics would have a bigger impact on our methodology, and it was definitely encouraged to take a look at existing pre-trained models that perform better on clinical text. The methods we employed for each implementation

will be discussed in the respective sections. In this thesis, we did not tackle the problem of absence or low prevalence of certain labels. In any future research on this topic, it may be possible to generate synthetic data alongside the existing data to enhance the datasets. However, various regulations surrounding the use of actual patient data (from existing datasets) make this process more tedious.

For the `Jessa-CHADSVASc` datasets specifically related to the clinical significance of the CHA_2DS_2–VASc score, patients with a significant CHA_2DS_2–VASc score (≥ 2) had significantly longer discharge notes (mean: 1,187 words) compared to those not requiring treatment (mean: 980 words), with a meaningful effect size ($p < 0.001$). The 207-word difference in average likely reflects the increased documentation requirements for patients with multiple comorbidities that contribute to higher CHA_2DS_2–VASc scores.

2.4 Data Preprocessing

Data pre-processing is an important step to take when working with textual data. The pre-processing steps that we defined are used to transform the human-readable hospital discharge notes in our datasets into a format suitable for machine learning models. This may seem rather simple at first glance, but it is generally known that the performance of machine learning or deep learning applications heavily depends on the pre-processing that is done on the data, as well as the approaches to represent the data in a machine-readable format (Kaur et al., 2023).

2.4.1 Text Preprocessing Pipeline

Our pre-processing pipeline consists of several sequential steps designed to clean, normalize, and structure the medical text data such that it can be properly interpreted by our following approaches to classify documents or extract information. We will use following generated Dutch discharge note excerpt to illustrate each pre-processing step:

```
PATIËNT: [NAAM] [VOORNAAM], geboortedatum: 15/03/1952
OPNAME DATUM: 23/11/2023 - ONTSLAG: 28/11/2023
DIENST: Cardiologie

BESLUIT EN BESPREKING:
De 71-jarige patiënt werd opgenomen wegens voorkamerfibrillatie de
   novo.
CHA2DS2-VASc score van 4 werd berekend (leeftijd 2, hypertensie 1,
   diabetes 1).
Anticoagulatie gestart met Rivaroxaban 20mg 1x per dag.

MEDICATIE BIJ ONTSLAG:
- Rivaroxaban 20mg 1x/dag
- Metoprolol 50mg 2x/dag
- Losartan 100mg 1x/dag
```

Different experiments were done to determine a good pre-processing pipeline for the various datasets we used to train the models mentioned in this thesis. All datasets went through various stages of text pre-processing. These steps included:

1. Data cleaning
2. Data normalization
3. Data annotation & Label mapping
4. Stop-word removal
5. Lemmatization or stemming

6. Tokenization
7. Vector representation conversion

2.4.2 Data Cleaning

The first step involves cleaning the data, and removing any unnecessary information from the dataset. This can include unused columns, or certain elements from the discharge notes that are not relevant or would influence our approaches negatively. In our case, this meant removing patient identification, metadata, and formatting artifacts. Although the data for both use cases had been anonymized, certain leftovers were still present in the data, such as sentences containing '[FIRSTNAME] [LASTNAME] was admitted to Cardiology dept.' These identifiers, including other examples like birth or admission dates, were removed from the text. We also removed characters such as training whitespaces and line breaks. For certain datasets, such as the `Jessa-CHADSVASc` sets, we removed starting section headings specifically.

After data cleaning, our text may look as follows:

```
De 71-jarige patiënt werd opgenomen wegens voorkamerfibrillatie de
   novo.
CHA2DS2-VASc score van 4 werd berekend (leeftijd 2, hypertensie 1,
   diabetes 1).
Anticoagulatie gestart met Rivaroxaban 20mg 1x per dag.
MEDICATIE BIJ ONTSLAG:
Rivaroxaban 20mg 1x/dag
Metoprolol 50mg 2x/dag
Losartan 100mg 1x/dag
```

2.4.3 Data Normalization

When talking about data normalization, we refer to the practice of standardizing text to handle common variations or elements that are present in the dataset. For example, in the case of our AF classification models, we made sure to perform case normalization, such that all text is set to lowercase. 'Diabetes' and 'DIABETES' would therefore both me represented in the same way, and would become 'diabetes'. As mentioned in the data cleaning, processed such as whitespace normalization can also be considered data cleaning. However, data normalization often refers to a more aggressive normalization that will have a bigger impact on the approaches that follow.

An example of a normalization that can be applied to our data is abbreviation expansion, such as expanding common medical abbreviations to their full forms (such as AF to Atrial Fibrillation). We can also perform format standardization, such as standardising the way CHA_2DS_2–VASc scores are being written in our training set. For example, we can normalize our data by making sure a CHA_2DS_2–VASc score written as 'CHA2DS2VASC is 2' or 'CHADSVASC of 2' will be transformed to follow a predefined format, such that both terms would be normalized to 'CHADSVASc = 2'. We decided not to implement this, as this also meant reducing the possible variations in our dataset. For us, it was important for our model to train on as many variations as we had available, such that it would also learn to generalize and work well on unseen data. For other purposes our approaches, heavier forms of normalization may be beneficial. However, in our case, apart from using whitespace and case normalization after our data cleaning, our data was left as authentic as possible.

As may be apparent from our implementation, data normalization is a step that is often merged into other steps of the pre-processing pipeline.

2.4.4 Stop-word Removal

Stop-word removal is a pre-processing technique that eliminates frequently occurring words that typically provide little discriminative value for classification tasks, often referred to as noise. In Dutch, some common stop-words are '*de*' ('the'), '*en*' ('and'), '*te*' ('too'), and '*van*' ('of'). While these words appear frequently throughout clinical documents, they rarely contribute meaningful diagnostic information that distinguishes between different medical conditions. The benefits of stop-word removal are dimensionality reduction, noise reduction computational efficiency, and focus on discriminative terms. By removing stop-words, we reduce the amount of terms in our document, which leads to lower-dimensional vector representations of our textual data. This also leads to computational efficiency, while also reducing the size of the vocabularies that are built during the fitting-stage of TF-IDF-vectorization. By removing stop-words, we also reduce noise by removing words that do not directly have any clinical value or diagnostic power.

However, stop-word removal in medical contexts can also impact the training negatively. Some terms can carry important clinical meaning depending on the context, particularly in the context of negation ('not diagnosed with'). Alternative approaches, such as skip-gram subsampling described by Mikolov et al. (2013), preserve stop-words but reduce their influence through probabilistic downsampling based on frequency. Instead of the complete removal of stop-words, this technique uses a form of subsampling where words that appear frequently in the text are kept with some probability attached, which will reduce their influence during training. Terms are downsampled more aggressively as they appear more frequently in the text (Mikolov et al., 2013). This may be necessary when completely removing stop-words risks losing contextual information that may sometimes be necessary to keep information on the relationships between words.

For AF Classification Models (XGBoost + TF-IDF), we implemented comprehensive stop-word removal using NLTK's predefined Dutch stop-word list (Loper and Bird, 2002) and common administrative phrases such as *'werd opgenomen in de dienst cardiologie'* ('got admitted to the cardiology department'). This more aggressive approach is justified because our TF-IDF vectorization focuses on discriminative term frequencies, making stop-words counterproductive. While XGBoost focuses on individual features and grammatical relationships are not directly targeted in our approach, the removal of stop-words will have a more positive influence.

For our CHA_2DS_2–VASc extraction model we chose to preserve stop-words in our fine-tuning approach. This was mostly chosen as transformer models can rely on grammatical context to understand relationships between concepts. While we used pre-trained embeddings, they include representations for stop-words that contribute to overall sentence understanding.

In our example, we will remove common stop-words and perform our data normalization, which will result in the following text:

```
71-jarige patiënt opgenomen wegens voorkamerfibrillatie novo.
cha2ds2-vasc score 4 berekend (leeftijd 2, hypertensie 1, diabetes
   1).
anticoagulatie gestart rivaroxaban 20mg dag.
rivaroxaban 20mg dag
metoprolol 50mg 2x dag
losartan 100mg dag
```

2.4.5 Lemmatization & Stemming

For our `Jessa-AF` dataset, we applied stemming using NLTK's SnowballStemmer for Dutch. Stemming reduces words to their root form by removing suffixes, and is a rather 'harsh' algorithm compared to lemmatization.

For our **MIMIC-IV-Ext-Cardio** dataset, we used lemmatization using NLTK's WordNetLemmatizer. This reduces words to their base form, also called canonical form, while preserving its

Original Word	Stemmed Form (Porter)	Lemmatized Form (WordNet)
diagnosed	diagnos	diagnose
diagnosing	diagnos	diagnose
admission	admiss	admission
admitted	admit	admit
medications	medic	medication

Table 2.2: A table showing various common terms used in discharge notes with their respective lemmatized and stemmed version

meaning. Table 2.2 shows some examples of common terms and how they would be lemmatized or stemmed.

We can see from the examples shown in the table that lemmatization produces similar words, as similar terms are often reduced to the same base form. With stemming on the other hand, we notice that it simply 'cuts off' the pre- or suffixes of words and reduces words to their root form (stem). In this sense, stemming is considered more aggressive. Lemmatization therefore requires language-specific rules that must be followed, and is often a bit slower than stemming.

Adding stemming to our previous example would give following result:

```
71-jarig patient opgenom weg voorkamerfibrill novo.
cha2ds2-vasc score 4 bereken (leeftijd 2, hypertens 1, diabet 1).
anticoagulat gestart rivaroxaban 20mg dag.
rivaroxaban 20mg dag
metoprolol 50mg 2x dag
losartan 100mg dag
```

As mentioned previously, stemming was used on our Dutch datasets, while lemmatization was used on our English datasets. This will be discussed further for each implementation throughout the following chapters.

2.4.6 Tokenization

The tokenization step takes the input text and splits it into individual tokens. This is usually done using a predefined 'tokenizer', which provides the appropriate functionality to split text into tokens based on certain rules. A simple way of tokenizing human-readable text is by splitting the text on whitespaces. Given our example, this would produce the following tokens:

```
['71-jarig', 'patient', 'opgenom', 'weg', 'voorkamerfibrill', 'novo
    .',
 'cha2ds2-vasc', 'score', '4', 'bereken', 'leeftijd', '2', '
   hypertens',
 '1', 'diabet', '1', 'anticoagulat', 'gestart', 'rivaroxaban', '20mg
    ',
 'dag', 'rivaroxaban', '20mg', 'dag', 'metoprolol', '50mg', '2x', '
   dag',
 'losartan', '100mg', 'dag']
```

2.4.7 Vector Representation Conversion

Another important pre-processing step when working with various NLP machine learning methods for classification or extraction is the conversion of text or tokens into vector representations. The text has to be represented in some machine-readable format, as most machine learning algorithms can only work with numerical data. To create a mapping from the categorical textual

Component	Specification
Device	MacBook Pro (14-inch, 2023)
Operating System	macOS Sonoma 14.0
CPU	Apple M2 Pro chip, with a 10-core CPU with 6 performance cores and 4 efficiency cores
GPU	Apple M2 Pro chip with integrated 16-core GPU
Memory	Apple M2 Pro chip with 16 GB unified memory
Storage	512 GB SSD with roughly 100 GB of free space

Table 2.3: Specifications for MacBook Pro (14-inch, 2023), personal device used in phase 1

data to numerical data, the text is represented in some numeric format. In this thesis, we use various approaches depending on the methods we will be using.

The simplest way to represent text as a numerical vector is to create a matrix and treat each unique word as a separate dimension in a high-dimensional space. For example, given the text 'patient has atrial fibrillation' we would like to encode, we could represent each word as follows:

```
'patient':       [1, 0, 0, 0]
'has':           [0, 1, 0, 0]
'atrial':        [0, 0, 1, 0]
'fibrillation':  [0, 0, 0, 1]
```

Encoding large documents like this will logically produce sparse, high-dimensional vectors, as most elements in the vector will be zero, and they will be quite long. This is why other approaches exist, such as embeddings, which create dense and lower-dimensional representations. For example, they can choose to represent similar words in a similar vector, and can therefore reduce the amount of dimensions that an embedded vector would need. The distances between vectors can also be calculated using various methods. More extensive approaches like pre-trained embeddings exist to better capture contextual meaning by analyzing relationships within sentences, including the morphological patterns and dependencies within sentences or documents. These methods, for example, can recognize that identical word combinations can have different meanings, depending on context, and may assign different numerical representations that reflect their significance.

The exact methods used in our approaches will be discussed in their corresponding chapters.

2.5 Hardware and Software Environment

2.5.1 Local Devices

The initial development of a classification model for automated AF classification on `MIMIC-IV-Ext-Cardio`, was conducted mostly on a personal laptop with fairly limited computational resources. The specifications of this personal computer can be found in table 2.3.

The original Jessa Hospital dataset was later securely retrieved and stored on another laptop provided by Hasselt University (UHasselt). In accordance with the pre-defined agreements between the ethical committees and relevant departments of Jessa Hospital and UHasselt, the Jessa Hospital dataset and any subsets created from it were only accessed through the UHasselt laptop. The specifications of this laptop can be found in figure 2.4.

Component	Specification
Device	Thinkpad T495
Operating System	Windows 11 Home
CPU	AMD Ryzen 7 PRO 3700U
GPU	Integrated Radeon Vega Mobile Gfx
Memory	16 GB
Storage	1TB SSD

Table 2.4: Specifications for ThinkPad T495, device provided by Hasselt University, used in phase 2

Component	Specification
Device	GPU node with NVIDIA P100 on partition `gpu_p100`
CPU	2 Intel Xeon Gold 6140 CPUs@2.3 GHz (Skylake), 18 cores each, with 1 NUMA domain and 1 L3 cache per CPU
GPU	4 NVIDIA P100 SXM2 GPUs@1.3 GHz, 16 GiB GDDR each, connected via NVLink
Memory	192 GiB RAM, default 5000 MiB per core
Storage	200 GB SSD local disk

Table 2.5: Specifications for a GPU node with NVIDIA P100 GPUs on the `gpu_p100` partition

2.5.2 High-Performance Computing on VSC

However, for the Jessa Hospital dataset, we were given permission to utilize the Flemish Supercomputer Center (Vlaams Supercomputer Centrum) (VSC) to conduct any further research with this dataset. The use of the high-performance computing infrastructure of the VSC was mostly chosen to speed up compute times and allowed us to work with large language models that may have required more memory than was available on our personal devices.

The VSC provides various tiers of parallel processing. In the context of this thesis, we gained access to the KU Leuven/UHasselt Tier-2 Clusters and made use of the `Genius` and `wICE` clusters.

The `Genius` cluster has thin nodes, large memory nodes, and GPU nodes. We mostly used GPU nodes for any model training or fine-tuning of existing language models. These GPU nodes provided an interface to work with NVIDIA GPUs, which proved to be very useful in reducing overall compute times compared to running our code on a personal laptop. For the training of the Jessa-CHADSVASc model, we used the HuggingFace Transformers library (Wolf et al., 2020), which in its turn also uses the PyTorch library (Paszke et al., 2019), and enables us to use GPU acceleration with CUDA. Table 2.5 shows the specifications of a standard GPU node on the `gpu_p100` partition. While these nodes offer solid performance, they are typically shared among multiple users, which can lead to longer queue times. However, GPUs are reserved per user, so once a job starts, the assigned GPUs are dedicated. In our experience, requesting 2 GPUs provided a good balance between having enough memory available for training while keeping the waiting time in the queue relatively short.

The `wICE` cluster also has thin nodes, large memory nodes, interactive nodes and GPU nodes. The `wICE` cluster was mostly used for its interactive nodes, as a way to manage and create jobs that run our Python scripts to create and train our models. The GPU nodes on the `wICE` cluster, which cost a few extra credits per usage time, were not used. This was mainly because the `Genius` GPU nodes appeared to be sufficiently performant for our use case. To train our binary classification model for AF detection, we did use the regular IceLake compute nodes on the `wICE`

Component	Specification
Device	wICE interactive node
Operating System	Red Hat Enterprise Linux
CPU	2 Intel Xeon Platinum 8360Y CPUs@2.4 GHz (IceLake), 36 cores each, with 1 NUMA domain and 1 L3 cache per CPU
GPU	N/A
Memory	256 GiB RAM, default 3400 MiB per core
Storage	960 GB SSD local disk

Table 2.6: Specifications for an interactive node on the wICE cluster of the VSC

cluster. Each of these compute nodes had significantly more cores, memory, and storage, than the personal laptop. According to the VSC documentation (VSC (Vlaams Supercomputing Center), 2025), the clusters run the Linux kernel and a GNU operating system. Specifically, they state, all HPC clusters currently run some flavor of Red Hat Enterprise Linux. The exact specifications of an interactive compute node on `wICE` can be found in figure 2.6.

The VSC uses the Slurm workload manager. Several Slurm-files were created to submit jobs on the appropriate clusters with a set of pre-defined specifications.

Chapter 3

Atrial Fibrillation Classification

3.1 XGBoost and TF-IDF Baseline Model

To create a baseline model to compare the rest of our results to, we decided to use XGBoost in combination with TF-IDF vectors. This decision was based on previous research surrounding automated classification of ICD codes from hospital discharge notes by Falter et al. (2024).

Our AF classification implementation consists of two main components: a custom TF-IDF preprocessing pipeline and an XGBoost classifier with hyperparameter optimization.

3.1.1 TF-IDF Vectorization

We start out with human-readable discharge notes that we need to be able to feed into our models. For our AF classification models, we decided to use TF-IDF as our approach to turn sentences, words, and terms into a machine-readable format, ready for text processing. TF-IDF is a technique that works by calculating two different values:

- Term frequency (TF), for a term t in document d, calculated as

$$TF(t,d) = \frac{\text{appearances of t in d}}{\text{total terms in d}}$$

- Inverse Document Frequency (IDF), for a term t and a corpus (vocabulary) D

$$IDF(t,D) = log\frac{\text{number of documents in D}}{\text{number of documents in D that contain t}}$$

One is often added to the denominator of the IDF-score to prevent division by zero in case no documents are found, and a logarithm is used such that the scores scale and do not significantly fluctuate given a small or large number of documents. The final TF-IDF score is then calculated by

$$TF - IDF(t,d,D) = TF(t,d) * IDF(t,D)$$

The IDF values are calculated based on the vocabulary (corpus) that was discovered during fitting. This means that once set, these values remain fixed for all subsequent transformations.

The TF-IDF vectorization was configured with unigrams (n-grams with n = 1) to focus on individual medical terms. While our unigram approach shares similarities with a traditional Bag-of-Words model in terms of considering individual terms, TF-IDF differs by weighting terms based on both their frequency within documents (TF) and their rarity across the entire corpus (IDF). This weighting helps emphasize distinctive medical terminology that is particularly informative for classification. A custom tokenizer was used to pre-process our text and return appropriate tokens for the next (training) steps, and calculation of our TF-IDF vectors is done on the full training set.

At training time, we calculate TF-IDF on our training subset of the data and use these as features for the classification model. The 'vocabulary' that is discovered through this initial fitting of the TF-IDF vectors is based on the training data. Of course, the TF-IDF vectorization process influences model performance and its generalization capabilities. Our approach finds a balance between building a large enough vocabulary without having to compromise on potentially introducing data leakages.

At training time, the TF-IDF vectorizer will analyze the training subset to construct a vocabulary of unique terms and calculate their inverse document frequency (IDF) values. This vocabulary will be used for all subsequent text transformations and represents the model's 'known terms'. We implemented a two-stage approach:

1. Vocabulary fitting
2. Transformation

In the vocabulary fitting stage, the TF-IDF vectorizer is fit on the full training data subset (incl. validation) to ensure we capture as many terms from all available training data. The transformation of the data is done separately, where each subset (train, validation, test) will be transformed using the initial vectorizer, but no additional fitting is done to prevent accidental data leakages to the test set. We note that the vectorizer does take both the training and the validation set into account when building the vocabulary, which seems counter-intuitive. However, doing this only affects the feature weights marginally as we are discovering vocabulary and not learning predictive patterns from the data yet, and helps in solving a bigger problem: Out-of-Vocabulary (OOV) words.

OOV problems are introduced when unseen terms are encountered at various steps in the training and testing process. They may also artificially deflate the performance during validation. For example, we consider a document from the validation set with following sentence 'Patient presents with paroxysmal atrial fibrillation and bradyarrhythmia'. If the term 'bradyarrhythmia' was not present in the training-only vocabulary, the TF-IDF transformation would process this as: 'Patient presents with paroxysmal atrial fibrillation and [IGNORED]'. This may cause useful features or variations of spellings for specific terms relating to any diagnostic information for AF to be ignored.

The custom tokenizer we defined for our TF-IDF approach, helps us handle the specific characteristics of medical text and have more control over what happens while text is split up into tokens. The preprocessing pipeline includes the following steps:

1. Tokenization
   Since we are working with human-written and -readable medical text, whitespace-based splitting seemed most effective, where terms are typically separated by whitespaces. This step will return a list of individual tokens from the discharge note.
2. Lemmatization & Stemming
   Using the `WordNetLemmatizer` by NLTK (Loper and Bird, 2002), we performed lemmatization on our tokens (words) for the `MIMIC=IV-Ext-Cardio` dataset, which reduced each word into its base form while still preserving the meaning. For example, 'diagnosed' and 'diagnosing' will both be lemmatized into 'diagnose' as their base form. For the `Jessa-AF`

dataset, we performed stemming with the `SnowBallStemmer` for Dutch by NLTK. We did this since this was the easiest approach without including any additional libraries. However, we note that spaCy, another interesting library in Python, offers various interesting features to set up pre-processing pipelines as well.

3. Noise Removal
We removed unnecessary numerical values such as dates and timestamps, as well as non-alphanumeric tokens (punctuation and special character sequences). We also removed stopwords retrieved from NLTK's dataset of common English stopwords (Loper and Bird, 2002). This way we reduce text length and dimensionality of the resulting vectors while retaining relevant clinical information. The specifics for each dataset are discussed in one of the earlier sections on the datasets.

3.1.2 XGBoost

XGBoost (eXtreme Gradient Boosting) is a more traditional machine learning method that implements gradient-boosted decision trees (GBDT). The XGBoost algorithm creates a number of weak models (usually decision trees) sequentially, where each model corrects errors made by the previous model. This is often called an 'ensemble method', indicating that we combine multiple individual 'weaker' models to create one more performant model. More specifically, at each iteration of the algorithm, XGBoost fits a new tree to correct the residual error of the last iteration. The final prediction is the sum of predictions from all the trees that were generated, weighted by their learning rates. This makes XGBoost good at classification problems on structured data like our datasets of discharge notes. Therefore, for the task of AF classification, we take advantage of a few benefits that come with XGBoost.

One of these benefits is the proper handling of sparse features, which is a term used to describe features in a dataset with mostly zero or null values. While we use TF-IDF vectors with a pre-defined vocabulary, as we mentioned before, these vectors may often contain zero values for OOV terms, and due to the nature of TF-IDF vectors, sparse features will often be generated. XGBoost is known to handle sparse feature matrices well. XGBoost also includes built-in regularization methods. More specifically, L1 and L2 regularization is done during training to prevent overfitting. To understand how these influence the training, we must first look at how XGBoost works behind the scenes.

XGBoost uses K decision trees to make predictions. For a given input x_i, the prediction is:

$$\hat{y}i = \phi(x_i) = \sum k = 1^K f_k(x_i)$$

where f_k represents the k-th tree and $\phi(x_i)$ is the final prediction (Chen and Guestrin, 2016).

The algorithm has the following regularized objective function:

$$L(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k)$$

that it tries to minimize, where l is a loss function (such as logistic loss for binary classification) and Ω is the regularization term that penalizes model complexity (Chen and Guestrin, 2016).

The regularization term $\Omega(f_k)$ for each tree is defined as:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda |w|^2$$

where

- T is the number of leaves in the tree
- w represents the leaf weights, which are the raw prediction scores that each leaf outputs.

```
# Pseudocode for split evaluation
for each_feature in features:
    for each_possible_threshold in feature_values:
        # Split samples into left and right based on threshold
        left_samples = samples where feature < threshold
        right_samples = samples where feature >= threshold

        # Calculate gradients sums for each side
        G_L = sum(gradients of left_samples)
        G_R = sum(gradients of right_samples)
        H_L = sum(hessians of left_samples)
        H_R = sum(hessians of right_samples)

        # Calculate gain from this split
        gain = calculate_gain(G_L, G_R, H_L, H_R)

        # Keep track of best split so far
        if gain > best_gain:
            best_gain = gain
            best_split = (feature, threshold)
```

Figure 3.1: Pseudocode for gain calculation in XGBoost, adapted from *'XGBoost: A Scalable Tree Boosting System'* by Chen and Guestrin (2016)

- γ controls the minimum loss reduction required for splits (L1-like regularization on tree structure)
- λ controls the L2 regularization on leaf weights

XGBoost begins with all training samples in a single root node. Each sample has a gradient g_i and Hessian h_i from the current ensemble model's predictions. Then, XGBoost considers every possible split on every feature present in that ensemble model. XGBoost builds trees one split at a time. When it considers making a split in a tree, it calculates the gain, also defined as loss reduction, from that split to determine if it wants to create a split or not.

The gain is defined as follows:

$$\text{Gain} = \frac{1}{2}\left[\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right] - \gamma$$

where

- G_L, G_R = sum of gradients in left and right child nodes
- H_L, H_R = sum of second-order gradients (Hessians) in left and right nodes
- λ = L2 regularization parameter
- γ = minimum gain threshold

In figure 3.1, pseudocode is defined on how XGBoost decides to create a split in the tree. XGBoost will create a split in the decision tree if the best gain found is larger than zero. If no split achieves a positive gain, the created node will become a leaf, and no further splits will be created from it (Chen and Guestrin, 2016).

For example, when creating our AF classification model, in a given node, the features of our TF-IDF vectors will be analyzed and used to determine splits. Features can be the presence of a term in our document, defined by a certain score in our TF-IDF vector. For simplification, we will use an example referring to three possible features: 'atrial', 'age $>$ 65' and 'gender =

female'. This introduces three possible splits that will be evaluated. For each split, the gain will be calculated from the first- and second-order gradients of the corresponding TF-IDF vector scores. If, for example, the gains for 'atrial' and 'age > 65' are positive and larger than zero, these features will be considered in creating a new split. If the split for 'gender = female' would generate a negative or zero gain, it will not be made, and this node will become a leaf. For the two features that generated a positive gain, the best one will be chosen to create a split. Let us assume the presence of the term 'atrial' gives us the highest positive gain, then this will be used to create two child nodes: a left and a right node. The process will then repeat for each child node.

Understanding how the splits and trees are built now makes it possible for us to understand how we can add L1 and L2 regularization.

In the case of L1 regularization, we have a γ term, also known as the minimum split loss, that will penalize the number of leaves. Higher γ values will lead to simpler trees with fewer splits, which in its turn performs a more selective feature selection at the tree level and reduces overfitting. To demonstrate this, we can refer to our example with three possible features to split on.

Without L1 regularization, the gain is calculated as previously mentioned, using the first- and second-order gradients of the vectors. Let us assume the following gains:

1. Split on 'atrial', gain = 0.1
2. Split on 'age > 65', gain = 0.05
3. Split on 'gender = female', gain = -0.05

This will result in the first split being chosen, as it has the biggest gain. For example, if the first feature indicates the presence of 'atrial' in the text, one child node will be created with a feature indicating 'atrial = 0' ('atrial' has a TF-IDF score of 0), and another child node with 'atrial > 0' ('atrial' has a TF-IDF score > 0). However, the second option can still propagate through to the child nodes and can still be a viable option for a split later on in the decision process.

With γ, we add a penalty to the gain function. In this case, given a $\gamma = 0.08$ the gains would be calculated as follows:

1. Split on 'atrial', gain = 0.1 - 0.08 = 0.02
2. Split on 'age > 65', gain = 0.05 - 0.08 = -0.03
3. Split on 'gender = female', gain = -0.05 - 0.08 = -0.13

We can see that with this penalty added, both options 2 and 3 will be rejected for a split. This will also propagate through the child nodes, and splitting on these features will be less likely as their gain will be reduced. This immediately implies the resulting trees will be simpler and will only consider splits with a strong gain.

For simplicity in our example, we split on features that indicated a presence or absence of a word from a document using the TF-IDF score. However, in reality, often times certain thresholds are considered. For example, a feature may be described as 'atrial with TF-IDF > 0.3', indicating the corresponding TF-IDF score for the term 'atrial' should be > 0.3. If used for a split, child nodes will be created that consider either documents with 'atrial with TF-IDF > 0.3' or documents with 'atrial with TF-IDF $\leq$ 0.3'. This means that the deeper we go into the tree, the less training data a node gets as it tries to learn patterns to recognize the classes effectively, as each split filters out data that doesn't meet the condition. This means that when we look into the node that was created for 'atrial with TF-IDF > 0.3', we only consider notes that conform to this condition and try to find other possible features for these documents that produce a high gain to create further splits.

This process is repeated until certain stopping criteria are met. Either no positive gain is found, which stops the tree from splitting. Another option is that the maximum depth of the tree has been reached. This is a hyperparameter that can be set to prevent trees from becoming too complex. Another parameter is `min_child_weight`, which will enforce a certain minimum to be reached before accepting a split. The 'child weight' refers to the sum of Hessians that is calculated during the split decision process. If, for example, the left child yields a high enough value, but the right does not, a split will not be created. Another reason the process may stop is when a node only predicts one class, indicating it has no point to split any further. Other stopping criteria can also be implemented but will not be discussed any further in this thesis.

For our models, we performed various hyperparameter optimization search techniques and ran several tests to find the best hyperparameters for both our AF classification models on both the `MIMIC-IV-Ext-Cardio` and `Jessa-AF` datasets. The final hyperparameters for each model differ slightly and are specific to our input data. The results are presented in our results section, where we go more in depth on this topic.

In our implementations, XGBoost serves as the classifier that learns to distinguish between AF-positive and AF-negative discharge notes based on the TF-IDF feature representations.

3.2 Training and Validation Approach

Our training and validation approach utilized a predefined three-way split of the `MIMIC-IV-Ext-Cardio` and the `Jessa-AF` dataset for both experiments. For each experiment, we kept a training set used during training, a validation set used during training to validate across different steps and help the model 'learn' the patterns in the text, and a test set, which is completely omitted during training and used to evaluate the final performance of the model.

Many machine learning approaches require at least some form of hyperparameter tuning to get the most optimal results given the dataset you provide.

To optimize our XGBoost classifier's hyperparameters while addressing the significant class imbalance between AF-positive and AF-negative samples in our dataset, we employed a stratified k-fold cross-validation strategy during our hyperparameter optimization.

In k-fold cross-validation, the training data is split into k folds that all have the same size. During training, each fold is used exactly once as a validation set, while the remaining $k - 1$ folds are then combined to form the training set. The model is trained k times, once for each fold, and then performance metrics are averaged across all folds to get a good estimate of model generalization.

However, with imbalanced datasets, a regular split can produce folds where the proportion of AF-positive records is much smaller or larger than in the original dataset, which can bias training and evaluation, and will not follow our pre-defined proportions. To prevent this, we used StratifiedKFold, which ensures that each fold has the same class distribution as the full training set. In our case, this meant that every fold contained roughly the same ratio of AF-positive and AF-negative samples.

In our experiment with both `MIMIC-IV-Ext-Cardio` and `Jessa-AF`, we implemented 5 folds during training. This resulted in five stratified subsets. In each iteration during training, four folds were used for training, and one fold was used for validation. The dataset was shuffled before being split into 5 folds, but a `random_state` variable was set to ensure reproducibility.

3.2.1 Hyperparameter optimization

For the hyperparameter optimization, we implemented scikit-learn's `HalvingRandomSearchCV` Pedregosa et al. (2011). Using successive halving hyperparameter optimization through the HalvingRandomSearchCV interface, we systematically reduced the search space of possible

parameters that would work best for our dataset. We finally looked at the following parameter space after running a few smaller tests with different search strategies and search grids:

- *max_depth*: [1, 2, 3, 4, 5, 6]
- *n_estimators*: [150, 200, 300]
- *learning_rate*: [0.01, 0.02, 0.03, 0.04, 0.05]
- *gamma*: [1, 1.5, 2, 2.5, 3]
- *subsample*: [0.6, 0.8, 1.0]

Scoring in each step of the training process was done on a pre-defined evaluation metric. We chose to use the F1-score to evaluate the model's performance. This means during each step of the training process, the model was evaluated on the validation set at that step, and the F1-score was calculated. Evaluation metrics will be discussed in more detail in section 3.3. In our case, at each step in the training, all folds will be used to train a model with its pre-determined train/validation split. At this step, the F1-scores from all folds will be averaged to obtain the model's cross-validation score. During training, the models with a higher F1-score will be favored over others. During the hyperparameter search, this means that the grid in searched in a way that favors a model with a higher F1-score.

3.2.2 Training/Validation/Test Split

Due to recommendations from the research team and having done some experiments, we employed the following strategy for the final training of our model AF classification models after running different experiments with different combinations of hyperparameters:

- The training (and validation) set maintained a proportion of 50% AF-positive, and 50% AF-negative cases to ensure training did not favor either class.
- The test set kept the original, imbalanced ratio between AF-positive and AF-negative cases to ensure generalizability and provide a good benchmark of how to model would perform in real-life scenarios.

Early stopping was implemented as well to prevent overfitting of the model. This allows the training process to stop when the performance of the model stops improving, to prevent overfitting to the training data.

To assess our `MIMIC-IV-Ext-Cardio` AF classification model's generalizability, we evaluated the optimized model on two test sets:

- The original test set, created from the cardiology department discharge notes from MIMIC-IV-Ext-Cardio, which the training and validation set was also sampled from.
- General MIMIC-IV test set: Discharge notes from other departments than the cardiology department, from the `MIMIC-IV-Ext-Diagnoses` dataset (excl. cardiology department notes), to further test performance across different types of discharge notes with less cardiology-related data and terms

This approach allows us to understand both the model's performance on similar data and its ability to generalize to a broader hospital context.

3.3 Model Evaluation Metrics

As we previously mentioned, the best hyperparameter configuration was selected based on the highest cross-validated F1-score. The F1-score balances the precision and recall and is particularly important for medical applications where both false positives and false negatives may have clinical implications. Focusing on accuracy was not our goal, but focusing on making

the least amount of false negatives (and positives) was of more importance in a clinical setting like ours.

Given the medical context and class imbalance inherent in the detection of AF, we employed following evaluation metrics to assess model performance:

- F1-score. Calculates the harmonic mean of the precision and recall, defined by

$$F1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} \times \text{Recall}}$$

  This harmonic mean implies that the more precision and recall deviate from each other, the worse the F1-score will be. This way, the F1-score equally weighs the importance of correctly identifying an AF-positive record (recall) and minimizing false AF diagnoses (precision).

- Precision. Measures the proportion of predicted AF records that are actually AF records. It is calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

  While seemingly less important than missing positive cases, a high precision also implies we avoid unnecessary reporting due to having too much false positives.

- Recall. Also known as sensitivity, it measures the proportion of the actual AF cases that were correctly identified. It is calculated as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

  A high recall value is crucial for clinical contexts to ensure no AF cases are missed.

Another metric that is often used in training contexts is the accuracy of a model. The accuracy is the proportion of correct predictions across both classes, and is calculated as follows:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Negatives}}{\text{Total Predictions}}$$

Accuracy may be misleading due to the fact we are working with imbalanced datasets. However, it is still a useful metric to get an overall overview of the model's capabilities.

We have different approaches to evaluate the model. We compute F1-scores for each class (AF-positive and AF-negative), as well as a macro-averaged and a weighted F1-score. A macro-averaged F1-score will take the unweighted mean of metrics across both our AF-positive and AF-negative class, given equal important to both AF and non-AF performance. The weighted average will weigh the F1-score by its class's support (number of samples), which provides a better measure when working with class imbalance like in our case. This is why when talking about a general F1-score, we often mention a 'weighted' F1-score.

3.3.1 Larger n-grams

To improve the model's ability to capture contextual information and handle negation patterns commonly found in clinical discharge notes, we enhanced our initial unigram-based approach by implementing variable-length n-gram features.

Our emphasis was mostly on negation handling. We did a limited amount of feature-engineering, where we extracted the most common negation patterns found in our training data, including '*geen_vkf*' ('no_af') and '*vkf_uitgesloten*' ('af_excluded'), and enforced common negation patterns to be treated as single tokens by replacing mentions terms like 'geen vkf' and 'zonder vkf' by their concatenated alternative, but abandoned this to not influence the performance indication of larger n-grams, and continued with our regular n-gram approach.

3.4 FastText Embeddings Exploration

To potentially improve the standard approach with TF-IDF and XGBoost, some experiments were done to improve our baseline approach by using FastText embeddings. FastText was chosen over other embedding methods (such as Word2Vec) due to its ability to better capture subword information, which is particularly useful in medical text where the terminology can be highly specialized. FastText captures subword information through character n-grams, enabling us to capture variations of relevant terms.

We utilized the pre-trained Common Crawl English FastText model provided by Facebook Research (Bojanowski et al., 2016), which contains 300-dimensional vectors trained on 600 billion tokens. To use FastText embedding with our XGBoost approach, we evaluated two different approaches:

1. Using XGBoost in combination with averaged FastText word embeddings
2. Using XGBoost in combination with TF-IDF weighted FastText word embeddings
3. Using XGBoost in combination with FastText sentence embeddings

3.4.1 Averaged FastText Word Embeddings

FastText can give us word-level embeddings, where each individual word gets a vector with 300 dimensions. A discharge note might have 1000+ words, with variable lengths. While our traditional XGBoost approach expects a fixed size input, it expects each document to be represented by the same number of features.

In this approach, we retrieved the FastText embedding vector for each word in the document. A document would be represented by a matrix, where each row is a word's embeddings. For example, the sentence 'Patient has atrial fibrillation' would be represented in a 4×300 matrix, as it has 4 words with a FastText vector of dimension 300.

The first problem we encountered was the variable length of our documents. We solved this by padding documents by zero vectors, so they would all have a matrix representation of the same length. Now we needed a way for our document to be represented using a single vector. We could flatten the matrix to feed it into XGBoost, but this led to incredibly long features, and padding with zeros would reduce the performance of our approach. We could also average our documents to a single vector, but this would lose all useful information that is captured.

Therefore, this approach was quickly abandoned.

3.4.2 TF-IDF Weighted FastText Embeddings

In an attempt to combine the strengths of both approaches, we experimented with weighting FastText embeddings by TF-IDF importance scores. For each word in a document, we calculate its TF-IDF importance score. Then, we get the FastText embedding vector for that word, and multiply it by the TF-IDF score to weigh the vector. Instead of averaging all embeddings, we computed a weighted average. This way, we tried to combine sparse (TF-IDF) and dense (embedding) representations.

However, this approach also yielded inferior results compared to the baseline XGBoost + TF-IDF approach, confirming the issues in our architectural approach, as well as the excellent baseline performance of XGBoost combined with TF-IDF. Our AF classification approach appears to benefit more from exact term presence than semantic similarity (embedding features).

3.4.3 FastText Sentence Embeddings

Instead of manually averaging word vectors, FastText has a built-in method specifically designed for sentence or document-level representations. Sentence vectors are also created by getting

weighted averages for the words in the sentence (or document). Using FastText's internal aggregation method yielded slightly better results (a weighted F1-score of 0.65 compared to 0.56 for manual averaging), but still significantly performed below the baseline.

Multiple attempts (TF-IDF weighting, dimensionality reduction, hybrid approaches) to enhance the FastText approach consistently failed to achieve competitive performance. The use of FastText was abandoned in favor of the second phase of our study: CHA_2DS_2–VASc extraction.

Chapter 4

CHA_2DS_2–VASc Score Extraction

In this chapter we discuss the implementation of our CHA_2DS_2–VASc score extraction model, which was created by using a pre-trained Dutch Medical Language Model, named MedRoBERTa.nl (Verkijk and Vossen, 2021). MedRoBERTa.nl is an implementation of RoBERTa that has been fine-tuned. Before we dive into how this model works, we first take a look at the background and current implementation of the transformer architecture, which lies at the base of the model that was used.

4.1 Transformers

Before the transformer architecture was introduced, NLP relied heavily on Recurrent Neural Network (RNN)s and Convolutional Neural Network (CNN)s. However, RNNs suffered from the vanishing gradient problem when processing long (medical) documents. This meant that it was difficult to capture dependencies between distant tokens, and it became increasingly difficult to as the document's length increased. Traditional implementations of RNNs will therefore 'forget' potentially useful context or relationships between words, the more tokens it processes. A popular example of an RNN is the Long Short Term Memory (LSTM) network, which reduces the vanishing gradient problem in traditional RNNs. It does this by using a gate-system to control which information should be forgotten, used as input, or used as output. CNNs, though more parallelizable than RNNs, have architectural constraints that make them less suitable for capturing a lot of context. However, we will not go into more detail on these architectures as these are not directly relevant for our implementation.

Most of these existing methods struggled to perform well in modeling longer context, and require many layers to capture long-range dependencies (Vaswani et al., 2023). Vaswani et al. also state that the introduction of the Transformer model directly targeted the inefficiencies of earlier methods such as RNNs and CNNs.

Apart from capturing long-term dependencies, it is also necessary to properly capture contextual information and understanding. This is where the self-attention mechanism comes into play. The self-attention mechanism, as it is referred to in 'Attention Is All You Need' (Vaswani et al., 2023), is used to weigh the importance of words in a given input sequence to better capture the relationships between them. For example, from each word in an input text, we can look at and relate to every other word in that input text. This is of course particularly useful for clinical text processing. Medical notes often contain references that span across sentences (referring to the long-range dependencies Vaswani et al. describe). It is also important to have a level of contextual understanding, as the same medical term can have a different meaning based

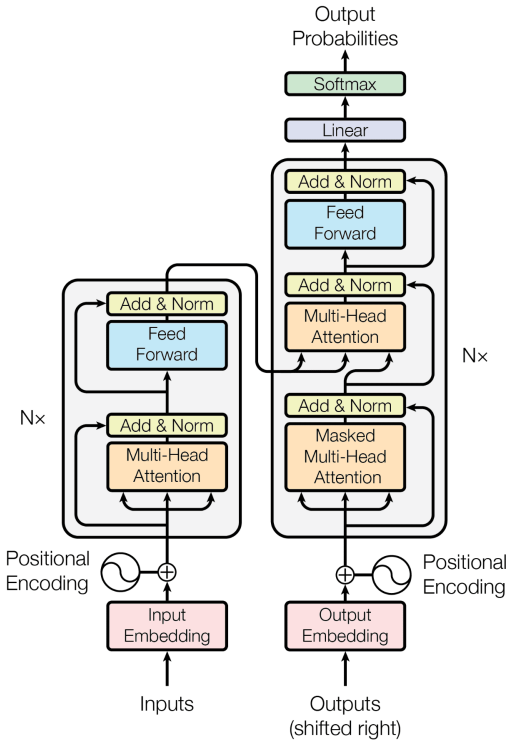

Figure 4.1: A visualization of the different components of the Transformer architecture, Figure 1 of Vaswani et al. (2023)

on the context. (e.g. 'patient has no history of stroke' and 'patient has risk of stroke' only have a one word difference). This also plays into the fact that negation handling is important, because the difference between 'not diagnosed with AF' and 'diagnosed with AF' make a big difference.

In figure 4.1 we see the architecture of the original transformer as described in Vaswani et al. (2023). Transformers can be divided into two parts: the encoder and the decoder. The key difference is that in the case of the encoder, it receives some input text and is responsible for understanding and extracting the relevant information, finally returning an embedded representation of the input. In the case of the decoder, it receives some embedded input and returns some expected output, which is often times human-readable Vaswani et al. (2023).

Originally, the transformer was developed for a translation task (English-to-French and English-to-German), which used both encoder and decoder. However, many popular implementations of the architecture only take on one of the tasks: encoding or decoding. Popular examples of models based on the decoder-only transformer architecture are the different GPT-variants by OpenAI, such as GPT-3 and GPT-4. However, in this thesis, we mostly experimented with the encoder-only models, more specifically, Bidirectional Encoder Representations from Transformers (BERT) and Robustly optimized BERT approach (RoBERTa).

4.2 BERT and RoBERTa Architecture

RoBERTa (Liu et al., 2019), as the name suggests, is an optimized version of BERT that has implemented some improvements. It has the same basic transformer architecture as BERT, but has been re-trained with optimized training procedures, as the research by Liu et al. stated that BERT underperformed, and no architectural changes were necessary to get the more performance out of the architecture (Liu et al., 2019). Both BERT and RoBERTa use the

encoder-only transformer architecture, consisting of multiple transformers blocks. Each of these blocks contain a multi-head self-attention mechanism. This means that instead of using only one attention mechanism, BERT and RoBERTa use multiple 'heads' that can run in parallel. Each of these heads will learn to focus on different types of relationships between words or tokens in the input data. For example, one head might capture dependency between words, such as subject-verb relationships, while another head might capture broader syntactic structures in the text. However, some studies, such as the ones done by Voita et al. (2019) and Michel et al. (2019), suggest that only a limited amount of self-attention heads do the heavy lifting, and many can be left out without a strong loss of performance.

A big difference between both architectures is that BERT focuses on two objectives, while RoBERTa removes the Next Sentence Prediction (NSP) objective that was present in BERT and focuses exclusively on Masked Language Modeling (MLM) Liu et al. (2019). Training for this specific objective already improved the accuracy for benchmark tests, such as MNLI-m, SQuAD 2.0 and GLUE, compared to BERT, according to Liu et al. (Liu et al., 2019).

In contrast to BERT, RoBERTa uses dynamic masking, larger batch sizes, more extended training, and byte-level BPE encoding with a larger vocabulary size. We will explain how these mechanisms come into play for our implementation in this section.

4.3 Fine-tuning of MedRoBERTa.nl

To create our CHA_2DS_2–VASc extraction model, we researched various possible options. It quickly became obvious that using an existing language model would be the best way to move forward. In figure 4.2 we present a diagram that shows the pipeline of our CHA_2DS_2–VASc extraction approach.

Previous research (Kim et al., 2022) (Harnoune et al., 2021) (Liu et al., 2021) (Ji et al., 2021) indicates good results are possible for various classification and extraction tasks within electronic health records or clinical context using BERT and RoBERTa-based language models. Numerous studies were conducted using language models that were trained on English medical text, as there was a scarcity of high-quality models that were trained on Dutch clinical text. The input datasets should often be translated to work well with pre-trained models on English medical text, and according to translation, multiple studies confirm that even when the solutions were effective when translating (bio)medical texts using machine translation, only a subset of them complied with rigorous translation quality assessment criteria (Zappatore and Ruggieri, 2024). Research by Névéol et al. indicates that medical texts contain a 'mixture of Latin and English terminology in addition to the local language' which 'adds a layer of complexity' that translation systems struggle to handle effectively (Névéol et al., 2018).

We therefore went on a search for any pre-trained models on Dutch medical language before trying out popular options such as BioBERT, ClinicalBERT and PubMedBERT. During this search, we stumbled upon the study by Muizelaar et al. (Muizelaar et al., 2024a), on the creation of a language model for Dutch electronic health records.

The research by Muizelaar et al. (Muizelaar et al., 2024a) indicates the feasibility of deep BERT models on the task of patient lifestyle classification specifically, and also shows promise for other types of fine-tuning tasks. Our CHA_2DS_2–VASc extraction model for the `Jessa-CHADSVASc` datasets builds upon this pre-trained model. The MedRoBERTa.nl model was trained from scratch on approximately 10 million Dutch hospital notes from Amsterdam University Medical Centers, providing domain-specific language crucial for accurate clinical information extraction (Verkijk and Vossen, 2021).

While many existing studies that worked with BERT- and RoBERTa-based models utilized English medical language models with translated datasets, recent work by Muizelaar et al. (2024b) and Muizelaar et al. (2024a) specifically validate the effectiveness of deep BERT models for Dutch clinical text classification and extraction tasks, which supported our choice of

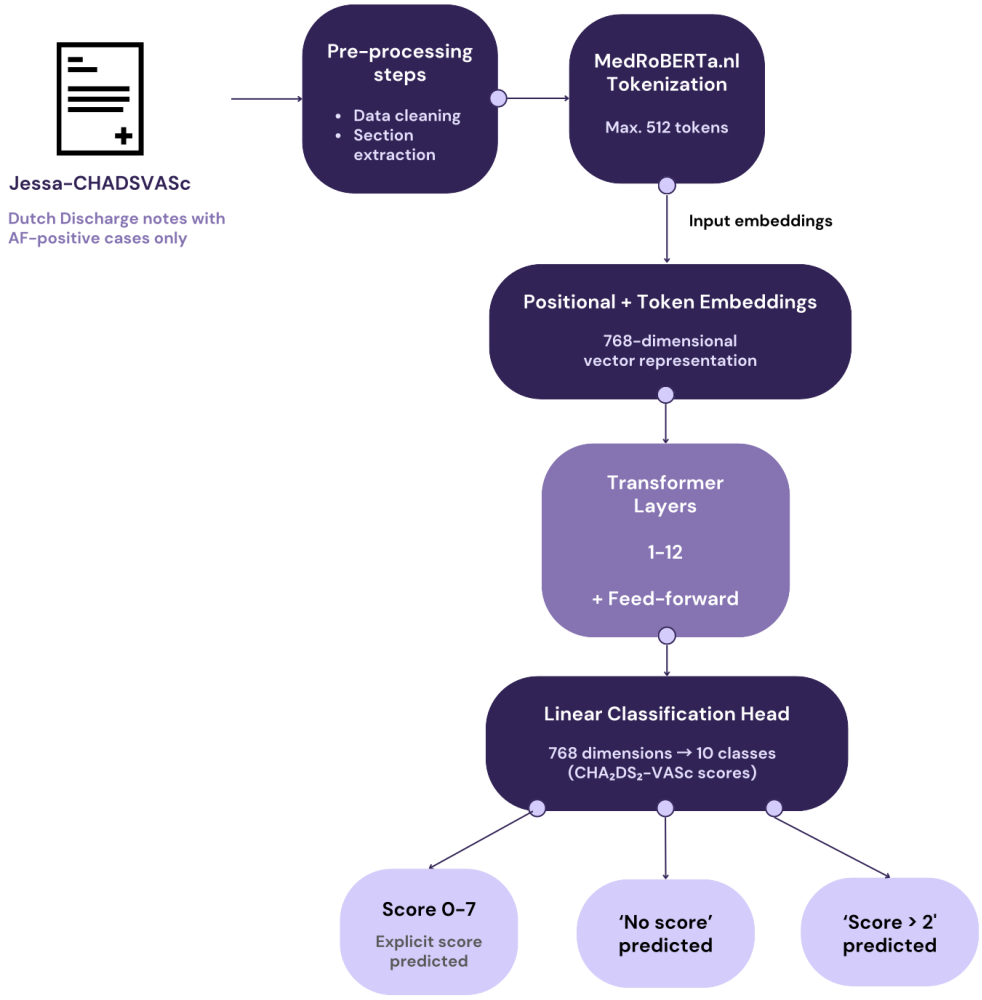

Figure 4.2: A diagram showing the different steps that our input data goes through in our CHA_2DS_2–VASc extraction and classification pipeline

MedRoBERTa.nl.

In figure 4.3 we zoom in on the Transformer layers as they would appear in our pipeline. The tokenized input gets converted into dense 768-dimensional vectors through two combined embedding types:

- **Token embeddings**: Each token ID maps to a learned 768-dimensional vector
- **Positional embeddings**: Since transformers do not inherently keep track of sequence order, positional embeddings encode the position of each token in the sequence

These are added together to create the input representations that capture both semantic meaning and positional information.

The model uses 12 transformer layers, each containing two main sub-components:

- Multi-Head Self-Attention
- Feed-Forward Networks

The multi-head self attention mechanism creates three vectors: Query (Q), Key (K), and Value (V) by multiplying the input with learned weight matrices, and computes attention scores by taking the dot product of queries with all keys. This can be written as:

$$Attention(Q, K, V) = softmax(QK^T/d_k)V$$

(Vaswani et al., 2023). This uses the softmax function, which is useful to convert raw attention scores into a probability distribution that sum to 1. When we compute QK^T/d_k, we get raw attention scores. The softmax function will normalize their weight by making all values positive and summing to 1, and each score represents how much attention to pay to each token, creating a robust attention mechanism that attaches a proper amount of weight to more important tokens. Scaling using the division by the square root of the key dimension (d_k) is used to balance results proportionally (Vaswani et al., 2023).

As explained earlier, the model uses multiple heads (12 in our case) to attend to difference relationships at the same time. Once the attention mechanism has aggregated contextual information for each token, the resulting representations are passed through a Position-wise Feed-Forward Network (FFN). This FFN is applied independently to each position in the sequence. In other words, every token embedding (up to 512 tokens per sequence) goes through the same feed-forward network with shared weights, but the computation is performed independently for each token, allowing for parallel processing. The feed-forward network does not capture relationship between positions, but rather handles non-linear transformations within each token's representation, which allows the model to build richer features at the token level. The feed-forward network has two linear transformations with a GELU activation (similar to ReLU) in between (Vaswani et al., 2023). The linear transformations are used to expand or reduce the dimensionality of the vectors. Concretely, in the case of RoBERTa (and MedRoBERTa.nl), each token representation starts as a 768-dimensional vector. The first linear transformation result in a vector of 3072 dimensions, by expanding our original vector by a factor of 4. The GELU activation is then applied to introduce non-linearity. This is what makes the layers so powerful, by providing a way to capture non-linear relationships within our embeddings.

The GELU activation function provides a gate-like mechanism, where small inputs get partially suppressed (not completely thrown out like ReLU would) and large inputs pass through easily. Negative inputs are not discarded, but are smoothed out. This is the main difference with the ReLU activation function, which would throw away negative values and possible get rid of important nuances. This is why RoBERTa models often use GELU instead of ReLU activation (Liu et al., 2019) (Verkijk and Vossen, 2021). For example, the input embedding for 'possible hypertension' might be weakly positive. While ReLU would eliminate this completely if it was negative, GELU preserves embeddings with low-confidence but potentially relevant evidence, enabling more nuanced medical language processing.

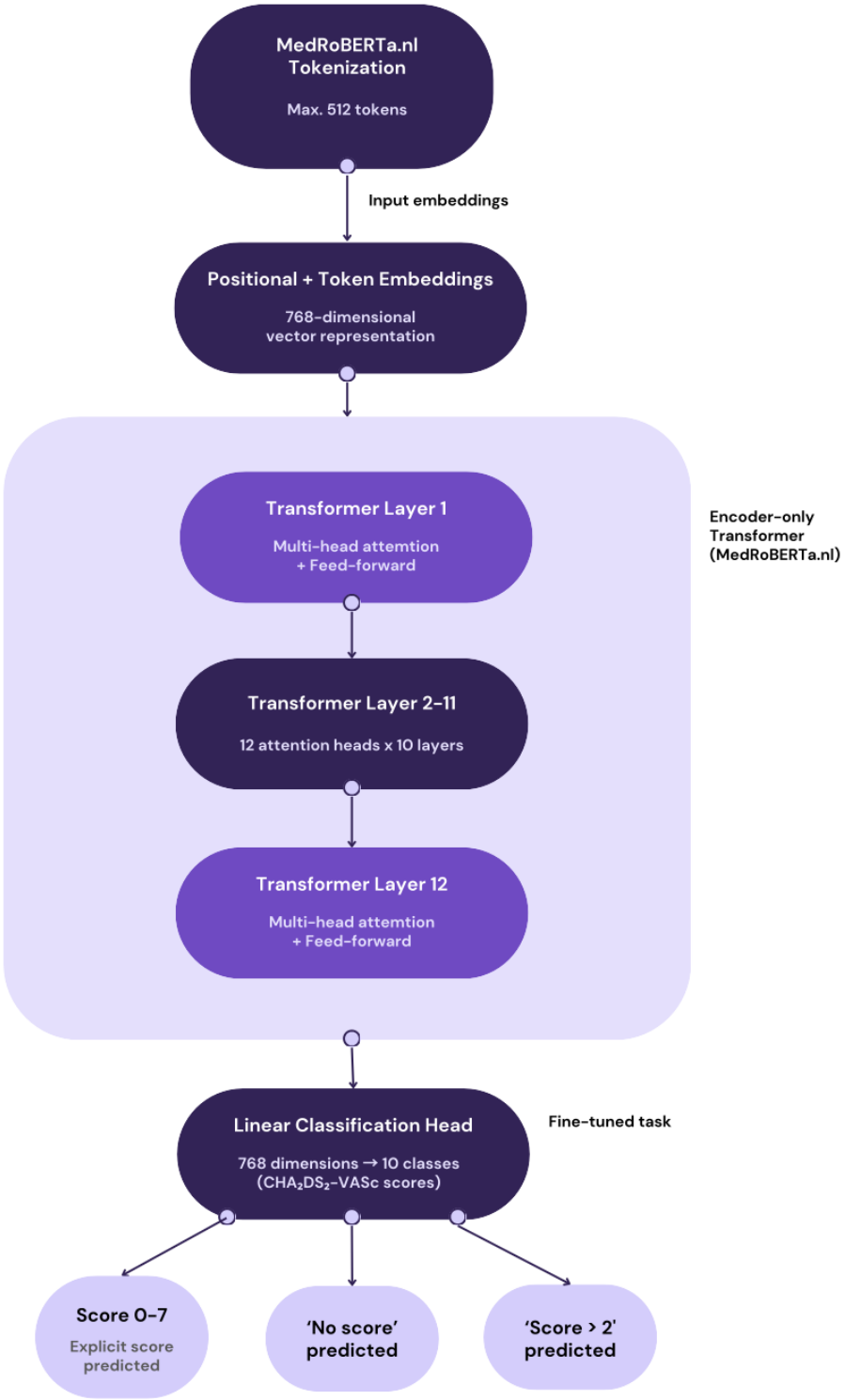

Figure 4.3: A diagram showing the different layers in our fine-tuned MedRoBERTa.nl approach and the expected output

After GELU activation is applied, the second linear transformation reduces the dimensionality back to 768, ensuring consistency for the following connections and layers.

The first layer is followed by 10 transformer layers with 12 attention heads each, capturing relationships within the text. Each subsequent layer builds on contextualized representations created by a previous layer, which gradually refines the model's understanding of the sequence of tokens.

Finally, the last transformer layer outputs highly contextualized embeddings for all tokens in the text. The model then extracts the representation associated with the special `[CLS]` token, which serves as a learned summary of the entire input sequence. This vector is then passed as an input to the fine-tuning task, which in our case is the linear classification head. The linear classification head maps the embeddings into logits, a term used to describe raw outputs of a neural network without any functions such as softmax applied to it. In our case, the classification head produces logits across ten possible CHA_2DS_2–VASc score categories (0-7, 'no score', or '>2). Then, the softmax function is applied to normalize the logits into a probability distribution over all the possible classes. This gives us a 'confidence score' for the prediction and ensures we learn how much probability the model assigns to each prediction.

The training of the classification head is done by minimizing the loss function between the predicted probability distribution and the true label.

4.4 Text Pre-processing

For the CHA_2DS_2–VASc extraction task, we developed a specific pre-processing pipeline that prioritizes clinically relevant sections of the discharge notes, specific to the `Jessa-AF` dataset. The *'BESLUIT EN BESPREKING'* (Conclusion and Discussion) section was identified as the primary location of the CHA_2DS_2–VASc score, containing approximately 80% of explicit score mentions in our `Jessa-CHADSVASc` dataset. In our pre-processing, we employed regular expressions to identify relevant sections using various terminology patterns. When the target section was successfully found, we removed all content before that section header is encountered. For cases where the section identification failed, we had a fallback strategy where we extracted the *'MEDISCHE ANTECEDENTEN'* ('Medical History') section first and handled it the same way. In case neither of these sections was found, the first 512 tokens of the discharge note were used to maintain consistency.

All text processing that was done for the CHA_2DS_2–VASc extraction models utilized the tokenizer from the MedRoBERTa.nl model. We used a sequence length of 512 tokens and did not implement any sliding window approach. Example code showing how this was done can be found in figure 4.4. For longer documents, we prioritized content from the most relevant sections, as previously discussed, and tested if this approach would be sufficient for our extraction task.

It is important to note that any introduction meta-data was removed during pre-processing of our `Jessa-CHADSVASc` datasets. By introduction meta-data, we primary mean certain repetitive phrases that were used to introduce a patient, or imply the fact they were admitted to the cardiology department. This was mostly to normalize our training data and make sure that repetitive patterns not relating to any useful clinical information were removed beforehand.

4.5 Implementation Details

We used the Transformers library provided by HuggingFace (Wolf et al., 2020) to fine-tune the MedRoBERTa.nl model. The process of fine-tuning is described by HuggingFace as adapting a pretrained model to a specific task, with a smaller specialized dataset (Wolf et al., 2020). Logically, Wolf et al. support the argument that this approach requires far less data and computations compared to training a model from scratch. The HuggingFace Datasets library

```
from transformers import AutoTokenizer

# Retrieve tokenizer from MedRoBERTa.nl
model_name = "CLTL/MedRoBERTa.nl"
tokenizer = AutoTokenizer.from_pretrained(model_name)

def prepare_dataset_for_model(df, text_col, label_col):
    """
    Convert raw text + labels into tokenized dataset ready for
    training
    """

    # Tokenize all texts at once
    encodings = tokenizer(
        df[text_col].tolist(), # List of discharge note texts
        truncation=True,       # Truncate text after max_length
        padding=True,          # Pad shorter texts to same length
        max_length=512,        # Standard BERT/RoBERTa sequence
    length
        return_tensors='pt'    # Return PyTorch tensors
    )

    # Create HuggingFace Dataset object for future use
    dataset = Dataset.from_dict({
        'input_ids': encodings['input_ids'], # Tokenized text
        'attention_mask': encodings['attention_mask'],
        'labels': df[label_col].tolist() # CHADS-VASC scores
    })
```

Figure 4.4: Code snippet displaying the tokenization of the data before training

(Lhoest et al., 2021) was used to create and handle the datasets used during the training process and facilitate sharing and rebuilding our implementations.

Our implementation utilizes MedRoBERTa.nl as the base model with the following specifications:

- RoBERTa-based Masked Language Model, with 12 transformer layers, 768 hidden dimensions and 12 attention heads
- Vocabulary size of 52,000 subword tokens optimized for Dutch medical text
- Pre-trained on 2.4 million Dutch hospital notes (2.5 GB of text data) from Amsterdam UMC
- Maximum sequence length of 512 tokens (limitation of absolute positional embeddings, and inherent to RoBERTa-based models)
- Fine-tuned for classification of CHA2DS2–VASc scores. This means we added a linear classification head that transforms the 768-dimensional token representation to our 14-class or 17-class output space (depending on the dataset that was used).

The domain-specific pre-training approach enables MedRoBERTa.nl to internalize Dutch medical terminology, abbreviations, and clinical language patterns that would be poorly represented in other general language models.

The original MedRoBERTa.nl research (Verkijk and Vossen, 2021) demonstrated that training from scratch on domain-specific data produces better semantic understanding than adapting general models.

Our implementation employed PyTorch (Paszke et al., 2019) and HuggingFace's Transformers library (Wolf et al., 2020) to create a sequence classification model for multi-label CHA_2DS_2–VASc score classification. The model configuration is shown in figure 4.5.

By using the AutoModelForSequenceClassification functionality provided by the Transformers library, we do not have to implement the architecture-specific details manually. Behind the scenes, this will add a linear classification head to the existing model, which maps the output of the model's hidden layers to our pre-defined number of classes.

4.6 Jessa-CHADSVASc Model Architecture

The 52,000 subword vocabulary of MedRoBERTa.nl includes medical-specific tokens that enable better processing of Dutch clinical text. Unlike general Dutch models, MedRoBERTa.nl should be able to better handle medical compounds like 'atrial fibrillation' as coherent units rather than slitting them into non-medical subwords or sub-tokens.

4.6.1 Masked Language Modeling

Masked Language Modeling (MLM) is the primary objective in RoBERTa-based models. The goal is to develop deep bidirectional representations. As described in the original paper that introduced BERT (Devlin et al., 2019), a masked language model will randomly mask a number of input tokens, and will then try to predict the original representation of the masked word in the vocabulary, based on the surrounding context. Unlike other left-to-right language model pre-training implementations, MLM allows us to use both the left and right context to make predictions and to pretrain a deep bidirectional Transformer Devlin et al. (2019).

However, Devlin et al. note that by masking certain terms in each training step (epoch) by replacing them with the '[MASK]' token, we are creating a mismatch between pre-training and fine-tuning, as the '[MASK]' token will not be present in the fine-tuning step of the model. To mitigate this, Devlin et al. made sure that the words that will be masked are randomly chosen and masked as follows. About 15% of the words in a sentence are randomly chosen. Then, 80%

```
# Define label mappings for CHA2DS2-VASc scores
# These labels represent the labels used for the
# Jessa-CHADSVASc-10 dataset.
labels_to_score_dict = {
    0: 0,   # Score 0
    1: 1,   # Score 1
    2: 2,   # Score 2
    3: 3,   # Score 3
    4: 4,   # Score 4
    5: 5,   # Score 5
    6: 6,   # Score 6
    7: 7,   # Score 7
    10: 'no_score',        # No score mentioned
    13: 'greater_than_2',  # Score > 2 (clinical threshold)
}

# Create bidirectional mappings
label2id = {str(v): k for k, v in labels_to_score_dict.items()}
id2label = {k: str(v) for k, v in labels_to_score_dict.items()}

from transformers import AutoModelForSequenceClassification,
   AutoTokenizer

# Load pre-trained MedRoBERTa.nl model for sequence classification
model = AutoModelForSequenceClassification.from_pretrained(
    "CLTL/MedRoBERTa.nl",
    num_labels=14,  # Specific to the dataset used
    label2id=label2id,
    id2label=id2label
)

tokenizer = AutoTokenizer.from_pretrained("CLTL/MedRoBERTa.nl")
```

Figure 4.5: Model Configuration and Initialization

of those are replaced with the special '[MASK]' token, 10% are swapped with random words, and the last 10% are left as they are (Devlin et al., 2019). For medical contexts, this kind of training is especially helpful as it allows the model to get familiar with medical terms, common abbreviations, and the way clinical information is usually written.

RoBERTa's masking approach is also dynamic. This implies that for each training epoch (cycle), a new masking pattern is created and used, instead of the static masks that are created during preprocessing in the case of BERT-based models. This approach will make sure the model encounters a larger variety of token masking combinations. Clinical abbreviations like 'AF' or 'afib' for Atrial Fibrillation and 'DM' for diabetes mellitus can be present inside of multiple or even a single discharge note, and require the model to have some understanding of the clinical context for accurate prediction.

4.6.2 Multi-Head Attention & Hidden state

One of the most important features of the architectures we are using is the attention mechanism. MedRoBERTa.nl, the model we are fine-tuning for our Jessa-CHADSVASc model, makes use of 12 attention heads per layer. These attention heads enable transformer models to focus on different parts of representations at the same time. Each attention head learns to capture distinct linguistic patterns. This parallel processing is useful in the case of CHA_2DS_2–VASc score extraction, where it is a possibility for the model to simultaneously identify risk factors, and understand their clinical significance. However, in our case, due to the nature of the available data and the pre-processing we have done, we hypothesized that our proposed implementation would result in more of a simple extraction task and any clinical understanding would be limited. This will be discussed further in the Discussion section of this thesis.

Each of the 12 transformer layers that MedRoBERTa.nl has, has 12 attention heads on its own. This means we have 144 total attention mechanisms that can process different aspects of medical documentation present in our discharge notes.

4.6.3 Hidden state layers

When our data passes through the model. the layers of the model progressively refine the representations of our data. Lower layers capture surface-level features (word boundaries, basic syntax), middle layers encode semantic relationships (medical entity recognition, negation scope), while upper layers integrate contextual information for task-specific decisions (Tenney et al., 2019). This hierarchical processing of the data enables the model to understand both medical terminology and global document structure.

4.6.4 Positional Embeddings

RoBERTa uses learned absolute positional embeddings. While attention mechanisms are useful, they do not look at the order of tokens in a document like Recurrent Neural Networks or Convolutional Neural Networks would. This is where the positional embeddings come into play. While different types of positional embeddings exists, RoBERTa uses the absolute type. These embeddings are created by assigning unique position-specific vectors to each token position up to the maximum sequence length, which in our case is 512 tokens, as we are limited by the pretrained model's architecture. These embeddings are added directly to the input embeddings at the bottoms of the encoder and decoder stacks (Wolf et al., 2020), before being processed by any transformer layers. To implement such positional encoding, we could use one-hot-encoding to embed the position of each token in a 512 vector, with all points being set to 0, except the actual tokens position, which would get a 1. Positional embeddings, since they have the same dimension as the input vector, can then be summed with the input embeddings. However, positional embeddings are often implemented in a smarter way, such as with embedding vocabulary lookup tables, or using sine and cosine functions as discussed in the foundational paper for the Transformer architecture (Wolf et al., 2020).

```
tokenizer = AutoTokenizer.from_pretrained("CLTL/MedRoBERTa.nl")
sentence = "Patiënt meldt zich met voorkamerfibrillatie.
CHADS-VASc van 4 is vastgesteld."

# Tokens
tokens = tokenizer.tokenize(sentence)

# Input ID's
input_ids = tokenizer.encode(sentence, add_special_tokens=True)

model = AutoModelForSequenceClassification.from_pretrained(
    model_name)
input_tensor = torch.tensor([input_ids]) # Vector with 23 points
position_tensor = torch.tensor([position_ids]) # Vector with 23
    points

# Get the embeddings layer that is used internally
embeddings_layer = model.roberta.embeddings

# Get the position embeddings
with torch.no_grad():
    position_embeddings = embeddings_layer.position_embeddings(
    position_tensor)
```

Figure 4.6: Caption

There are learned and fixed absolute positional embeddings. As mentioned before, RoBERTa uses the learned approach. This indicates the positional embeddings are randomly initialized and are later 'learned' through the training process (Wolf et al., 2020) (Liu et al., 2019), instead of being pre-defined such as the one-hot-encoding or the sine and cosine implementation.

For example, we look at the sentence *'Patiënt meldt zich met voorkamerfibrillatie. CHADS-VASc van 4 is vastgesteld.'* ('Patient presents with atrial fibrillation. CHADS-VASc of 4 is established.'). We use our tokenizer that was pre-initialized and pre-trained from MedRoBERTa.nl to tokenize our sentence. We can find the example code in figure 4.6. The generated tokens from this sentence will be the following: `['Patiënt', 'meldt', 'zich', 'met', 'voor', 'kamer', 'fibrill', 'atie', '.', 'CHADS', '-', 'VASc', 'van', '4', 'is', 'vastgesteld', '.']`.

Each position in the sequence receives a unique learned embedding vector. Table 4.1 shows the first 8 dimensions of each positional embedding for our example. We notice that, for example, at position 10 and 21, the same token `'.'` is used, but it is embedded very differently. This is of course important, as the order of the tokens matter, especially in our case, where we wish to extract a specific score from the discharge notes.

Position	Embedding Vector	Token
0	[0.021, -0.003, -0.006, -0.005, 0.026, -0.008, 0.029, -0.002]	`<s>`
1	[-0.034, -0.012, -0.021, -0.025, -0.005, 0.010, -0.004, 0.007]	`Patient`
2	[-0.007, -0.002, -0.050, -0.006, 0.023, -0.024, 0.006, 0.050]	`present`
3	[-0.038, -0.023, 0.010, 0.039, 0.028, 0.046, 0.020, 0.005]	`s`
4	[-0.017, -0.039, 0.014, -0.017, 0.010, -0.016, -0.005, 0.010]	`with`
5	[0.003, -0.031, 0.028, 0.002, 0.010, 0.019, -0.030, 0.015]	`at`
6	[0.005, -0.026, 0.009, -0.032, -0.008, -0.029, 0.006, -0.005]	`rial`

Table 4.1 continued from previous page

Position	Embedding Vector	Token
7	[0.015, -0.024, 0.023, -0.035, -0.003, 0.012, -0.020, 0.026]	fib
8	[-0.005, -0.036, 0.022, -0.019, 0.028, -0.002, 0.029, 0.001]	rill
9	[-0.006, -0.020, 0.011, -0.010, 0.061, 0.019, -0.035, 0.020]	ation
10	[-0.014, -0.007, 0.009, -0.002, 0.044, -0.010, 0.033, -0.001]	.
11	[-0.024, 0.010, 0.000, 0.004, 0.009, 0.004, -0.018, 0.034]	CHADS
12	[-0.034, -0.010, -0.024, 0.017, 0.038, -0.046, 0.049, -0.005]	-
13	[-0.025, 0.027, 0.011, 0.011, 0.023, -0.013, 0.027, 0.005]	VASc
14	[-0.044, -0.032, -0.007, 0.012, 0.023, 0.050, 0.031, 0.005]	of
15	[-0.006, 0.026, -0.018, -0.013, -0.003, 0.024, -0.015, 0.032]	4
16	[-0.002, -0.015, 0.003, 0.008, 0.022, 0.047, 0.027, 0.030]	is
17	[-0.003, 0.028, 0.005, -0.024, -0.006, -0.005, 0.002, 0.012]	e
18	[-0.006, -0.010, 0.002, -0.028, 0.041, -0.016, 0.013, 0.017]	stab
19	[-0.041, 0.008, -0.002, -0.051, -0.013, 0.000, 0.033, -0.007]	lis
20	[-0.012, 0.004, 0.020, -0.007, 0.028, 0.003, 0.005, 0.035]	hed
21	[-0.044, 0.000, -0.002, -0.009, 0.002, 0.025, 0.023, 0.024]	.
22	[-0.006, 0.025, 0.006, -0.004, -0.003, -0.006, 0.001, 0.034]	</s>

Table 4.1: Positional embeddings for each token position (first 8 dimensions)

Hyperparameter Search To find the optimal configuration for the CHA_2DS_2–VASc classification model, we implemented an automated hyperparameter search using Optuna (Akiba et al., 2019), a Python optimization framework. Rather than manually testing different combinations of hyperparameters, we looked for an approach similar to our hyperparameter search in the AF classification model using XGBoost. The approach using Optuna works similarly to the approach with PyTorch in the case of XGBoost, as it systematically explores the search space to maximize the model's performance, given a pre-defined evaluation metric.

The hyperparameter search was integrated with Hugging Face Transformers' built-in optimization functionality. The search space covered six key hyperparameters:

- Learning rate
- Batch size: Tested categorical options of 2, 4, and 8 for both training and evaluation, considering lower batch sizes performed better in tests
- Number of epochs: Varied from 1 to 10, as more epochs seemed to overfit model quickly
- Weight decay: Used to further fine-tune the model after the previous ones were established
- Warmup steps: Tested to see influence on model performance for gradual learning rate scheduling

Optuna uses an estimator algorithm, which builds probabilistic models of the objective function based on previous trials (Akiba et al., 2019). This optimization approach should be more efficient than random search, as it learns from earlier results to guide future hyperparameter selection. In contrast to the successive halving we used for XGBoost optimization, the algorithm Optuna allows for more variety in combinations of hyperparameters. The search was configured to maximize the F1-score, to once again get a balanced evaluation metric that balances both precision and recall.

The final hyperparameters that were used to create the final model are shown in figure 4.7.

The number of epochs was carefully selected after testing, where overfitting was happening after three epochs. The other hyperparameters were then experimented with further. The higher weight decay value of 0.2057 shows that strong L2-regularization helps prevent overfitting on this dataset.

Hyperparameter	**Value**
Learning rate	2e-5 (linear decay schedule)
Batch size	4 (per device, so 8 in total)
Training epochs	3 (early stopping on validation loss)
Weight decay	0.2057
Warmup steps	458 (almost 1 full epoch)

Figure 4.7: Hyperparameters used in the experiment for the CHA_2DS_2–VASc extraction model

By default, the AdamW (Adam with decoupled Weight decay) optimizer is used. This optimizer is widely used for training transformer models and is an improved version of the standard Adam optimizer, where weight decay is applied directly to the weights and not to the gradients. It is also the default optimizer when using the `Trainer` and `TrainingArguments` interface through the Transformers library (Vaswani et al., 2023).

The learning rate warmup steps will allow us to gradually increase the learning rate from 0 to our target learning rate over the specified number of steps. This is mainly used to prevent loss spikes early in training, where the model is still unstable.

4.7 Model Evaluation Metrics

We evaluated model performance using several complementary metrics appropriate for multi-label classification. Precision, Recall, and F1-Score were calculated both per label and macro-averaged across both classes. These metrics are crucial for clinical applications, where both false positives and false negatives have significant implications. The same arguments used for the AF classification task can be used here, as the data behind the model remains mostly the same.

Chapter 5

Results

In this chapter, we discuss the experimental results of our methodologies applied to the previously discussed datasets, both for atrial fibrillation classification and CHA_2DS_2–VASc score extraction from hospital discharge notes. The results will be discussed in three main sections: Atrial fibrillation classification, CHA_2DS_2–VASc score extraction and quality indicator analysis for AF patients.

5.1 Atrial Fibrillation Classification

5.1.1 Baseline Model Performance

MIMIC-IV-Ext-Cardio Dataset Results
Our XGBoost classifier combined with TF-IDF vectors performed well on the `MIMIC-IV-Ext-Cardio` dataset, with an F1-score of 0.90 for the positive class (AF) and an F1-score of 0.96 for the negative class (non-AF). This brings the model to a weighted F1-score of 0.95, as seen in table 5.1. The F1-score was chosen as the most meaningful evaluation metric for the performance of our models, given both the clinical context and the importance to capture as little false positives and false negatives as possible.

The F1-score is typically calculated for each class individually and then averaged to provide an overview of the model's overall performance. F1-scores can be both macro-averaged or weighted on each class. Macro-averaging computes the unweighted mean of F1-scores across all classes, treating each class with the same importance regardless of how many samples it contains. Weighted averaging, in contrast, assigns importance to each class's F1-score, weighted to its prevalence (support) in the dataset.

As we often work with a very imbalanced dataset, with much more non-AF cases than AF-cases, the weighted F1-score provides the most clinically relevant performance measure. However, the macro-average is also shown, as this may show a more realistic view for our use case, where a class imbalance will always be likely due to the prevalance of AF patients within cardiology departments.

We also report individual F1-scores to ensure we can discuss performance on both positive and negative AF class.

The resulting optimized XGBoost model achieved 95% overall accuracy on the cardiology department discharge notes from the test set, but in our case, looking at the F1 score gives a better overall view of the performance of the model.

The high precision (0.96) for non-AF cases indicates that there are minimal false positives. The strong recall (0.94) for AF cases in its turn also indicates an effective identification of true AF patients. Both metrics result in a well-balanced F1-score for both the non-AF class

Class	Precision	Recall	F1-Score	Support
Non-AF	0.98	0.95	0.96	6,216
AF	0.87	0.94	0.90	2,279
Overall Accuracy			**0.95**	**8,495**
Macro avg	0.92	0.94	0.93	8,495
Weighted avg	0.95	0.95	0.95	8,495

Table 5.1: A table showing the results of the performance metrics for the XGBoost + TF-IDF AF classification model on the MIMIC-IV-Ext-Cardio dataset

Class	Precision	Recall	F1-Score	Support
Non-AF	0.95	0.86	0.90	100,300
AF	0.65	0.85	0.74	30,580
Overall Accuracy			**0.86**	**130,880**
Macro avg	0.80	0.86	0.82	130,880
Weighted avg	0.88	0.86	0.86	130,880

Table 5.2: A table showing the results of the performance metrics for the XGBoost + TF-IDF AF classification model on the MIMIC-IV-Ext-No-Cardio dataset

(0.96) and the AF class (0.90). However, we do notice a reduction in precision for the AF class, with a precision of 0.87 compared to the non-AF class precision of 0.98. This indicates that while the model excels at correctly identifying non-AF cases, it produces more false positives when predicting AF, meaning some non-AF patients are incorrectly classified as having atrial fibrillation.

To evaluate the robustness across different clinical contexts for the optimized XGBoost classifier, we tested it on the MIMIC-IV-Ext-No-Cardio test dataset. This test set contained 130,880 records (significantly more than our cardiology-only dataset), with approximately 30% AF cases (30,580) and 70% non-AF cases (100,300).

In table 5.2, we can see the performance metrics measured on the predictions of the MIMIC-IV-Ext-No-Cardio test dataset. The F1-score for the AF class dropped substantially from 0.90 to 0.74, and AF precision declined significantly from 0.87 to 0.65. This indicates false positives increase when using our model in a context it was not trained for. However, we notice that AF recall had less of a decrease (from 0.94 to 0.85), indicating the model captures a lot of true AF cases. In clinical contexts, it is of course better to have a higher recall, where it may be better to over-diagnose than to under-diagnose. In our case, the model would primarily used for reporting purposes, but the argument still holds.

Overall, these results indicate the model works well on the cardiology data, and will most likely perform well in a similar context. These findings highlight the importance of domain-specific training data and suggest that broader generalization may need more training data from various other clinical contexts or a more robust feature engineering approach.

Jessa Hospital Dataset Results

Applying the same XGBoost and TF-IDF methodology to the Dutch hospital discharge notes from the Jessa Hospital dataset, resulted in the performance metrics shown in table 5.3. While the model achieved a high overall accuracy of 93%, the AF F1-score of 0.71 reveals more nuanced performance challenges when compared to the English datasets. The model demonstrates strong performance on non-AF cases with an F1-score of 0.96, but shows concerning limitations for AF detection:

With an AF F1-score of 0.71, it is substantially lower than the F1-score we reached for the model trained on the `MIMIC-IV-Ext-Cardio` dataset. The precision of 0.71 and recall of 0.72 for the AF class create a F1 score of 0.71 for the AF class, while the F1-score is 0.96 for the

Class	Precision	Recall	F1-Score	Support
Non-AF	0.96	0.96	0.96	2,233
AF	0.71	0.72	0.71	306
Overall Accuracy			**0.93**	**2,539**
Macro avg	0.84	0.84	0.84	2,539
Weighted avg	0.93	0.93	0.93	2,539

Table 5.3: A table showing the results of the performance metrics for the XGBoost + TF-IDF AF classification model on the Jessa Hospital dataset

Class	Precision	Recall	F1-Score	Support
Non-AF	0.97	0.97	0.97	2,233
AF	0.76	0.79	0.78	306
Overall Accuracy			**0.94**	**2,539**
Macro avg	0.87	0.88	0.87	2,539
Weighted avg	0.95	0.94	0.95	2,539

Table 5.4: A table showing the results of the performance metrics for the XGBoost + TF-IDF AF classification model on the Jessa Hospital dataset using the enhanced n-gram approach with n-range=(1,3)

negative class. This indicates the model performs well in detecting cases not related to AF, but struggles to correctly identify Atrial Fibrillation (AF) samples well. The recall of 0.72 means the model fails to identify a significant number of actual AF cases, which is dissapointing.

Interestingly, the Jessa model shows more balanced precision and recall (0.71 and 0.72) compared to the MIMIC model, which typically had higher precision than recall. This suggests the model struggles equally with both false positives and false negatives when applied to Dutch clinical text. The maintained high weighted F1-score (0.93) and accuracy (93%) are largely driven by good performance on the majority non-AF class, and is a good indication of why class-specific F1-scores provide more meaningful insight into model quality for our application.

We will discuss possible causes for these performance differences in the discussion chapter of this thesis.

5.1.2 Larger N-gram Approach

Our initial analysis of the results revealed that our unigram-based model struggled with negation patterns that are common in the discharge notes, such as mentions of '*geen vkf*' ('no af') and '*uitgesloten voorkamerfibrillatie*' ('excluded atrial fibrillation'). To address this limitation of our model, we implemented an enhanced approach with larger n-grams. Instead of capturing unigrams, which only capture one term at a time, we experimented with an n-gram range of 1 to 3, so we could capture important features in unigrams, bigrams and trigrams.

For this model, the only element we changed in our approach was the size of the n-grams. The hyperparameter search was performed again for this approach, and the model was trained on the `Jessa-AF` dataset again.

This approach seemed to already yield better results, as can be seen in table 5.4.

The model seemed to better capture negations, where this time *'vkf'* ('af'), '*ablatie*' ('ablation'), '`ablatie uitgevoerd`' ('ablation performed'), '*snel ventriculair*' ('fast ventricular'), '*cardioversie uitgevoerd*' ('cardioversion performed'), '*voorkamerfibrillatie*' ('atrial fibrillation'), '*lixiana*' (antigoagulant medication), were highly important features.

Most terms directly indicate the presence of AF, while others indicate a common treatment for patients with AF, such as catheter ablation or anticoagulant medications such as Lixiana.

Model	Test Dataset	Test size	AF test cases	Language	Dept.
MIMIC	MIMIC-IV-Ext-Cardio	8,495	2,279 (27%)	English	Cardiology
MIMIC	MIMIC-IV-Ext-No-Cardio	130,880	30,590 (23%)	English	All Depts
Jessa	Jessa Hospital	2,539	306 (12%)	Dutch	Cardiology

Table 5.5: Test set characteristics for the three different experiments including the MIMIC-IV-Ext-Cardio and the Jessa Hospital model. MIMIC = MIMIC-IV-Ext-Cardio model, Jessa = Jessa Hospital model

Model	Accuracy	AF Precision	AF Recall	AF F1
MIMIC-IV-Ext-Cardio	0.95	0.87	0.94	0.90
MIMIC-IV-Ext-Cardio (On Non-Cardio)	0.86	0.65	0.85	0.74
Jessa Hospital	0.94	0.78	0.65	0.71

Table 5.6: Performance results for the positive AF class for the three different test sets including the MIMIC-IV-Ext-Cardio and the Jessa Hospital model.

5.1.3 FastText Embeddings Exploration

During the development phase using the MIMIC-IV data, we investigated the potential of FastText embeddings as an alternative to TF-IDF vectorization. We hypothesized that the introduction of word embeddings would positively influence the performance of our XGBoost model. However, our experiments revealed that our approach using the FastText embeddings did not outperform the traditional TF-IDF methodology, and took up too much time to our liking.

Given the better performance of TF-IDF combined with XGBoost on the MIMIC-IV dataset, we proceeded with this approach for the Jessa Hospital experiments, rather than investing more time and computational resources in the FastText optimization.

5.1.4 Result Analysis

Comparing the performance of both classification models across the different datasets reveals some notable insights. Table 5.6 shows the performance metrics for the positive class for both models, once for the Jessa model and twice for the `MIMIC-IV-Ext-Cardio` model: Once for both the regular test set and once for the second test set with notes from outside the cardiology department.

We notice here that the cardiology-focused datasets yield better AF detection performance regardless of the size of the dataset. Both models trained on `MIMIC-IV-Ext-Cardio` and the Jessa hospital dataset yield better precision for the positive class. However, for the Jessa hospital, the recall appears to be lower than both other models. The Jessa dataset of course has fewer samples (2.539 total records in the test set with only 12% being positives). These results will again be discussed in more detail in chapter 6 of this thesis.

We notice that for the Jessa model's results, the recall for the positive class (0.74) is actually quite similar to that of the `MIMIC-IV-Ext-Cardio` model tested on the MIMIC-IV-Ext-No-Cardio test set (0.71).

In our first hypothesis (H1), we predicted that XGBoost with TF-IDF could achieve >90% accuracy for AF identification. Our results partially support this hypothesis:

Metric	Score
Overall Accuracy	0.95
Macro F1-Score	0.82
Weighted F1-Score	0.95

Table 5.7: Table showing the overall evaluation metrics on the final CHA_2DS_2–VASc extraction model, trained on `Jessa-CHADSVASc-10`

- Our implementation on `MIMIC-IV-Ext-Cardio` achieved 95% accuracy.
- Our implementation on `Jessa-AF` achieved 93% accuracy.

Our hypothesis therefore is confirmed. However, AF-specific F1-scores were more variable (0.90 vs 0.71), and provide more insight into the actual performance of the mode, suggesting language and dataset-specific challenges are still at play and should be looked into further.

5.2 CHA_2DS_2–VASc Score Extraction

5.2.1 Model Performance

For CHA_2DS_2–VASc score extraction from the Dutch hospital discharge notes from the Jessa dataset, we fine-tuned the MedRoBERTa.nl model (Verkijk and Vossen, 2021). This approach was selected based on recommendations from the research team's prior experience, as well as prior research demonstrating superior performance of domain-specific and language-specific pre-trained models for clinical information extraction tasks as discussed in the Methodology section of this thesis.

Fine-tuning Results on Jessa-CHADSVASc-10
Our fine-tuned MedRoBERTa.nl model achieved promising results on the CHA2DS2–VASc score extraction task. The model was trained on the `Jessa-CHADSVASc-10` dataset, which contained 1,482 records with 10 different classes after removing records with extremely rare scores.

The optimized model achieved the performance metrics on the test set as presented in table 5.7.

The training process showed clear convergence over 3 epochs, with both training and validation loss decreasing steadily. The validation F1-score improved from 0.40 in epoch 1 to 0.95 in epoch 3, while validation accuracy increased from 0.55 to 0.95, demonstrating effective learning without overfitting.

As presented in table 5.8, the model demonstrated strong performance across all classes, with F1-scores consistently above 0.89 for most categories, except for the 'Greater than 2' class. This result was expected, as the training, validation and test data only contained very few samples for this class.

The model performed exceptionally well at identifying when no explicit CHA_2DS_2–VASc score was mentioned (F1-score: 0.967). This is useful in our research objective to identify quality indicators from discharge notes through modern Natural Language Processing techniques. The absence of a reported CHA_2DS_2–VASc score is an important metric. Most individual score classes achieved perfect F1-scores, partially due to the very low number of records in the test set, with only minor performance drops for score 3 (F1: 0.889) and the 'greater than 2' category, which obviously had insufficient training examples as its one test case got predicted as a score of '2' instead of 'greater than 2'.

CHA2DS2–VASc Score	F1-Score	Support
0	1.000	5
1	1.000	7
2	1.000	12
3	0.889	14
4	0.909	12
5	0.909	5
6	1.000	2
7	1.000	1
No score	0.967	44
Greater than 2	0.000	1

Table 5.8: Table showing the per-class evaluation metrics on the final CHA_2DS_2–VASc extraction model, trained on `Jessa-CHADSVASc-10`

5.2.2 Result Analysis

The results demonstrate that fine-tuning MedRoBERTa.nl for CHA_2DS_2–VASc score classification achieves good performance levels that could be clinically useful, and show promise for other possible fine-tuning tasks. The model's ability to achieve 95% overall accuracy and a 0.82 macro F1-score suggests our approach in fine-tuning a Dutch pre-trained model specific to clinical text shows promise for future further implementations.

We analyzed the misclassifications for our evaluation of the Jessa-CHADSVASc model. One case contained two different CHA_2DS_2–VASc scores mentioned in the same document (both 4 and 3), likely due to score recalculation during an extended hospital stay. The model predicted score 3 while the ground truth was 4. This explains the slight drop in F1-score. Another test case included a mentioned score of 'CHADSVASC >2' and was incorrectly predicted as 'no_score.' This error likely stems from limited training examples for inequality expressions and variations in Dutch phrasing in the training data (such as *'groter dan 2'* ('greater than 2'). The remaining errors in the test set occurred when the text was truncated at the 512-token limit, causing the actual score mention to be cut off. This represents a fundamental architectural limitation rather than a model learning issue. Earlier tests with a subset of the `Jessa-CHADSVASc` dataset that only included 512 tokens around the CHA_2DS_2–VASc score as input-data performed well but were not continued due to time constraints.

Apart the very few records that this impacted in our testset, our preprocessing approach, which prioritized the *'BESLUIT EN BESPREKING'* (Conclusion and Discussion) section of the discharge notes, proved highly effective.

From a clinical perspective, the model's overall performance is good. The ability to correctly identify explicit score mentions with >95% accuracy allows for more in the future. The model's few errors represent edge cases or limitations of our architectural design choices, and mostly still correctly identifies the presence of a score, given that it is in the input text.

H2 predicted >85% accuracy for CHA2DS2-VASc extraction using MedRoBERTa.nl. Results strongly support this hypothesis with 95% overall accuracy, demonstrating the effectiveness of domain-specific Dutch medical language models. Again, the F1-score provides more insight as an evaluation metric in our case. We note a weighted F1-score of 0.95 for our CHA_2DS_2–VASc classification model trained on `Jessa-CHADSVASc-10`.

5.3 Quality Indicator Analysis

Building on our AF classification and CHA_2DS_2–VASc extraction models, we analyzed adherence to ESC quality indicators for AF patients in the Jessa Hospital dataset.

5.3.1 Proportion of AF Patients with Reported CHADS-VASc Score

Using our combined pipeline of AF classification followed by CHA_2DS_2–VASc extraction, we analyzed documentation practices for the first quality indicator: the proportion of AF patients with a reported CHA_2DS_2–VASc score.

Of the 1,489 AF-positive cases identified in our `Jessa-CHADSVASc-10` dataset

- 662 cases (44.5%) had an explicit CHA_2DS_2–VASc score documented
- 827 cases (55.5%) had no explicit score mentioned in the discharge notes

This finding suggests that despite the guidelines put forward by the ESC, recommending CHA_2DS_2–VASc score calculation for all AF patients, reporting of these scores in discharge notes remains rather inconsistent.

The distribution of the various scores present in `Jessa-CHADSVASc` is shown in table 5.9, both as a percentage of the discharge notes that had a reported score, as well as the percentage in regards to the total dataset of patient diagnosed wit AF.

CHA2DS2–VASc Score	Count	Percentage of Reported	Percentage of Total AF
0	58	8.8%	3.9%
1	74	11.2%	5.0%
2	133	20.1%	8.9%
3	149	22.5%	10.0%
4	129	19.5%	8.7%
5	71	10.7%	4.8%
6	21	3.2%	1.4%
7	10	1.5%	0.7%
8	3	0.5%	0.2%
9	3	0.5%	0.2%
>2 (unspecified)	11	1.7%	0.7%

Table 5.9: A table showing the distribution of CHA_2DS_2–VASc scores across the `Jessa-CHADSVASc-10` dataset

The distribution shows a spread across different scores, with scores 2 to 4 being most common (combined they form 62.1% of total documented scores).

5.3.2 Proportion of AF Patients with Reported CHADS-VASc Score > 2

Among the 1,489 AF-positive cases, 26.7% of them have a reported CHA_2DS_2–VASc score strictly greater than two.

We identified 11 cases in our `Jessa-CHADSVASc-10` dataset where scores were documented using inequality expressions (e.g., 'CHADSVASC >2'). This represents a small but clinically significant subset where instead of noting exact scores, a typical threshold was used. Our

Score Category	Count	Percentage of Total AF	Percentage of Documented Scores
Score >2	397	26.7%	60.0%
Score = 2	133	8.9%	20.1%
Score <2	132	8.9%	19.9%

Table 5.10: A table listing the distribution of records with a CHA_2DS_2–VASc score of 2 or greater

approach for CHA_2DS_2–VASc extraction did not sufficiently test these cases due to the lack of adequate training data.

Chapter 6

Discussion

In this chapter, we analyze and interpret our research findings and examine the performance of the various approaches we researched for automated AF classification and CHA_2DS_2–VASc score extraction. We will discuss the further implications of our findings, acknowledge the limitations, and aim to situate our findings within a broader context of medical NLP research.

6.1 XGBoost as a Baseline Solution

Our results demonstrate that XGBoost in combination with TF-IDF features provides a good baseline for AF classification tasks. The strong performance that was achieved on the `MIMIC-IV-Ext-Cardio` dataset (95% accuracy, 0.95 weighted F1-score) confirms that traditional machine learning approaches, like the one we used, remain competitive for well-defined classification tasks in clinical text. This is supported by the research of Falter et al., which has found good results for the classification of diseases using ICD-10 codes using the same approach (Falter et al., 2024).

However, we noticed a strong performance drop in our model created for the `Jessa-AF` dataset.

6.1.1 Feature Importance

Analysis of the most predictive features for AF classification in the Jessa Hospital dataset reveals both the strengths and limitations of our TF-IDF approach. The top 5 features were:

- *snel* (rapid/fast), capturing symptomatic descriptions often related to patients with Atrial Fibrillation
- *vkf*, an abbreviation of Atrial Fibrillation in Dutch (*voorkamerfibrillatie*)
- *voorkamerfibrillatie* (Atrial Fibrillation)
- *cardiologie* (cardiology), indication of the related department, often mentioned in AF positive samples
- *novo*, related to frequent term 'AF de novo', indicating the presence of new-onset AF

These features demonstrate that the model effectively learned both explicit diagnostic terminology ('vkf', 'voorkamerfibrillatie') and clinical descriptors ('snel', 'novo'). However, they also reveal potential limitations. There seems to be a heavy reliance on explicit terminology, such as the bias towards mentions of 'cardiologie'. Not only does this influence the classification of Atrial Fibrillation (AF) cases, but we make the hypothesis that this would also reduce generalizability of the model outside of the current context.

While the term 'novo' may often refer to 'AF de novo' in our training set, this is not always the case, as the term can be used for other diagnoses as well. This may have played an important role in the reduced F1-score for AF positive cases.

6.1.2 TF-IDF Vectorization

The effectiveness of a more traditional approach with TF-IDF vectorization can be attributed to several factors. TF-IDF vectorization seems to clearly capture the importance of specific medical terminology related to AF, such as 'atrium', 'fibrillation', 'vkf' and 'novo', which proved to be highly predictive features in both of our AF classification models. The term 'novo' was mostly predictive in the case of our `Jessa-AF` dataset, as many discharge notes mentioned 'AF de novo', with 'de novo' indicating a new onset AF that was not previously diagnosed in the patient. We hypothesize this may have been a very predictive term due to the small number of samples in the Jessa dataset, indicating the model might lose some of its predictive power if used in a broader context, as this term might be used in combination with other diagnoses. This of course has to be thoroughly tested and researched. However, the only available Dutch discharge notes we had at our disposal were the ones in the Jessa Hospital dataset. This dataset only includes discharge notes from their cardiology department, which has already filtered out more noise and less relevant discharge notes from other departments.

The bag-of-words representation (the 'Term-Frequency' part of the TF-IDF method) using unigrams (n-gram with n=1), loses sequential information. For example, we may consider the following sentences:

1. 'Patient has no history of atrial fibrillation'
2. 'Patient has atrial fibrillation, no complications'

Our approach would consider most of the same key terms for both sentences: 'patient', 'atrial', 'fibrillation', and 'no', with 'history' and 'complications' being the outliers. However, all of these key terms appear exactly once in either sentence, indicating they are all equally important, which is not the case.

However, as we previously mentioned, the presence of certain diagnostic terms, regardless of order, look to be highly predictive in our experiments. This is where our traditional approach with XGBoost and TF-IDF may lack. While the use of bi- and trigrams improved our AF-classification model for the `Jessa-AF` dataset, it seems there are still some pre-processing steps that can be taken and tested to further improve our models. Some of these can include better negation handling and improving OOV-terms by using pre-trained embeddings specific for clinical data.

Further research is necessary to discover how well pre-trained embeddings would work with a traditional XGBoost classifier and if they have a significant influence on the performance of the model.

6.1.3 XGBoost

XGBoost's gradient boosting algorithm excels at handling high-dimensional and sparse feature spaces created by TF-IDF vectorization. The algorithm's ability to automatically select relevant features through its tree-building process contributed significantly to the strong results. In the case of our `Jessa-AF` classification model, '*vkf*' ('af') and other terms specific to the treatment of AF seem to be highly predictive. We notice that some form of understanding is needed to make proper nuances for when Atrial Fibrillation is mentioned in the discharge notes as part of medical history, and if it is no longer present. This is where our `Jessa-AF` model seemed to lack performance, as terms like '*vkf*' are mentioned often in these samples.

6.1.4 Enhanced N-gram Approach

The enhanced n-gram approach (n-range=(1,3)) showed meaningful improvements for the `Jessa-AF` dataset, with AF F1-score increasing from 0.71 to 0.78. This improvement can be attributed to better capture of contextual patterns and negation phrases that are crucial in Dutch discharge notes. We now notice the importance of capturing more than just the presence of certain terms in clinical text handling.

The most important features identified by the enhanced model included both direct diagnostic terms (often bigrams) and treatment options used in patient with AF, suggesting the model learned to associate AF not only with explicit mentions but also with common therapeutic interventions for AF patients.

The performance drop between our English (MIMIC) and Dutch (Jessa) datasets suggests some language-specific challenges might still come into play in clinical NLP, despite using similar methodologies. The `Jessa AF` dataset's smaller size (2,539 records vs 8,495 for `MIMIC-IV-Ext-Cardio`) likely contributed to reduced generalization capability and caused our model to overfit more to the small number of records it was trained on.

6.1.5 Embedding-Based Approaches

Our exploration of FastText embeddings as an alternative to TF-IDF vectorization revealed mixed results. While we hypothesized that word embeddings would better capture semantic relationships and handle Out-of-Vocabulary (OOV) words common in clinical text, our experiments showed that FastText did not outperform the traditional TF-IDF methodology on our datasets. The underwhelming performance is likely due to our implementation architecture, where we averaged FastText vectors to reduce dimensionality, and therefore losing much of the semantic information that makes embeddings valuable.

However, it still seemed FastText had many of the same OOV words as our traditional TF-IDF approach when it came to medical specific terms, indicating more domain-specific pre-trained embeddings would have been necessary to capture more relevant context. As this approach was quickly abandoned, no further arguments can be made as we had only tested our approach with averaged FastText vectors. We hypothesize different approaches in using FastText vectors in combination with XGBoost may yield better results.

For a simple binary classification task, XGBoost in combination with TF-IDF seems to perform well enough. Further research is needed to decide its relevance in multi-label classification, such as the automated classification of ICD-10 codes related to AF, which would require a more nuanced understanding of the types of AF a patient can present with.

6.2 Technical Achievements and Limitations

Our CHA_2DS_2–VASc score extraction model using fine-tuned MedRoBERTa.nl achieved strong performance (95% overall accuracy, 0.82 macro F1-score) on a rather small test set, demonstrating the value of domain-specific and language-specific pre-trained models for clinical information extraction tasks. Unlike traditional approaches such as our TF-IDF + XGBoost baseline for AF classification, transformers can capture more complex relationships within text, making them more suitable for tasks requiring contextual understanding. However, do to the nature of our dataset, as well as the small size, we cannot make significant claims on how well our model makes use of the clinical context. The CHA_2DS_2–VASc extraction task is a rather simple classification task, and our training data often had exact scores mentioned in the notes. We want to stress the importance that further testing is needed on different splits of the dataset, as well as test sets with more variety, such as only having the risk factors indicated but no actual mention of the score itself. We hypothesize our model will not perform well in that case, as it is fine-tuned to our training set.

Several technical limitations emerged during our implementation. The 512-token limitation inherent to the RoBERTa and BERT architecture presents a significant constraint for clinical documents, which can often exceed this length. We addressed this through prioritized text preprocessing, focusing on clinically relevant sections such as only using the *'BESLUIT EN BESPREKING'* (Conclusion and Discussion) section. While this approach has proven it sometimes fails in classifying a proper CHA_2DS_2–VASc score in longer documents, the impact on our dataset is rather small.

We could have addressed this limitation through a sliding window approach, processing documents in overlapping segments and perhaps aggregating predictions, but time constraints prevented implementation of other approaches.

6.2.1 Dutch Medical Text Challenges

Working with Dutch medical text presented unique challenges compared to working with English records. The scarcity of high-quality Dutch medical language models limited our options, though MedRoBERTa.nl proved to be an excellent choice for our CHA_2DS_2–VASc extraction task. The model's training on 2.4 million Dutch hospital notes provided the domain-specific vocabulary necessary for understanding clinical abbreviations and Dutch medical terminology.

6.2.2 Data Limitations

The significant class imbalance in our `Jessa-AF` dataset (88% AF-negative vs 12% AF-positive) poses challenges for model training and evaluation. While this imbalance seems clinically representative, it required careful consideration of evaluation metrics and training strategies. Our approach of using balanced training sets while maintaining realistic test set distributions helped us to prevent overfitting in our models, but we notice that performance could have been even better. More thorough research is needed to find out if more training data or significantly different splits would have made a difference for the models trained on our Jessa Hospital dataset.

The relatively small size of our Jessa Hospital dataset compared to MIMIC-IV limited our ability to train a more complex model. This constraint was particularly evident in our CHA_2DS_2–VASc extraction model, where several score categories had very few examples, resulting in limited training and test data for rare classes. However, the different experiments proved the feasibility of using modern NLP techniques to form an aid to hospital diagnosis reporting using modern LLMs that have been pre-trained on similar data (Dutch clinical text).

The manual annotation process for CHA_2DS_2–VASc scores on the `Jessa-AF` dataset revealed some inconsistencies. Some records contained contradictory score information or used various notation formats, which proposed challenges during the creation of the models.

6.2.3 Generalizability Considerations

In the case of our AF-classification model `MIMIC-IV-Ext-Cardio`, its performance varied significantly across the test set of `MIMIC-IV-Ext-Cardio` and the test set of `MIMIC-IV-Ext-No-Cardio`. The model showed reduced performance when applied to non-cardiology departments (accuracy dropped from 95% to 86%), highlighting the importance of domain-specific training data. This suggests that while our approaches work well within their intended clinical context, broader generalization may require additional training or a different approach altogether.

6.2.4 Interpretability

An important consideration for clincial use of the proposed models is model interpretability. Our XGBoost models provide feature importance and confidence scores that can help clinicians understand which terms drive AF classification decisions, and provide insight into the confidence

our models put behind a prediction. The transformer-based CHA_2DS_2–VASc extraction model, while it works with pre-trained embeddings and builds in some form of reasoning, still operates as a black box that is harder to interpret. Future work could incorporate attention visualization or other interpretability techniques to help better understand how the model arrives at its predictions.

6.3 Hypothesis Validation

6.3.1 Primary Hypotheses

Addressing RQ1 and H3 on ESC quality indicator measurement, our combined pipeline revealed only 44.5% of AF patients had documented CHA_2DS_2–VASc scores, while 26.7% had clinically significant scores (>2). Given the model we trained by fine-tuning the pre-trained MedRoBERTa.nl model, we believe it is feasible to use modern NLP techniques for (partially) automated measurement and reporting for guideline adherence. Using confidence scores, our models can indicate how certain it is about a given prediction, making it possible for automated tools to have a human-in-the-loop approach, where certain low-confidence predictions can still be manually checked, but significantly reduce the time needed in reporting. More extensive research is needed to see how these approaches would generalize and work in real-life clinical settings to confirm their performance.

6.3.2 Secondary Hypotheses

We hypothesized AF-classification models trained on cardiology department data would show reduced performance when applied to other departments (H4). This is confirmed by our results on the **MIMIC-IV** model. Performance degraded from 95% accuracy to 86% when applying cardiology-trained models to general hospital data, and confirms our hypothesis about domain-specific training importance. However, we attribute the loss in accuracy mostly due to the approaches used. As our models were trained on the `MIMIC-IV-Ext-Cardio` using unigrams, it was highly specialized towards our department-specific terminology, and only used unigrams, indicating a more thorough clinical understanding or reasoning system might be necessary to better capture certain nuances that are present in a broader scope, such as many different departments in a hospital.

While H5 was not directly tested by comparing general language models to our fine-tuned implementation of the pre-trained MedRoBERTa.nl model, we can provide some insight into the feasibility of creating a domain-specific task using a related pre-trained model. With our pre-trained approach, the lack of excellent performance for our CHA_2DS_2–VASc extraction model for Jessa can be mostly attributed to the heavy class imbalance and lack of samples for each class, given our linear classification head that was applied to the pre-trained model.

We hypothesize that models such as MedRoBERTa.nl will provide a better starting ground than any other general language models would, but identify that there is still more research to be done to truly extract its full potential, given our CHA_2DS_2–VASc extraction task. The multi-class classification approach we employed at first, might yield worse results than other fine-tuning tasks using MedRoBERTa.nl. We therefore note that other types of fine-tuning tasks should be tested before making any further conclusions.

6.4 Future Work for AF classification

Future research should explore approaches that incorporate word embeddings with XGBoost, rather than simple vector averaging as done in our FastText approach. Techniques that capture semantic information, as well as reducing OOV-terms for clinical texts, while maintaining the benefits of gradient boosting, seem to be the way forward.

Our current binary AF classification could be extended to multi-label classification, distinguishing between different types of AF (paroxysmal, persistent, permanent), which would provide more clinically relevant information for treatment decisions and closely relates to ICD-10 coding.

6.5 Future Work: Analysis of Anticoagulants

A natural extension of our work would be to analyze anticoagulant prescription patterns in relation to CHA_2DS_2–VASc scores. This could help assess the second ESC quality indicator: 'Proportion of AF patients with significant CHA_2DS_2–VASc score ($\geq$2) that received anticoagulant therapy.' This analysis would require additional natural language processing to extract medication information from discharge notes, and validate whether patients with a clinically significant score received the proposed treatment.

Chapter 7

Conclusion

This thesis provides insight into how Natural Language Processing techniques can aid in hospital reporting, particularly relating to guidelines set in place by the ESC in regards to patients diagnosed with Atrial Fibrillation.

7.1 Summary of Contributions

7.1.1 Technical Contributions

We developed a robust XGBoost classifier for detection of AF, utilizing the power of TF-IDF vectorization to achieve a model with 95% accuracy on English cardiology discharge notes from the `MIMIC-IV-Ext-Cardio` dataset, and a 93% accuracy on our `Jessa-AF` dataset with Dutch discharge notes. While accuracies seem excellent, we emphasized the importance of other evaluation metrics in clinical contexts as well as when working with a large class imbalance. The metric we often used was the F1-score, both weighted as well as calculated for each class seperately. With a weighted average F1-score of 0.95 for both the `MIMIC-IV-Ext-Cardio` model and the `Jessa-AF`, we can conclude Natural Language Processing (NLP) techniques, including traditional machine learning approaches, are effective and provide a good aid for hospital diagnosis reporting.

We successfully fine-tuned an existing model (MedRoBERTa.nl), pre-trained on a large dataset of Dutch clinical text, for our CHA_2DS_2–VASc score extraction task, achieving both an accuracy and a weighted F1-score of 95%. This represents one of the first implementations of a score extraction task using a pre-trained transformer-based model in Dutch healthcare settings. While the original study by Verkijk and Vossen (2021) tested the model's performance on a downstream task for classifying discharge notes across broad domains, we proved the feasibility of other downstream tasks using domain-specific pre-trained models, given differently structured Dutch data from a different hospital.

7.1.2 Research Contributions

Our analysis of the `Jessa-AF` and `Jessa-CHADSVASc` dataset revealed that only 44.5% of AF patients had a documented CHA_2DS_2–VASc score, highlighting the importance of NLP-based reporting aids.

We established a pre-processing pipeline specifically designed for handling Dutch medical text, including the analysis of clinical abbreviations and domain-specific terminology. We provided a baseline performance metric for AF-classification and CHA_2DS_2–VASc score extraction tasks that can serve as reference points for future research in clinical NLP applications, as well as

provide a good baseline for similar downstream tasks using language-specific pre-trained large language models.

7.2 Results

Our research addressed the primary research questions:

- RQ1: Modern NLP techniques can measure ESC quality indicators through discharge letters with good accuracy. Our combined pipeline identified AF patients and extracted CHA_2DS_2–VASc scores with sufficient accuracy. Given the models' lower confidence scores on wrong predictions, it is feasible to create a tool that could aid hospital reporting with a human-in-the-loop approach, where humans can manually check cases flagged as uncertain.
- RQ2: XGBoost with TF-IDF achieved an accuracy of >90% for the classification of AF, confirming our first hypothesis (H1). The enhanced n-gram approach improved performance on the Jessa dataset from an F1-score of 0.71 to 0.78, indicating there is still room for improvement and our model provides a good baseline.
- RQ3: The fine-tuned MedRoBERTa.nl model succesfully extracted CHA_2DS_2–VASc scores with 95% overall accuracy, which strongly supports Hypothesis H2 and demonstrates the effectiveness of our approach.

7.3 Future Research

7.3.1 Automated ICD-10 Coding

Extending the binary AF classification to multi-label classification for the different types of Atrial Fibrillation (e.g. paroxysmal, persistent, permanent) would provide interesting insights in how the more traditional approach using XGBoost and TF-IDF features hold. It will also enable us to perform more precise ICD-10 coding and providing clinically relevant information for treatment decisions, as well as even more thorough reporting.

7.3.2 Model Improvements

More sophisticated approaches to better capture contextual understanding in medical text for our AF classification models should be researched. Our experiments with FastText yielded poor results, but there is room for improvement. It might be interesting to see how well other approaches, such as fine-tuning of a Large Language Model pre-trained on Dutch medical text, such as MedRoBERTa.nl, would perform in other downstream tasks related to Atrial Fibrillation.

More research is needed to incorporate word embeddings with XGBoost beyond our simple approaches and to test their feasibility within a clinical context.

Another approach we did not have time to finish was a sliding window approach and attention visualization techniques to handle longer clinical documents and improve model interpretability for clinical decision support.

To further test approaches for CHA_2DS_2–VASc extraction, it would be interesting to directly focus on inferring CHA_2DS_2–VASc scores from risk factors rather than any explicit score extraction. However, we hypothesize this task will be much more difficult given a smaller dataset and relies heavily on the pre-trained approaches that would be used.

7.3.3 Additional Quality Indicators

Apart from the quality indicators mentioned in this thesis, many other indicators exist for patients with Atrial Fibrillation. A logical next step in our pipeline would be to asses anticoagulant therapy for diagnosed patients. The quality indicator 'Proportion of AF patients with significant CHA_2DS_2–VASc score ($\geq$2) that received anticoagulant therapy.' set by the ESC can be researched further. We see promise in the fine-tuning of existing clinical models for more advanced information extraction from Dutch discharge notes. We note that in this thesis, we did not test the performance of any approaches that included translation of our original dataset, and think further research in this topic would also give meaningful insight into the possibilities for other quality indicators.

7.4 Personal Reflection

This thesis provided a great learning opportunity for me and allowed me to work with real clinical data from Jessa Hospital. I got to sharpen my knowledge on modern techniques such as Large Language Models, as well as other interesting Natural Language Processing techniques that were discussed in this thesis.

It was extremely interesting to learn about the ethical considerations and the responsibilities that come with working with real human clinical data, and found it a great addition to my learning journey to take certain courses on Human Subjects Research that allowed me to work with datasets such as MIMIC-IV (Johnson et al., 2024) (Johnson et al., 2023a).

I had anticipated more comprehensive outcomes at the start of this academic year, but I soon came to the conclusion that careful testing, documentation, and well-considered strategies are necessary when working on a bigger domain-specific project.

This work has shown that while modern Natural Language Processing (NLP) techniques hold great promise for improving reporting in healthcare settings and quality assessment, successful implementation requires careful consideration of domain knowledge, ethical implications, and a thorough feature engineering and pre-processing approach. The results suggest that both more traditional techniques like XGBoost in combination with TF-IDF, as well as Large Language Models can serve as a valuable aid to healthcare professionals, but in my opinion, I think human oversight and clinical judgment remain important, especially in more hazardous contexts outside of reporting.

The baseline approaches that were established in this thesis show there are many future investigations and practical applications for the reporting of quality indicators possible and there is still a lot of research to be done using modern NLP techniques.

Bibliography

Akiba, T., Sano, S., Yanase, T., Ohta, T., and Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Bojanowski, P., Grave, E., Joulin, A., and Mikolov, T. (2016). Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606*.

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794. ACM.

Constante, A. D., Suarez, J., Lourenço, G., Portugal, G., Cunha, P. S., Oliveira, M. M., Trigo, C., Pinto, F. F., and Laranjo, S. (2024). Prevalence, management, and outcomes of atrial fibrillation in paediatric patients: Insights from a tertiary cardiology centre. *Medicina (Kaunas)*, 60(9).

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.

Ergün, B., Ergan, B., Sözmen, M. K., Küçük, M., Yakar, M. N., Cömert, B., Gökmen, A. N., and Yaka, E. (2021). New-onset atrial fibrillation in critically ill patients with coronavirus disease 2019 (COVID-19). *J. Arrhythm.*, 37(5):1196–1204.

Falter, M., Godderis, D., Scherrenberg, M., Kizilkilic, S. E., Xu, L., Mertens, M., Jansen, J., Legroux, P., Kindermans, H., Sinnaeve, P., Neven, F., and Dendale, P. (2024). Using natural language processing for automated classification of disease and to identify misclassified ICD codes in cardiac disease. *Eur. Heart J. Digit. Health*, 5(3):229–234.

FOD Volksgezondheid, Veiligheid van de Voedselketen en Leefmilieu (2016). Minimale ziekenhuis gegevens (mzg). `https://icd.who.int/browse10/2019/en`. Accessed: 2025-02-14.

Gažová, A., Leddy, J. J., Rexová, M., Hlivák, P., Hatala, R., and Kyselovič, J. (2019). Predictive value of CHA2DS2-VASc scores regarding the risk of stroke and all-cause mortality in patients with atrial fibrillation (CONSORT compliant). *Medicine (Baltimore)*, 98(31):e16560.

Goldberger, A., Amaral, L., Glass, L., Hausdorff, J., Ivanov, P. C., and Mark, R. Stanley, H. E. e. a. (2002). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, [Online] 101(23):e215–e220.

Harnoune, A., Rhanoui, M., Mikram, M., Yousfi, S., Elkaimbillah, Z., and El Asri, B. (2021). Bert based clinical knowledge extraction for biomedical knowledge graph construction and analysis. *Computer Methods and Programs in Biomedicine Update*, 1:100042.

Hindricks, G., Potpara, T., Dagres, N., Arbelo, E., Bax, J. J., Blomström-Lundqvist, C., Boriani, G., Castella, M., Dan, G.-A., Dilaveris, P. E., Fauchier, L., Filippatos, G., Kalman, J. M., La Meir, M., Lane, D. A., Lebeau, J.-P., Lettino, M., Lip, G. Y. H., Pinto, F. J., Thomas, G. N., Valgimigli, M., Van Gelder, I. C., Van Putte, B. P., Watkins, C. L., and

ESC Scientific Document Group (2020). 2020 ESC Guidelines for the diagnosis and management of atrial fibrillation developed in collaboration with the European Association for Cardio-Thoracic Surgery (EACTS): The task force for the diagnosis and management of atrial fibrillation of the European Society of Cardiology (ESC) developed with the special contribution of the European Heart Rhythm Association (EHRA) of the ESC. *European Heart Journal*, 42(5):373–498.

Ji, S., Hölttä, M., and Marttinen, P. (2021). Does the magic of BERT apply to medical code assignment? A quantitative study. *Computers in Biology and Medicine*, 139:104998.

Johnson, A., Pollard, T., Horng, S., Celi, L. A., and Mark, R. (2023a). MIMIC-IV-Note: Deidentified free-text clinical notes (version 2.2). *PhysioNet*.

Johnson, A. E. W., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T. J., Hao, S., Moody, B., Gow, B., Lehman, L.-w. H., Celi, L. A., and Mark, R. G. (2023b). MIMIC-IV, a freely accessible electronic health record dataset. *Scientific Data*, 10(1):1.

Johnson, A. E. W., Bulgarelli, L., Shen, L., Gayles, A., Shammout, A., Horng, S., Pollard, T. J., Hao, S., Moody, B., Gow, B., Lehman, L.-w. H., Celi, L. A., and Mark, R. G. (2024). MIMIC-IV (version 3.1). *PhysioNet*.

Kaur, R., Ginige, J. A., and Obst, O. (2023). AI-based ICD coding and classification approaches using discharge summaries: A systematic literature review. *Expert Systems with Applications*, 213.

Kim, J., Verkijk, S., Geleijn, E., van der Leeden, M., Meskers, C., Meskers, C., van der Veen, S., Vossen, P., and Widdershoven, G. (2022). Modeling Dutch Medical Texts for Detecting Functional Categories and Levels of COVID-19 Patients. In Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Odijk, J., and Piperidis, S., editors, *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4577–4585, Marseille, France. European Language Resources Association.

Lhoest, Q., Villanova del Moral, A., Jernite, Y., Thakur, A., von Platen, P., Patil, S., Chaumond, J., Drame, M., Plu, J., Tunstall, L., Davison, J., Šaško, M., Chhablani, G., Malik, B., Brandeis, S., Le Scao, T., Sanh, V., Xu, C., Patry, N., McMillan-Major, A., Schmid, P., Gugger, S., Delangue, C., Matussière, T., Debut, L., Bekman, S., Cistac, P., Goehringer, T., Mustar, V., Lagunas, F., Rush, A., and Wolf, T. (2021). Datasets: A Community Library for Natural Language Processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Liu, X., Hersch, G. L., Khalil, I., and Devarakonda, M. (2021). Clinical trial information extraction with bert.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach.

Loper, E. and Bird, S. (2002). Nltk: The natural language toolkit.

Michel, P., Levy, O., and Neubig, G. (2019). Are Sixteen Heads Really Better than One?

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality.

Muizelaar, H., Haas, M., van Dortmont, K., van der Putten, P., and Spruit, M. (2024a). Extracting patient lifestyle characteristics from Dutch clinical text with BERT models. *BMC Medical Informatics and Decision Making*, 24(1):151.

Muizelaar, H., Haas, M., van Dortmont, K., van der Putten, P., and Spruit, M. (2024b). Extracting patient lifestyle characteristics from Dutch clinical text with BERT models. *BMC Med. Inform. Decis. Mak.*, 24(1):151.

Névéol, A., Dalianis, H., Velupillai, S., Savova, G., and Zweigenbaum, P. (2018). Clinical Natural Language Processing in languages other than English: opportunities and challenges. *J. Biomed. Semantics*, 9(1):12.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Tenney, I., Das, D., and Pavlick, E. (2019). Bert rediscovers the classical nlp pipeline.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2023). Attention Is All You Need.

Verkijk, S. and Vossen, P. (2021). MedRoBERTa.nl: A Language Model for Dutch Electronic Health Records. *Computational Linguistics in the Netherlands Journal*, 11:141–159.

Voita, E., Talbot, D., Moiseev, F., Sennrich, R., and Titov, I. (2019). Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned.

VSC (Vlaams Supercomputing Center) (2025). Access: Terminal Interface &x2014; VSC documentation. `https://docs.vscentrum.be/compute/terminal/index.html`. [Accessed 04-05-2025].

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. (2020). Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

World Health Organisation (2019). International Statistical Classification of Diseases and Related Health Problems 10th Revision. `https://icd.who.int/browse10/2019/en`.

Zappatore, M. and Ruggieri, G. (2024). Adopting machine translation in the healthcare sector: A methodological multi-criteria review. *Comput. Speech Lang.*, 84(C).