UHASSELT
KNOWLEDGE IN ACTION

Maastricht University

Faculteit Wetenschappen
School voor Informatietechnologie

master in de informatica

Masterthesis

Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?

Dongho Chun

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

PROMOTOR :

Prof. dr. Stijn VANSUMMEREN

BEGELEIDER :

dr. ir. Axel FAES

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

Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?

Dongho Chun

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

PROMOTOR :

Prof. dr. Stijn VANSUMMEREN

BEGELEIDER :

dr. ir. Axel FAES

Dankwoord

Graag wil ik mijn begeleider, dr. ir. Axel FAES, bedanken voor zijn onmisbare begeleiding en waardevolle feedback gedurende een intensieve periode van 12 maanden. De literatuur die hij in het begin van dit onderzoek aanbood, samen met zijn verdere begeleiding, heeft het mogelijk gemaakt om tijdens de onderzoeksperiode een duidelijk en concreet beeld van het onderzoek te vormen. Bovendien ben ik mijn begeleider ook zeer dankbaar voor de tijd en moeite die hij heeft besteed aan zowel de vergaderingen als het beantwoorden van mijn vragen gedurende het volledige traject.

Daarnaast wil ik ook mijn promotor, prof. dr. Stijn VANSUMMEREN, bedanken voor de tijd die hij heeft genomen voor de tussentijdse presentatie en voor zijn onmisbare feedback tijdens deze gelegenheid. En natuurlijk kan ik zijn snelle en deskundige hulp bij bijkomende vragen niet vergeten. Dankzij zijn snelle antwoorden heb ik mijn verplichtingen tot een goed einde kunnen brengen.

Samenvatting

Hart- en vaatziekten vormen in de moderne wereld nog steeds één van de belangrijkste doodsoorzaken bij de mens. In de praktijk kunnen cardiologen op een goedkope en efficiënte manier hartritmestoornissen (aritmieën) bij een patiënten ontdekken door bepaalde patronen in de hartslagen te identificeren met behulp van een elektrocardiogram (ECG). Met de opkomst van artificiële intelligentie en publiek beschikbare ECG-data bestaan er talloze studies waarin AI modellen getraind worden om automatisch aritmieën te onderscheiden in ECG-data. Echter zijn de ECG-data waarop dergelijke AI modellen getraind worden, voorzien van extra informatie door de cardiologen. Als gevolg zijn er minder onderzoeken binnen het andere domein van AI waar de ECG-data niet gelabeld zijn met extra informatie door cardiologen: unsupervised learning. Unsupervised learning binnen dit domein brengt een zeer groot voordeel met zich mee dat cardiologen niet nodig zijn voor het trainen van het AI model, waardoor er een veel grotere hoeveelheid aan ECG-data beschikbaar is voor het trainen van het model. Het doel van dit onderzoek is om methodes en technieken binnen dit domein te onderzoeken en de resultaten hieruit te vergelijken om conclusie te vormen over de effectiviteit van deze methodes. Hieruit volgt de hoofdonderzoeksvraag van dit onderzoek: Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?

Clustering algoritmen zijn technieken binnen unsupervised learning die toegepast kunnen worden om automatisch groepen te vormen die gelijkaardige patronen bevatten binnen de data. In de praktijk zou dit gebruikt kunnen worden om dezelfde aritmieën in de ECG-data samen te groeperen, aangezien deze dezelfde kenmerken of patronen delen. Voordat zulke clustering algoritmen toegepast kunnen worden, is er een bijkomend onderzoek nodig om de ruwe ECG-data te verwerken. Dit is een essentiële stap, omdat ruwe ECG-data niets meer dan ruwe signalen zijn die verschillende vormen van ruis bevatten die tijdens de metingen in de praktijk optreden door bijvoorbeeld elektrische interferentie of ademhalingen van de patiënt. Vervolgens moet het AI model op een efficiënte manier getraind worden om het model toepasbaar te maken in de praktijk. Hiervoor worden er verschillende technieken onderzocht om de belangrijkste elementen in de signalen te identificeren en extraheren, terwijl de minder relevante elementen verwijderd worden. Op deze manier kan het AI model trainen op significant kleinere data, wat vanzelfsprekend verschillende voordelen met zich meebrengt, terwijl de prestatie van het model zelfs verbetert. Door deze diverse dataverwerking technieken te onderzoeken samen met verschillende clustering algoritmen is het mogelijk om vergelijkingen te maken tussen de prestaties van deze technieken.

Om de resultaten te evalueren worden er verschillende evaluatiemetrieken voorgesteld, waaronder er wordt gekeken naar hoe "zeker" de clustering algoritmen zijn van de clusters die ze aangemaakt hebben. Om zoveel mogelijk combinaties te vergelijken wordt er ook geëxperimenteerd met verschillende subsets van de primaire datasets die in dit onderzoek gebruikt worden. In de resultaten blijkt dat er een significant sterkere clustering is voor een bepaalde combinatie van dataverwerking technieken en een subset van de oorspronkelijke dataset die in dit onderzoek gebruikt wordt. Daarnaast wordt er gemeten hoe goed de aangemaakte clusters een bepaalde aritmie representeren. Echter wordt er in dit onderzoek geconcludeerd dat er geen betekenisvolle clusters aangemaakt worden, maar toch zijn er een aantal combinaties die opmerkelijk diverse resultaten opleveren.

De finale conclusie van deze thesis is dat het vergelijken van clustering algoritmen binnen ECG-analyse complexer is dan wat sommige bestaande studies suggereren. De uitdaging ligt voornamelijk in het verwerken van ruwe ECG-data en niet in de clustering algoritmen zelf.

Inhoudsopgave

1 Inleiding **7**

2 Literatuurstudie **11**
2.1 Elektrocardiogram . . . 11
2.2 Preprocessing en feature-extractie van ECG-data . . . 13
2.2.1 Ruisfiltering . . . 14
2.2.2 Segmentatie . . . 15
2.2.3 Dimensionality reduction . . . 17
2.3 Unsupervised learning in ECG-analyse . . . 17
2.3.1 K-means . . . 18
2.3.2 Agglomerative clustering . . . 18
2.3.3 Affinity propagation . . . 18
2.3.4 Spectral clustering . . . 19
2.4 Gebruikte combinaties van preprocessing en clustering technieken . . . 19
2.4.1 Studie 1: Rodriguez-Sotelo et al. (2007) [RSCFCD07] . . . 20
2.4.2 Studie 2: J.H. Abawajy et al. (2013) [AKC13] . . . 20
2.4.3 Studie 3: J.L. Rodriguez-Sotelo et al. (2010) [RSDTPO$^+$10] . . . 21
2.4.4 Een synthese . . . 23
2.5 ECG databases uit studies . . . 23

3 Methodologie **25**
3.1 Data . . . 25
3.2 Preprocessing . . . 25
3.2.1 Flatten . . . 25
3.2.2 Downsampling . . . 26
3.2.3 Sliding window . . . 27
3.2.4 Normalisatie . . . 27
3.3 Feature-extractie en dimensionality reduction . . . 28
3.3.1 Neuraal netwerk . . . 28
3.3.2 Autoencoder . . . 30
3.3.3 Deep autoencoder . . . 31
3.3.4 Training . . . 32
3.3.5 Toepassing . . . 33
3.4 Clustering . . . 34
3.5 Evaluatie . . . 35
3.6 Extra implementatie . . . 37
3.6.1 Data . . . 37
3.6.2 Preprocessing en clustering . . . 37

4 Implementatie en Resultaten **41**
4.1 Deep autoencoder en downsampling . . . 41
4.1.1 SR, AF en VA . . . 42

4.1.2 SR en AF . . . 43
4.1.3 Evenwichtige dataset: SR, AF en VA . . . 44
4.1.4 Evenwichtige dataset: SR en AF . . . 45
4.2 Deep autoencoder en sliding window . . . 46
4.2.1 SR, AF en VA . . . 47
4.2.2 SR en AF . . . 48
4.2.3 Evenwichtige dataset: SR, AF en VA . . . 49
4.2.4 Evenwichtige dataset: SR en AF . . . 50
4.3 Extra implementatie . . . 51
4.3.1 Naïeve methode . . . 51
4.3.2 Smoothing en deep autoencoder . . . 52

5 Discussie **53**
5.1 Resultaten op de volledige PTB-XL AFib dataset . . . 53
5.1.1 Verdeling en sterkte van de clustering . . . 53
5.1.2 Purity . . . 54
5.2 Resultaten op de PTB-XL AFib dataset zonder VA . . . 55
5.2.1 Verdeling en sterkte van de clustering . . . 55
5.2.2 Purity . . . 55
5.3 Resultaten op de evenwichtige PTB-XL AFib dataset . . . 56
5.3.1 Verdeling en sterkte van de clustering . . . 56
5.3.2 Purity . . . 56
5.4 Resultaten op de evenwichtige PTB-XL AFib dataset zonder VA . . . 57
5.4.1 Verdeling en sterkte van de clustering . . . 57
5.4.2 Purity . . . 57
5.5 Resultaten extra implementatie . . . 57

6 Conclusies **59**
6.1 Mijn bevindingen . . . 59
6.2 Persoonlijke reflectie . . . 60

Hoofdstuk 1

Inleiding

Hart- en vaatziekten, ook wel cardiovasculaire aandoeningen (CVD) genoemd, waren in 2019 wereldwijd verantwoordelijk voor 32% van alle sterfgevallen (17,9 miljoen mensen). Ze vormen nog steeds de belangrijkste doodsoorzaak op jaarbasis. Vroege en tijdige detectie van deze aandoeningen is daarom van cruciaal belang, zodat patiënten tijdig kunnen starten met de nodige behandeling en begeleiding [WHO21].
Hartritmestoornissen, ook wel aritmieën (*cardiac arrhythmia)* genoemd, zijn aandoeningen waarbij het normale hartritme verstoord is. Dit normale hartritme wordt ook wel het sinusritme (*sinus rhythm*) genoemd. Aritmieën kunnen onderverdeeld worden in twee verschillende types: tachycardie (*tachycardia*) en bradycardie (*bradycardia*). Zoals weergegeven in Tabel 1.1 wordt tachycardie getypeerd door een snelle hartslag en bestaat uit vijf verschillende types aritmieën. In tegenstelling tot tachycardie wordt bradycardie getypeerd door een trage hartslag die uit twee types bestaat [Sta23].

Tachycardie (snelle hartslag)	**Bradycardie (trage hartslag)**
Atriumfibrilleren (*atrial fibrillation*)	Sinusknoopziekte (*sick sinus syndrome*)
Boezemflutter (*atrial flutter*)	Geleidingsstoornis (*conduction block*)
Supraventriculaire tachycardie (*supraventricular tachycardia*)	
Ventrikelfibrilleren (*ventricular fibrillation*)	
Ventrikeltachycardie (*ventricular tachycardia*)	

Tabel 1.1: Vijf typen tachycardie en twee typen bradycardie.

Deze aritmieën hebben unieke kenmerken en kunnen geïdentificeerd worden door experts. Het is daarom essentieel om hartritmes nauwkeurig te analyseren en van elkaar te onderscheiden, zodat aritmieën tijdig opgespoord kunnen worden en het specifieke type kan worden geïdentificeerd. De analyse van het hartritme gebeurt doorgaans door middel van monitoring met een elektrocardiogram (ECG). Hierbij worden spanningsverschillen tussen verschillende punten op het lichaam gemeten door middel van elektroden die op verschillende lichaamsdelen worden geplaatst. Het ECG is het meest gebruikte en eenvoudigste onderzoek bij patiënten met hartklachten. De geregistreerde elektrische signalen worden vervolgens door een cardioloog geanalyseerd om belangrijke informatie zoals de hartfrequentie en de regelmaat van het ritme te verkrijgen [FCG08] [Har].
In de implementatie in deze thesis wordt er gebruikgemaakt van publieke ECG datasets die online beschikbaar gesteld zijn. Deze datasets bevatten enkel ECG-gegevens van patiënten samen met het type aritmie dat de patiënt heeft om uiteindelijk de performantie van de implementatie te valideren. Andere persoonlijke gegevens worden uiteraard niet vermeld in de datasets om privacyredenen.

ECG is binnen de geneeskunde reeds een gestandaardiseerde methode, voornamelijk vanwege

zijn simpliciteit, doeltreffendheid en lage kostprijs. Desondanks brengt het gebruik van ECG's enkele belangrijke uitdagingen met zich mee. Een van de grootste moeilijkheden bij de handmatige analyse van ECG-signalen is het accuraat detecteren en classificeren van diverse golfvormen en morfologische patronen binnen het signaal. Deze analyse is niet alleen tijdsintensief, maar ook vatbaar voor menselijke fouten. Bovendien is het ECG één van de meest toegepaste diagnostische technieken binnen de geneeskunde, met wereldwijd 200 miljoen registraties per jaar. Dit benadrukt de nood aan geautomatiseerde en computerondersteunde analysetechnieken die een snellere, nauwkeurigere en efficiëntere interpretatie van ECG-signalen mogelijk maken.
Computerondersteunde technieken voor de analyse van ECG-signalen bestaan inmiddels al langer dan vijftig jaar. Hoewel deze traditionele systemen bedoeld zijn om de interpretatie van ECG's te vergemakkelijken, brengen ze ook enkele beperkingen en uitdagingen met zich mee. Een eerste belangrijke beperking betreft de nauwkeurigheid van deze systemen. Uit een studie uit 1991 blijkt dat de prestaties van dergelijke computersystemen gemiddeld 6,6% minder accuraat zijn dan de interpretaties van ervaren cardiologen. Een tweede probleem is dat de toenemende afhankelijkheid van deze systemen ertoe heeft geleid dat medische studenten minder grondig worden opgeleid in de interpretatie van ECG's, terwijl de traditionele automatische systemen nog steeds niet het prestatieniveau van menselijke experts evenaren [RKN21] [SM98] [KFS18].

Met de opkomst van artificiële intelligentie (AI) wordt er gestreefd naar het verbeteren van traditionele computerondersteunde technieken voor ECG-analyse. In een vergelijkende studie werd de nauwkeurigheid van ECG-interpretaties geëvalueerd bij drie groepen: traditionele computersystemen, cardiologen en AI-gebaseerde systemen (AI-ECG), op basis van 500 ECG's. De resultaten toonden aan dat 202 interpretaties afkomstig van traditionele systemen, 123 van AI-ECG en 90 van cardiologen als "onaanvaardbaar" werden beoordeeld, wat betekent dat deze interpretaties grote fouten bevatten. Deze bevindingen suggereren dat AI-ECG een verbetering biedt ten opzichte van traditionele systemen, hoewel de prestaties van menselijke experts nog steeds superieur blijven [RKN21]. Gebruik van AI binnen elektrocardiografie brengt verschillende maatschappelijke voordelen met zich mee. Naast enkel de interpretatie van ECG kan een goed getraind model verschillende cardiovasculaire problemen voorspellen zoals een beroerte.
In de laatste jaren is er ook een groei van draagbare elektronische toestellen zoals smartwatches, sporthorloges en implanteerbare apparaten die allerlei metingen doen van onder andere het hart. Deze apparaten kunnen de metingen gebruiken om aritmieën te voorspellen in realtime en kunnen vervolgens de gebruikers verwittigen voor een mogelijk gevaar. Zoals eerder vermeld wordt ECG in een klinische instelling typisch gemeten door verschillende elektroden op het lichaam te plaatsen. Draagbare elektronische apparaten hebben vaak één elektrode (*single-lead ECG*) om het ECG te genereren van de gebruiker. De *AliveCor Kardia Mobile* heeft zelfs twee elektrodes voor accuratere metingen en heeft een nauwkeurigheid (*accuracy*) van 84% voor atriumfibrilleren.
Één van de belangrijkste kenmerken bij ECG-data is dat deze signalen zijn die oorspronkelijk veel ruis bevatten. Daarom is het essentieel om signaalverwerkingsalgoritmen toe te passen op deze data om deze ruis uit te filteren. AI kan in deze context toegepast worden om automatisch en zo accuraat mogelijk de ruis te identificeren in de data en deze te verwijderen. Dit maakt het mogelijk om patronen duidelijker te identificeren en te onderscheiden van elkaar om accurate voorspellingen te maken.
Verder is het ook mogelijk om non-cardiovasculaire ziektes te identificeren, omdat AI modellen ook patronen kunnen herkennen die niet duidelijk zichtbaar zijn voor het menselijke oog. Maar in deze thesis ligt de nadruk voornamelijk op cardiovasculaire ziektes, meer specifiek de aritmieën. [MSMB23].

In de voorbije jaren zijn er steeds meer studies verschenen waarin *deep learning* en *machine learning* technieken worden gecombineerd en toegepast op ECG-signalen, met als doel het classificeren of onderscheiden van verschillende aritmieën. *Supervised machine learning* is een specifieke vorm van machine learning waarbij elk ECG-signaal voorzien wordt van een label. In deze context verwijzen zulke labels doorgaans naar het type hartritme (bijvoorbeeld sinusritme of atriumfibrilleren). Het toekennen van deze labels vereist echter de expertise van

ECG-specialisten, zoals cardiologen.
Zoals eerder vermeld, worden er jaarlijks ongeveer 200 miljoen ECG's geregistreerd. De datasets die beschikbaar zijn voor supervised modellen vormen slechts een fractie van deze totale hoeveelheid, aangezien het handmatig labelen van zulke grote aantallen ECG's door experts tijdrovend en kostbaar is. Om die reden ligt de focus in deze thesis op *unsupervised machine learning* technieken. [NSHF23]

Unsupervised learning gebruikt machine learning algoritmen om modellen te trainen die in staat zijn om zelfstandig patronen te herkennen in de data, zonder dat handmatig gelabelde voorbeelden nodig zijn. Zoals er verschillende soorten supervised learning technieken bestaan zoals *regression* en *classification*, bestaan er drie soorten unsupervised learning technieken: *clustering*, *association* en *dimensionality reduction*. Hierbij zijn clustering algoritmen interessant en toepasselijk in deze context. Clustering algoritmen kijken naar de gelijkenissen en verschillen tussen verschillende samples om samples die lijken op elkaar samen te groeperen. Iedere sample kan gerepresenteerd worden door een ECG-opname, waardoor clustering algoritmen de ECG-opnames, die gelijkaardige kenmerken delen en dus een zelfde type hartritme voorstellen, samen groeperen. Hierdoor is het mogelijk om snel en automatisch een overzicht te verkrijgen van groepen die ieder een type hartritme representeren samen met de bijhorende ECG-opnames. Nadat een dergelijk clustering model is getraind en goede scores behaalt op verschillende evaluatietechnieken, is het mogelijk om in de praktijk het model toe te passen om nieuwe ECG-opnames uit patiënten snel en efficiënt te classificeren met bijvoorbeeld de aritmieën uit Tabel 1.1. Deze aanpak onderscheidt zich van supervised learning doordat het model een grote hoeveelheid data als invoer kan verwerken zonder dat gelabelde data vereist is. Het kan direct ruwe data van ECG-machines gebruiken, waardoor het handmatig labelen door een expert niet langer een *bottleneck* vormt in het proces.
Echter zijn er in de praktijk bijkomende stappen nodig vanwege de aard van de dataset. De ECG-data in de dataset zijn ruwe signalen die niet rechtstreeks als invoer voor het clustering algoritme gebruikt kunnen worden. Zoals bij andere AI-modellen is er dus ook bij clustering meestal een *preprocessing* stap. Deze stap omvat het verwerken van de ruwe dataset, zodat onnodige informatie zoals ruis wordt verwijderd. De eerder genoemde dimensionality reduction kan hierbij ook toegepast worden om de grootte van de dataset te reduceren, waardoor clustering efficiënter verloopt. Echter wordt associatie niet beschouwd, aangezien dit niet relevant is voor het onderwerp van deze thesis [YSD23] [IBM24].

Er bestaan echter verschillende studies die diverse combinaties van AI-modellen en preprocessing technieken toepassen en interessante resultaten opleveren. Toch blijft het onduidelijk welke methode binnen unsupervised learning de juiste oplossing biedt voor het classificeren van verschillende types aritmieën en in welke mate unsupervised modellen in staat zijn om relevante en interpreteerbare clusters van hartritmes te detecteren. Dit leidt tot de centrale onderzoeksvraag van deze thesis:

Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?

De focus van deze thesis ligt op het implementeren van verschillende clustering algoritmen in combinatie met verschillende preprocessing technieken. Ten slotte worden de verkregen resultaten geanalyseerd en vergeleken met bestaande technieken uit eerdere studies.

Deze thesis is opgebouwd uit de volgende hoofdstukken:

- In **hoofdstuk 2** wordt de bestaande literatuur besproken en met elkaar vergeleken. Deze literatuur omvat studies met een gelijkaardig doel: het groeperen en onderscheiden van aritmieën op basis van ECG-data door unsupervised learning technieken toe te passen.

- In **hoofdstuk 3** wordt de methodologie van deze thesis toegelicht. Hierbij worden twee verschillende implementaties besproken, die afwijken van de aanpakken uit de literatuurstudie.

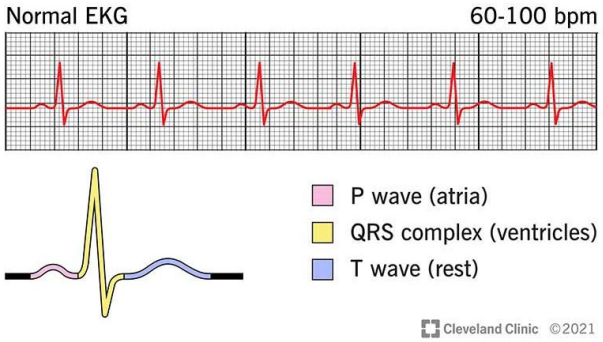
- **Hoofdstuk 4** geeft een overzicht van de resultaten van de implementaties, samen met de technische implementatiedetails.
- Verder in **hoofdstuk 5** worden de belangrijkste bevindingen uit deze resultaten besproken.
- Tot slot bevat **hoofdstuk 6** mijn persoonlijke reflectie waarin ik mijn bevindingen en het leerproces bij het uitvoeren van dit onderzoek toelicht.

Hoofdstuk 2

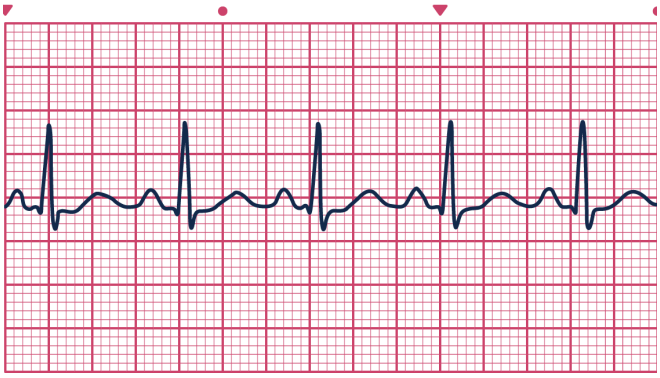
Literatuurstudie

2.1 Elektrocardiogram

Wanneer de elektrische activiteit van het hart wordt gemeten, wordt deze geregistreerd in een elektrocardiogram (ECG). Een ECG-machine plot dit signaal op elektrocardiogrampapier (*electrocardiograph paper*), zoals geïllustreerd in Figuur 2.1 met een standaard snelheid van 25 mm/s. De x-as stelt de tijd voor. De afstand tussen twee vetgedrukte verticale lijnen komt overeen met één seconde. Deze wordt verder onderverdeeld in vijf kleinere vakjes van 0,2 seconde, die op hun beurt opnieuw zijn onderverdeeld in vijf kleinere segmenten van 0,04 seconde. De y-as geeft de elektrische spanning weer in millivolt (mV).
In Figuur 2.1 worden de verschillende componenten van een enkele ECG-cyclus weergegeven. Deze cyclus stelt de elektrische activiteit van één hartslag voor en bestaat uit een P-golf, een QRS-complex en een T-golf. De cyclus begint met de P-golf, die de depolarisatie van de atria (de bovenste hartkamers) weergeeft.
Het QRS-complex, gekenmerkt door scherpe pieken, weerspiegelt de depolarisatie van de ventrikels (de onderste hartkamers). In normale omstandigheden duurt dit complex tussen de 60 en 100 milliseconden. Ten slotte representeert de T-golf de repolarisatie van de ventrikels, oftewel het herstel naar de rusttoestand [SC25].



Figuur 2.1: Respectievelijk: ECG bij sinusritme, de P-golf, QRS-complex en T-golf van één hartslag. [Cle20]



Figuur 2.2: ECG van een patiënt met sinusritme [oP].

Figuur 2.3: ECG van een patiënt met atriumfibrilleren [oP].

Figuur 2.4: ECG van een patiënt met boezemflutter [oP].

Door deze verschillende gegevens die verkregen kunnen worden uit verschillende componenten in het ECG kunnen cardiologen achterhalen over welk type aritmie het gaat. In Figuur 2.2, 2.3 en 2.4 worden respectievelijk het ECG van een normaal sinusritme, atriumfibrilleren en boezemflutter weergegeven. Het valt op eerste zicht op dat deze drie ECG's verschillende en unieke patronen hebben.
Bij het identificeren van het soort aritmie kunnen er vijf vragen gesteld worden:

1. Wat is de hartslag?
2. Wat is het ritme?

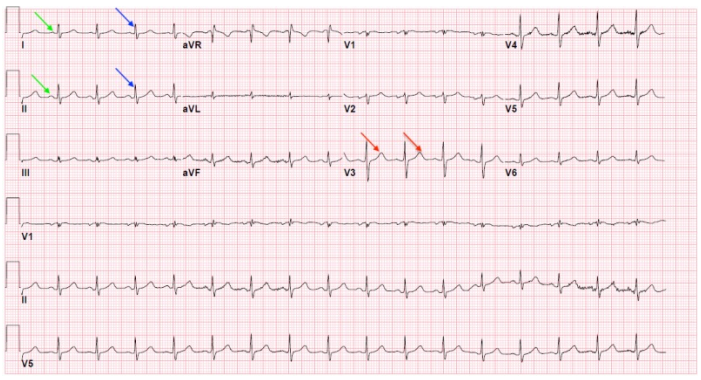
3. Is er een P-golf voor ieder QRS-complex en zijn deze positief en uniform?

4. Hoe groot is het PR-interval?

5. Zijn de QRS-complexen gelijkaardig en hoe groot zijn deze golven?

Door deze vragen te beantwoorden, kan een cardioloog de juiste aritmie vaststellen. Atriumfibrilleren in Figuur 2.3 wordt voornamelijk gekenmerkt door een onregelmatig ritme dat te zien is aan de frequentie van de QRS-complexen. In het geval van boezemflutter in Figuur 2.4 valt het voornamelijk op dat de golven een zaagtandpatroon vormen. Deze golven worden fluttergolven (f-golven) genoemd [oP].

In Figuur 2.5 is te zien dat, in tegenstelling tot 2.1, meerdere ECG-signalen zijn samengevoegd. Dit komt doordat er bij klinische ECG-metingen niet slechts één spanningsafleiding tussen twee elektroden wordt geregistreerd, maar meerdere. Aangezien het hart een driedimensionale structuur is, worden metingen uit verschillende invalshoeken uitgevoerd om een volledigere beeld te verkrijgen van de elektrische activiteit.
Bij een typische *12-lead ECG* worden gegevens verkregen via tien elektroden, die op vooraf bepaalde plaatsen op het lichaam worden aangebracht. Op basis hiervan worden twaalf afleidingen (ook wel *leads* genoemd) berekend. Deze afleidingen, aangeduid in Figuur 2.5 als I, II, III, aVR, aVL, aVF, V1 t.e.m. V6, geven elk een specifieke kijk op de elektrische activiteit van het hart [SC25] [PLT+23].
In openbare ECG-datasets die voor onderzoeksdoeleinden worden gebruikt, worden daarom vaak details gespecificeerd zoals het aantal afleidingen, het aantal opgenomen samples of patiënten, de *sampling rate*, en de totale duur van iedere sample.
Een sampling rate van 500 Hz voor een 12-lead ECG betekent dat er 500 datapunten per seconde per afleiding worden gemeten, wat neerkomt op 6 000 datapunten per seconde in totaal (12 afleidingen × 500 samples) [GGM+23].



Figuur 2.5: Elektrocardiogrampapier in de praktijk. De zwarte labels duiden de namen van de specifieke afleidingen aan. Groene pijlen geven de P-golven weer, blauwe pijlen het QRS-complex en rode pijlen de T-golven [SC25].

2.2 Preprocessing en feature-extractie van ECG-data

Het doel van deze thesis is om op basis van ECG-data een unsupervised model te ontwikkelen, dat zonder gelabelde gegevens, patronen in de data kan herkennen en automatisch aritmieën kan detecteren en van elkaar kan onderscheiden.
Vooraleer een dergelijk model getraind kan worden, is het cruciaal dat de inputdata van voldoende kwaliteit is.

2.2.1 Ruisfiltering

ECG-signalen die in de praktijk worden geregistreerd, bevatten vaak verschillende vormen van ruis (*noise*), wat de prestaties van het model aanzienlijk kan beïnvloeden. Deze ruis kan onder meer veroorzaakt worden door ademhaling van de patiënt, bewegingen van de huid, spiercontracties, elektrische interferentie of andere externe bronnen.
Het verwijderen van ruis uit ECG-signalen, een proces dat vaak *denoising* genoemd wordt, is daarom een essentiële stap in de pre-processing. Het doel hiervan is om het werkelijke signaal te versterken en irrelevante of storende fluctuaties te onderdrukken, zodat de nauwkeurigheid van het uiteindelijke model toeneemt.
In de literatuur zijn verschillende technieken beschreven om ruis in ECG-data te reduceren, waaronder:

- Baseline Wander Correction
- Band-pass filter
- Moving average filters
- Discrete Wavelet Transform
- EMD Transform

We bespreken deze technieken in meer detail in wat volgt.

Baseline wander, een veelvoorkomende vorm van lage frequentie ruis, wordt vaak veroorzaakt door ademhaling. Een veelgebruikte methode om deze vorm van ruis te onderdrukken, is gebaseerd op het gebruik van twee mediaanfilters. De eerste filter, met een venster van 200 ms, verwijdert de P-golven en QRS-complexen. Vervolgens verwijdert een tweede filter met een venster van 600 ms de resterende T-golven. Het resultaat is een benadering van de baseline wander, die vervolgens van het originele ECG-signaal wordt afgetrokken. Dit levert een opgeschoond signaal op, vrij van deze vorm van ruis [ARS+22].

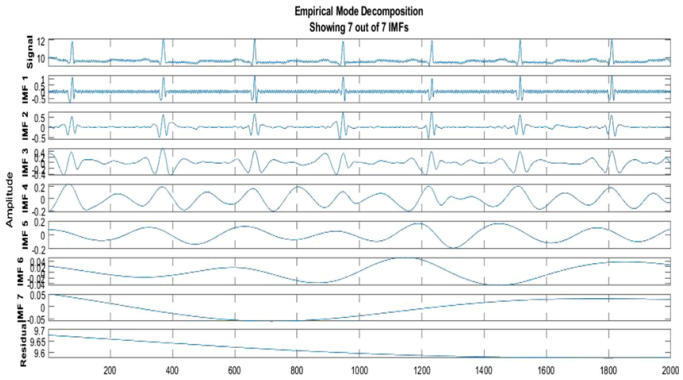
Een andere veelgebruikte methode voor ruisfiltering is het toepassen van frequentiefilters, zoals *high-pass*, *low-pass* en *band-pass* filters.

- **High-pass filters** laten alleen hoge frequenties door en blokkeren lage frequenties.
- **Low-pass filters** doen het omgekeerde: ze laten lage frequenties door en blokkeren hoge frequenties.
- **Een band-pass filter** combineert beide en laat enkel een specifiek frequentiebereik door.

Een bekend voorbeeld is de *Butterworth filter*, die typisch een frequentiebereik van 0,5 Hz tot 50 Hz doorlaat. Hiermee worden frequenties onder de 0,5 Hz (zoals baseline wander) en boven de 50 Hz (zoals spieractiviteit of elektrische ruis) onderdrukt. Gezien de meest relevante ECG-componenten (zoals de P- en T-golven) zich bevinden binnen het bereik van 0,5 tot 10 Hz, is de Butterworth filter een geschikte keuze voor het onderdrukken van ruis in ECG-signalen. Het exacte frequentiebereik kan echter worden aangepast in functie van de specifieke toepassing en context [ARS+22].

Binnen het domein van signaalverwerking is de *moving average* filter een veelgebruikte techniek. Deze methode vervangt elk datapunt door het gemiddelde van de omliggende waarden, wat een *smoothing effect* tot gevolg heeft. Hierdoor worden grote schommelingen in het signaal onderdrukt. In de context van ECG-analyse wordt deze techniek toegepast om pieken, zoals het QRS-complex, efficiënter te kunnen detecteren [MBBKH25] [NSHF23].

Een andere populaire methode voor ruisfiltering is de *Wavelet Transform*, die het originele signaal splitst in componenten met verschillende frequenties. Door deze decompositie wordt het mogelijk om ruis te isoleren. Een mogelijke aanpak is het definiëren van een drempelwaarde (*threshold*) waarbij de waveletcoëfficiënten (verkregen uit de decompositie) die onder



Figuur 2.6: Het bovenste signaal stelt het originele ECG voor. Daaronder volgen de *intrinsic mode functions* (IMF's), gerangschikt van hoge naar lage frequentie [DRMP21].

deze waarde liggen op nul worden gezet. Vervolgens kan het zuivere signaal gereconstrueerd worden via een *Inverse Wavelet Transform* [AD08].

De laatste geavanceerdere methode voor ruisfiltering die in deze thesis besproken wordt, is *Empirical Mode Decomposition* (EMD). Deze techniek splitst het oorspronkelijke signaal op in een som van *Intrinsic Mode Functions* (IMF's), elke IMF een signaalcomponent met een specifieke frequentie voorstelt (geïllustreerd in Figuur 2.6). De som van alle IMF's is gelijk aan het originele signaal.

Bij ruisfiltering worden IMF's die ruis bevatten verwijderd, zodat de som van de resterende IMF's een opgeschoond signaal vormt. Bij ECG-signalen vormt dit echter een uitdaging, aangezien ruis zich vaak bevindt in de IMF's met hogere frequenties. Deze frequentiebanden bevatten echter ook belangrijke informatie over het QRS-complex, waardoor het niet mogelijk is om eenvoudig alle hoge frequenties weg te filteren.

Daarom vereist EMD-gebaseerde ruisfiltering in ECG-toepassingen een meer verfijnde benadering. Meestal wordt hierbij het volgende stappenplan gevolgd:

1. Detectie van het QRS-complex in het oorspronkelijke signaal.
2. *Windowing* (vensters) rond de QRS-complexen om vervorming tijdens de IMF-decompositie te vermijden.
3. Statistische analyse van de IMF's om te bepalen welke componenten ruis bevatten.
4. Selectieve reconstructie van het signaal op basis van IMF's die als "zuiver" worden beschouwd.

Deze aanpak maakt het mogelijk om ECG-signalen te filteren of zuiveren zonder cruciale informatie over de hartritmestructuur (QRS-complex) te verliezen [BVWB08]. De bovenstaande stappen worden kort ter volledigheid opgelijst, maar worden in deze thesis niet in detail behandeld, aangezien voornamelijk de moving average methode toegepast zal worden.

2.2.2 Segmentatie

Voordat de overgang naar *feature engineering* gemaakt wordt, worden in verschillende studies twee bijkomende stappen uitgevoerd: het identificeren en vervolgens segmenteren van individuele hartslagen binnen het ECG-signaal. Deze stappen worden respectievelijk aangeduid als **QRS-detectie** en **segmentatie**. De nadruk ligt hierbij initieel op het detecteren van bijvoorbeeld de R-piek, aangezien dit punt doorgaans de hoogste amplitude heeft binnen één ECG-cyclus (één hartslag), en dus eenvoudiger te herkennen is dan de andere componenten (zoals de P- of T-golf).
Vervolgens is een consistente segmentatie van de ECG-signalen van cruciaal belang voor verdere

verwerking van de hartslagen, bijvoorbeeld tijdens de trainingsfase van een machine learning model.

Voor **QRS-detectie** zijn er verschillende algoritmen voorgesteld. Een studie beschrijft een typisch QRS-detectie algoritme bestaande uit drie stappen [SL05]:

1. **Lineaire filtering**: deze stap komt overeen met de eerder besproken methodes voor ruisfiltering. Er wordt een band-pass filter toegepast om ruis te onderdrukken en zo de signaal-ruisverhouding (SNR) te verbeteren.
2. **Niet-lineaire transformatie**: deze stap heeft als doel de QRS-complexen verder te versterken ten opzichte van de achtergrond, zodat deze beter te onderscheiden zijn.
3. ***Decision rule***: op basis van de verwerkte signalen wordt er bepaald waar de QRS-complexen zich bevinden.

De eerste twee stappen zijn voornamelijk bedoeld om de QRS-complexen te verscherpen en worden in de studie aangeduid als *preprocessor*. De derde stap staat vervolgens in voor de uiteindelijke detectie van de QRS-complexen.

Een **alternatieve QRS-detectie** methode die in een studie wordt toegepast, is het *Difference Operation Method* algoritme (DOM) [ARS+22], dat bestaat uit twee fasen:

1. **Detectie van de R-piek** via een differentiaalvergelijking, waarmee temporele veranderingen in het signaal worden opgespoord. Aangezien de R-piek de grootste amplitude heeft, levert deze methode een duidelijk maximum op rond dat punt.
2. **Bepaling van de Q- en S-punten**, respectievelijk als het laagste punt vóór en na de R-piek.

Een belangrijk voordeel van het DOM-algoritme is de eenvoud ervan. Er zijn geen complexe wiskundige berekeningen of geavanceerde signaalverwerkingstechnieken nodig om het QRS-complex te detecteren.

Door de volledige QRS-complexen te detecteren, wordt het mogelijk om over te gaan naar de **segmentatie** stap, waarbij het ECG-signaal opgesplitst wordt in afzonderlijke en correct uitgelijnde hartslagen door gebruik te maken van de gedetecteerde R-pieken uit de QRS-detectie stap. De segmentatie stap in de literatuur wordt typisch uitgevoerd door een venstergrootte te definiëren. Zo definieert de studie van Filipe Canento et al. (2013) [CLSF13] de duur van een typische hartslag als 600 ms, waarbij het startpunt van de hartslag 200 ms voor en het eindpunt 400 ms na de R-piek zich bevindt. Door dit venster te definiëren is het mogelijk om iedere individuele hartslag te extraheren na het identificeren van de overeenkomstige R-pieken. Over het algemeen bestaat er echter geen eenduidig "correct" antwoord voor de grootte van dit venster, tenzij er gebruik wordt gemaakt van geavanceerde algoritmen die op adaptieve wijze de P-golven en T-golven detecteren om vervolgens het beginpunt en het eindpunt van de individuele hartslagen te identificeren.

Een voorbeeld hiervan is te vinden in de studie van García-Isla, G. et al. (2021) [GIMC21] waarin meerdere groottes voor dit venster gedefinieerd worden op basis van het type segment dat geëxtraheerd moet worden (met de R-piek als referentiepunt):

- **P-golf**: [-300, 40] ms.
- **QRS-complex**: [-70, 60] ms.
- **PR-interval**: [-288, 0] ms.
- **Volledige hartslag**: [-300, 250] ms.

Afhankelijk van de context van het onderzoek is het essentieel om een keuze te maken voor de grootte van dit venster.

2.2.3 Dimensionality reduction

Na het segmenteren van het ECG-signaal volgt doorgaans de stap van feature engineering, een cruciale voorbereidende fase voor het trainen van een unsupervised clustering model. Deze fase omvat onder andere het verwijderen van irrelevante of overbodige *features* (kenmerken of eigenschappen), het toevoegen van nieuwe features of het transformeren van bestaande features. Een belangrijk onderdeel binnen feature engineering is dimensionality reduction, voornamelijk wanneer de dataset een zeer groot aantal features bevat. Het doel is om de dimensie van de data te reduceren met minimaal verlies aan informatie, zodat het clusteringsproces efficiënter en effectiever verloopt.

In de literatuur worden diverse technieken voor dimensionality reduction besproken. ***Principal Component Analysis* (PCA)** is een van de meest toegepaste dimensionality reduction methodes. PCA reduceert de dimensie van de data terwijl de belangrijkste informatie behouden blijft. Hoe "belangrijk" die informatie is, wordt bepaald door de variantie in de data, met andere woorden, de mate van de spreiding in de data. Hoe groter de variantie of spreiding, hoe belangrijker. PCA genereert achterliggend *principal components*, deze zijn nieuwe features verkregen door op een slimme manier de originele features samen te voegen. De eerste principal component bevat altijd de meeste variantie (en dus de meest relevante informatie), gevolgd door de tweede principal component, derde, enzovoort. Het aantal principal components dat behouden wordt (en dus de nieuwe dimensie van de dataset bepaalt), hangt af van de toepassingscontext en de mate van informatiebehoud die gewenst is. Dus als men n principal components kiest, zal de gereduceerde dataset n dimensies bevatten [AKC13].

Bovendien worden in de literatuur voornamelijk variaties van PCA toegepast, zoals *weighted-PCA*. In deze thesis wordt echter een alternatieve techniek voor dimensionality reduction gehanteerd, namelijk een **deep autoencoder**, waarvan de werking verder wordt toegelicht in de methodologie. Om deze redenen wordt in deze literatuurstudie enkel de algemene werking van PCA besproken.

2.3 Unsupervised learning in ECG-analyse

Zoals aangehaald in hoofdstuk 1 ligt de nadruk in deze thesis op het toepassen van unsupervised learning technieken, meer specifiek de clustering algoritmen om verschillende typen aritmieën van elkaar te onderscheiden op basis van ECG-data. Deze algoritmen groeperen (*clusteren*) datapunten op basis van gelijkenissen en verschillen tussen die datapunten. In deze context stellen de datapunten de verwerkte ECG-data voor die voortkomen uit de voorafgaande preprocessing en feature-extractie stappen.

Er zijn talloze studies in de literatuur die verschillende clustering algoritmen toepassen en evalueren afhankelijk van de context waarin deze toegepast worden. Daarom is het van belang om eerst een overzicht te geven van de verschillende typen clustering algoritmen. Er bestaan vier typen clustering algoritmen [IBM21b]:

- *Exclusive clustering*: elk datapunt behoort strikt tot één enkele cluster.
- *Overlapping clustering*: een datapunt kan tot meerdere clusters tegelijk behoren.
- *Hierarchical clustering*: in iedere iteratie wordt er een samenvoeging of een verdeling van groepen (clusters) uitgevoerd, afhankelijk van het type van het hierarchical clustering algoritme.
- *Probabilistic clustering*: datapunten worden toegewezen aan clusters op basis van statistieken.

Binnen elk van deze types bestaan verschillende algoritmen en variaties die gebruikt werden in verschillende studies. In het vervolg van deze sectie worden een aantal clustering algoritmen besproken die toegepast werden op ECG-data in eerdere onderzoeken [NSHF23].

2.3.1 K-means

Ook buiten de context van ECG-analyse is K-means een van de populairste clustering algoritmen voornamelijk vanwege de eenvoud in gebruik, de lage computationele complexiteit en de hoge snelheid van het algoritme. K-means behoort tot de categorie van exclusive clustering. Het finale doel van het algoritme is om k aantal clusters (groepen) te genereren, waarbij elke cluster bestaat uit datapunten die "gelijkaardig" zijn aan elkaar. In dit algoritme wordt elke cluster gerepresenteerd door een *centroid*. Een centroid is het middelpunt van de cluster en in iedere iteratie worden de coördinaten van deze centroids telkens geüpdatet totdat een bepaalde stopconditie bereikt wordt. Dit wordt *convergence* genoemd. De parameter k in K-means verwijst naar het aantal gewenste clusters dat initieel gedefinieerd moet worden voor het iteratieve proces in het algoritme. Het finale resultaat van K-means is een verdeling van alle invoerdata over k clusters, waarbij elk datapunt wordt toegewezen aan de dichtstbijzijnde centroid op basis van de Euclidische afstand. Aangezien elke centroid een cluster representeert, is het mogelijk om voor elk datapunt te bepalen tot welke groep het behoort [sl].

Binnen de context van deze thesis vormen de verwerkte ECG-signalen de invoerdata voor het K-means algoritme. Het finale resultaat bestaat uit clusters die elk een bepaald type aritmie representeren en overeenkomstige ECG-signalen groeperen. Indien bijvoorbeeld $k = 2$ wordt gekozen, kunnen ECG's met eigenschappen (features) van atriumfibrilleren gegroepeerd worden in één cluster, terwijl ECG's met de eigenschappen van een normaal sinusritme in een andere cluster gegroepeerd worden. In de studie van Nezamabadi et al. [NSHF23] vat men de meest toegepaste technieken samen in de context van ECG-analyse. En haalt aan dat K-means inderdaad toegepast wordt in deze context vanwege de eenvoudige implementatie en efficiëntie. Echter worden er twee relevante nadelen aangehaald: het vereiste om het aantal clusters vooraf te bepalen en de gevoeligheid voor ruis in de data.

2.3.2 Agglomerative clustering

Agglomerative clustering is een veelgebruikte vorm van hierarchical clustering en is een *bottom-up* variatie van hierarchical clustering. Dit houdt in dat initieel ieder datapunt beschouwd wordt als één individuele cluster. Bij elke iteratie worden vervolgens de twee dichtstbijzijnde (meest gelijkaardige) clusters geïdentificeerd op basis van een bepaalde afstandsmaat tussen de paren, zoals bijvoorbeeld de Euclidische afstand. De geïdentificeerde paren worden samengevoegd tot één nieuwe cluster. Dit wordt iteratief verder uitgevoerd voor alle clusters die aangemaakt worden en stop wanneer een bepaalde stopconditie wordt bereikt [sl].

In de studie van Nezamabadi et al. [NSHF23] worden opnieuw de voor- en nadelen besproken van dit algoritme binnen de context van ECG-analyse. Het grootste voordeel is dat ruis in de ECG-data deels automatisch behandeld wordt. Bovendien moet het aantal gewenste clusters (zoals de parameter k in K-means) niet vooraf bekend zijn. Hierarchical clustering biedt op dit vlak veel flexibiliteit door bijvoorbeeld gebruik te maken van een dendrogram waar men het hiërarchisch boomdiagram visueel kan waarnemen en kan beslissen hoeveel clusters er nodig zijn. Maar het is ook mogelijk om vooraf het aantal clusters te bepalen. Een nadeel van dit algoritme is dat het een hoge tijdscomplexiteit heeft. Naarmate de dataset toeneemt in grootte, stijgt de rekentijd. Dit maakt het algoritme minder geschikt voor grotere datasets.

2.3.3 Affinity propagation

Affinity propagation is een clustering algoritme dat zelf het meest optimale aantal clusters bepaalt. Dit algoritme gebruikt "berichten" en drie verschillende matrices om *exemplars* te selecteren. Exemplars zijn vergelijkbaar met de centroids in het K-means algoritme. Het belangrijkste verschil is dat centroids de middelpunten zijn van de clusters, terwijl exemplars effectief bestaande datapunten zijn uit de invoerdata die verkozen worden om de cluster te representeren.

Het algoritme start met het opstellen van een *similarity matrix*, waarin wordt bijgehouden in

welke mate alle paren van datapunten "lijken" op elkaar. Hieruit wordt een *responsibility matrix* $R(i,k)$ afgeleid die informatie bijhoudt over hoe geschikt een datapunt k is om een exemplar te zijn voor een ander datapunt i. Vervolgens wordt er een *availability matrix* $A(i,k)$ afgeleid uit de responsibility matrix die aangeeft hoe geschikt het is voor datapunt i om datapunt k als exemplar te kiezen, gebaseerd op wat de "meningen" zijn van andere datapunten. De responsibility en availability matrices worden iteratief geüpdatet totdat een stopconditie bereikt wordt. De "berichten" stellen de responsibility en availability matrices voor, omdat de datapunten conceptueel berichten met elkaar uitwisselen in de matrices. Tenslotte worden de exemplars geïdentificeerd door voor elk datapunt i te berekenen welk ander datapunt k de maximale som van $A(i,k) + R(i,k)$ oplevert. Als deze som maximaal is wanneer datapunt k, het datapunt i is, wordt het datapunt i aangeduid als exemplar. Het finale resultaat is een verzameling van deze geïdentificeerde exemplars en alle andere invoer datapunten die toegewezen worden aan de best overeenkomende exemplars (clusters) [sl] [FD07].

In de context van ECG-analyse heeft dit algoritme opnieuw als voordeel dat het aantal clusters niet vooraf gespecificeerd moet worden. Bovendien bepaalt het algoritme zelf hoeveel clusters er aangemaakt moeten worden op basis van hoe gelijkaardig de ECG-signalen zijn. Het nadeel dat in de studie vermeld wordt, is echter dat de schaalbaarheid gelimiteerd is vanwege de matrices die aangemaakt moeten worden. Deze matrices vergen hoge geheugen- en computationele kosten, aangezien er voor alle mogelijke paren van datapunten informatie bijgehouden moet worden. Dit vormt een aanzienlijke beperking bij het trainen van modellen op grote ECG-datasets. Een bijkomend nadeel is dat het gevoelig is voor ruis in de ECG-data [NSHF23]. Deze nadelen kunnen de nauwkeurigheid negatief beïnvloeden waardoor het mogelijks een slechtere keuze is binnen dit specifiek domein.

2.3.4 Spectral clustering

Alle eerder besproken clustering algoritmen gebruikten de invoerdata, in dit geval de ECG-data, rechtstreeks in het clustering algoritme. Spectral clustering zet deze data eerst om in een graafrepresentatie. Elk datapunt wordt voorgesteld als een knoop (*node*) in de graaf en het gewicht van iedere verbinding (*edges*) tussen de knopen stelt de gelijkenis tussen die twee knopen (twee datapunten) voor. Het doel van dit algoritme is om deze graaf op te splitsen in subgrafen of clusters, zodat de som van de gewichten van de verbindingen tussen verschillende clusters geminimaliseerd wordt, terwijl de verbindingen binnen de clusters zo sterk mogelijk blijven. Zoals in het affinity propagation algoritme wordt er een similarity matrix opgesteld die informatie bevat over de gewichten van de verbindingen tussen alle knopen. Een nadeel hiervan is opnieuw de hoge geheugen- en computationele kosten. Anderzijds biedt spectral clustering een belangrijk voordeel dat het effectief is bij hoog-dimensionale data. Een ander voordeel is dat dit algoritme robuust is tegen ruis in de data [NSHF23] [sl].

2.4 Gebruikte combinaties van preprocessing en clustering technieken

In dit onderdeel worden verschillende combinaties van preprocessing en clustering technieken besproken en vergeleken die bestudeerd en toegepast zijn in verschillende studies binnen het domein van ECG-analyse. Deze studies werden specifiek geselecteerd vanwege hun state of the art benadering binnen dit domein, waarbij het doel is om verschillende combinaties van unsupervised learning technieken en preprocessing technieken toe te passen om aritmieën effectief van elkaar te onderscheiden. In deze studies staan de clustering algoritmen (unsupervised learning) centraal waarbij diverse preprocessing technieken worden onderzocht en toegepast om de meest optimale resultaten te bekomen.

2.4.1 Studie 1: Rodriguez-Sotelo et al. (2007) [RSCFCD07]

In de studie van Rodriguez-Sotelo et al. [RSCFCD07] worden de volgende opeenvolgende stappen toegepast:

1. Hartslag detectie
2. Trace segmentatie
3. WT decompositie
4. Preclustering
5. Clustering algoritme

Hartslag detectie is de eerste stap die toegepast wordt op de ruwe invoer dataset. In deze studie wordt de MIT-BIH dataset gebruikt die publiek beschikbaar is. Typisch bestaan deze datasets uit ECG-opnames van hartslagen gedurende een bepaalde tijdsperiode per patiënt. Dergelijke opnames zijn continue signalen die bestaan uit meerdere opeenvolgende hartslagen. Het doel van deze stap is om elke individuele hartslag te identificeren en te extraheren, zodat elke hartslag een datapunt vormt die vervolgens de invoer vormt voor het trainen van het clustering model. Deze stap wordt initieel uitgevoerd door wavelettransformatie (WT) toe te passen op het ECG-signaal. Dit maakt het gemakkelijker om de QRS-complexen te detecteren om vervolgens de R-pieken te identificeren. In deze studie worden de R-pieken aangeduid met $m_{zc}(i)$, dus als $m_{zc}(i)$ de huidige R-piek is dan is $m_{zc}(i+1)$ de R-piek van de opeenvolgende hartslag.
Vervolgens wordt het startpunt van elke hartslag gedefinieerd als:

$$m_{zc}(i) - 0,25 * RR$$

en het eindpunt als:

$$m_{zc}(i) + 0,75 * RR$$

waarbij RR het interval is tussen de huidige R-piek en de R-piek van de volgende hartslag, aangeduid door $(m_{zc}(i), m_{zc}(i+1))$. Het signaal tussen elk berekend start- en eindpunt stelt één individuele hartslag voor.

Het is belangrijk om te vermelden dat individuele hartslagen in de praktijk niet altijd dezelfde lengte hebben. Een bepaalde hartslag kan bijvoorbeeld een langere duur hebben dan de vorige hartslag, terwijl beide nog steeds de kenmerken van een normaal sinusritme hebben. Om de data uit de vorige stap als invoer te kunnen gebruiken voor het clustering algoritme, wordt er een trace segmentatie stap uitgevoerd. In deze stap worden alle hartslagen op gelijke lengte gebracht door het signaal te resamplen, zodat elke hartslag een uniforme representatie krijgt.

De volgende stap is de feature-extractie, die in deze studie wordt aangeduid als WT decompositie. Het doel van deze stap is om de relevante features te extraheren uit de gesegmenteerde hartslagen uit de vorige stap, zodat ze gebruikt kunnen worden als invoer voor het clustering algoritme.
Preclustering is de voorlaatste stap in dit proces en omvat het uitfilteren van repetitieve hartslagen uit de oorspronkelijke dataset. Deze nieuwe verzameling van overblijvende hartslagen wordt in de studie aangeduid met de variabele M.
De laatste stap is de clustering stap waar vier verschillende clustering algoritmen toegepast worden op de verzameling M uit de vorige stap: K-means, *K-medians*, *HK-means* en *J-means*. Vervolgens worden de resultaten uit deze clustering algoritmen met elkaar vergeleken. Uit de resultaten blijkt dat J-means die de Manhattan afstand gebruikt (om de gelijkenis tussen datapunten te berekenen) de beste resultaten oplevert [RSCFCD07].

2.4.2 Studie 2: J.H. Abawajy et al. (2013) [AKC13]

In deze studie worden de volgende vier stappen toegepast:

1. Dimensionality reduction

2. Clustering algoritme
3. Consensus functies
4. *Fast supervised classification*

In de eerste stap wordt er geëxperimenteerd met drie verschillende dimensionality reduction technieken zoals principal component analysis (PCA), *rank correlation coefficient* (RCC) en *B-splines* met als doel om de grootte van de invoerdata te reduceren en om de irrelevante features uit te filteren, zodat zowel de efficiëntie als de kwaliteit van het clustering algoritme verbeteren. De output van deze stap vormt direct de invoer voor de tweede stap, de clustering.

In deze stap werd er opnieuw geëxperimenteerd met verschillende clustering algoritmen zoals *SimpleKMeans*, *Cobweb clustering*, *expectation maximization (EM) clustering* en *FarthestFirst clustering*. Het onderzoek in deze studie stopt echter niet na de clustering. De resultaten van deze clustering algoritmen worden gebruikt om tot één finale *consensus* clustering te komen. Dit betekent dat de resultaten van de verschillende clustering algoritmen niet direct met elkaar vergeleken worden, maar deze worden gecombineerd om één finale clustering als resultaat te bekomen.

In de derde stap wordt er gebruik gemaakt van een functie die de vier onafhankelijke resultaten uit de clustering algoritmen neemt als invoer en deze combineert tot één eindverzameling van clusters. Dergelijke functies worden in deze studie *consensus functions* genoemd. Er worden vier verschillende consensus functies gedefinieerd in dit onderzoek om de performantie en resultaten van iedere functie met elkaar te vergelijken. De vier onderzochte consensus functies zijn *Cluster-Based Graph Formulation (DBGF)*, *Hybrid Bipartite Graph Formulation (HBGF)*, *K-Means Consensus Function (KMCF)* en *Instance-Based Graph Formulation (IBGF)*. Aangezien deze functies buiten de scope van deze thesis vallen, worden ze verder niet in detail besproken.

De vierde en laatste stap in deze studie is Fast Supervised Classification. Hoewel de consensus functies al bruikbare resultaten opleveren, wordt in deze stap onderzocht hoe het model efficiënter en schaalbaarder toegepast kan worden op ECG datasets met een hoge dimensionaliteit. Hiervoor worden verschillende supervised learning algoritmen getraind en gevalideerd op de volledige ECG dataset, waaronder *BayesNet*, *NaiveBayes*, *SMO*, *LibLINEAR* en *LibSVM*. Ten slotte worden alle mogelijke combinaties van deze supervised learning algoritmen, dimensionality reduction technieken, consensus functies en verschillende hoeveelheden clusters met elkaar vergeleken.

De evaluatie van deze combinaties gebeurt aan de hand van de *F-measure* (F-score). F-measure is een maat die de balans weergeeft tussen precisie en recall om de performantie van deze combinaties te evalueren. F-measure is dus het harmonisch gemiddelde van precisie en recall en wordt als volgt gedefinieerd [AKC13] [Goo25]:

$$\text{F-measure} = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}}$$

Waarbij:

- **Precisie**: het aandeel van correct geclassificeerde positieve voorspellingen ten opzichte van alle positieve voorspellingen.
- **Recall**: het aandeel van correct geclassificeerde positieve voorspellingen ten opzichte van alle echte positieve gevallen.

Een hoge F-score duidt op een sterke performantie van het model, aangezien dit betekent dat zowel de precisie als de recall hoog zijn.

2.4.3 Studie 3: J.L. Rodriguez-Sotelo et al. (2010) [RSDTPO$^+$10]

In deze studie worden de volgende stappen toegepast:

1. Ruisfiltering en segmentatie
2. Feature-extractie
3. Dimensionality reduction
4. *Gaussian Expectation-Maximization Clustering* (GEMC)

Er wordt de MIT-BIH dataset gebruikt die ook in andere studies wordt gebruikt. In deze studie worden er twee soorten ruis geïdentificeerd: *power line interference* en baseline wander. Deze worden verwijderd uit de ECG-signalen uit de dataset door ruisfiltering technieken toe te passen. Vervolgens worden deze zuivere ECG-signalen gesegmenteerd door opnieuw de R-pieken te detecteren. Zoals eerder vermeld worden deze R-pieken gebruikt om individuele hartslagen te identificeren en vervolgens te extraheren. In dit onderzoek wordt er gebruikgemaakt van een venster met een vaste grootte van 200 ms om rond de R-pieken in de ECG-signalen de individuele hartslagen te extraheren.

De daaropvolgende stap is de feature-extractie stap. Aangezien ECG-signalen tijdsreeksdata zijn, is het efficiënter om een aantal relevante features te selecteren uit de gesegmenteerde hartslagen, zodat ze de invoer kunnen vormen voor het finale clustering algoritme. In tegenstelling tot de eerder besproken studies, wordt in deze studie een onderscheid gemaakt tussen drie soorten verzamelingen van features:

- *Prematurity* features
- Representatie features
- Morphologische features

Iedere verzameling bevat dus informatie over een specifiek soort feature. Prematurity features geven bijvoorbeeld meer informatie over de RR-intervallen, terwijl de representatie features meer informatie geven over de details in de ECG-signalen door deze signalen anders te representeren met behulp van wavelet- en *hermite* transformatie technieken. Morphologische features bevatten bijvoorbeeld informatie over de polariteit van QRS-complexen. Deze informatie is belangrijk omdat QRS-complexen een positieve of een negatieve polariteit kunnen hebben afhankelijk van het type hartritme.

De voorlaatste stap is dimensionality reduction en heeft in deze studie als doel om irrelevante features te elimineren en de computationele kost te reduceren. De gebruikte dimensionality reduction techniek is weighted-PCA. Zoals eerder vermeld, is dit een variant van het traditionele PCA algoritme. In tegenstelling tot het standaard PCA algoritme wordt hier initieel ondersteld dat alle features niet even belangrijk zijn. Dit algoritme bestaat uit twee fasen.
De eerste fase omvat het toewijzen van gewichten aan individuele features, op basis van specifieke formules in de studie. De tweede fase omvat het projecteren van de gewogen data door principal components te gebruiken.

Deze nieuwe feature set wordt vervolgens als invoer gebruikt in de finale clustering fase, die uit drie stappen bestaat:

1. Schatting van het aantal clusters.
2. Initialisatie van centroids.
3. *Gaussian Expectation-Maximization Clustering* (GEMC)

In de eerste stap wordt er een schatting gemaakt van het aantal clusters via *spectral analysis* en *affinity measure*. In de tweede stap wordt er een poging gedaan om de centroids zo goed mogelijk te initialiseren via het *J-H-means* algoritme (een variant van het K-means algoritme). De laatste stap is de finale clustering fase van de hartslagen door het Gaussian Expectation-Maximization Clustering algoritme (GEMC) erop toe te passen. GEMC is een *soft clustering* algoritme (ook wel *fuzzy clustering* genoemd), waarbij een datapunt niet strikt tot één cluster behoort (zoals bij exclusive clustering), maar een bepaalde kans heeft om tot meerdere clusters te

behoren [Dat25] [RSDTPO+10]. De werking van dit clustering algoritme wordt hier niet verder toegelicht, aangezien het niet wordt toegepast in de methodologie van deze thesis.

2.4.4 Een synthese

Het valt op dat studie 1 en studie 3 starten met de segmentatie stap, waarbij de samples in de dataset worden opgesplitst in individuele hartslagen. In studie 1 gebruikt men de wavelet transformatie voor segmentatie, terwijl in studie 3 een vaste venstergrootte gedefinieerd wordt om rond elke R-piek de corresponderende hartslag te extraheren. Deze tweede methode komt overeen met de segmentatie techniek die uitgebreid besproken werd in Sectie 2.2.2 en verder wordt toegepast in de implementatie van deze thesis.
In studie 2 is er geen sprake van segmentatie en worden de samples na dimensionality reduction direct als invoer gebruikt voor het clustering algoritme. Ook in studie 3 wordt dimensionality reduction toegepast voor de finale clustering stap, terwijl dit in studie 1 niet toegepast wordt.

In tegenstelling tot de eerste twee studies begint studie 3 met de ruisfiltering stap die in Sectie 2.2.1 in detail werd besproken. In studie 1 voegt men nog een trace segmentatie stap toe om de eerder gesegmenteerde hartslagen verder te verwerken. Zowel studie 1 als studie 3 hebben een feature-extractie stap waarbij studie 1 één soort algoritme toepast (WT decompositie) om de belangrijkste features uit de data te extraheren, terwijl studie 3 dieper ingaat op verschillende soorten features.

Voor de clustering stap worden in alle drie studies verschillende clustering algoritmen voorgesteld (vaak varianten van het K-means algoritme, maar ook andere). Hoewel studie 1 en studie 3 een aantal gemeenschappelijke stappen volgen zoals segmentatie, feature-extractie en algemene methodologie, valt het bij studie 2 op dat de clustering stap als een tussenstap wordt gebruikt. In studie 2 worden bovendien unieke technieken toegepast, zoals consensus functies en fast supervised classification algoritmen om de efficiëntie te verhogen.

Door deze studies te analyseren, biedt deze literatuurstudie verder een overzicht van verschillende combinaties van state-of-the-art benaderingen. Dit overzicht toont aan dat er niet één pad bestaat, maar talloze andere combinaties en paden. De onderzoeksvraag van deze thesis is: **Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?** Hierdoor is het waardevol om zowel de bestaande als nieuwe combinaties van technieken te onderzoeken en met elkaar te vergelijken.

2.5 ECG databases uit studies

Er zijn verschillende publieke ECG databases beschikbaar die gebruikt kunnen voor experimentele doeleinden. Deze databases worden zowel in de eerder besproken studies als in andere beschikbare literatuur binnen de context van ECG-analyse toegepast. In Tabel 2.1 wordt een overzicht van deze databases gegeven samen met hun bijbehorende specificaties.

				opname duur		
naam	# patiënten	# opnames	# afleidingen	min	max	# klassen
MIT-BIH Arrhythmia [MM01]	47	48	2	1800	1800	2
AF classification challenge 2017	8528	8528	1	9	61	4
SPH dataset	24 666	25 770	12	10	60	59
CU_SPH dataset	10 646	10 646	12	10	10	11
PTB-XL [WSB+20]	18 885	21 837	12	10	10	71
PTB-XL AFib [ptb21]	5698	6428	12	10	10	3

Tabel 2.1: Overzicht van publiek beschikbare ECG datasets uit [GGM+23] en PTB-XL AFib

De tabel geeft een overzicht van vijf verschillende databases samen met hun specificaties zoals het totaal aantal patiënten, opnames, klassen, afleidingen en de minimale en maximale duur van een opname. Opvallend is dat sommige databases meer opnames bevatten dan het aantal patiënten. Dit komt omdat er voor sommige patiënten meerdere opnames zijn gemaakt met telkens andere resultaten.

De MIT-BIH database bestaat uit 48 patiënten. De ECG-opname van elke patiënt bestaat uit twee soorten bestanden: een annotatie bestand en een CSV bestand. Het CSV bestand bevat een kolom met het sample nummer en twee kolommen met twee verschillende afleidingen. Zoals eerder vermeld, zijn de afleidingen de effectieve ECG-signalen die tussen twee elektroden worden gemeten. De MIT-BIH Arrhythmia database bevat dus twee afleidingen (typisch verkrijgbaar met drie elektroden), terwijl de PTB-XL database twaalf afleidingen bevat (verkrijgbaar met tien elektroden). Elke rij in het CSV bestand stelt één individueel gemeten sample voor, voor beide afleidingen. In het annotatie bestand worden de individuele hartslagen aangeduid door voor elke hartslag de tijdstempel, het type hartritme en het sample nummer bij te houden van de R-piek. Het type hartritme wordt aangeduid door één letter. De letter 'N' verwijst bijvoorbeeld naar een normaal sinusritme [MM01].

Tabel 2.1 geeft aan dat de minimale en maximale duur van een opname in de PTB-XL dataset beide tien seconden zijn. Dit geeft vervolgens aan dat elke opname in de dataset exact 10 seconden duurt. Verder bevat deze dataset 71 klassen die meer informatie bieden over het soort hartslag. Deze klassen bestaan niet enkel uit hartritme-klassen, maar ook uit de diagnostische en vormgerelateerde klassen. Er zijn 44 diagnostische klassen, waarin typische hartcondities gegroepeerd zijn in vijf superklassen (zoals NORM, MI, CD, STTC, HYP) met verdere onderverdeling in meerdere subklassen. De vormgerelateerde klassen beschrijven kenmerken met betrekking tot de structuur en vorm van ECG-signalen, zoals afwijkingen in het QRS-complex, hoge voltages of andere morfologische kenmerken [WSB$^+$20] [GGM$^+$23].

In Tabel 2.1 werd nog een extra database toegevoegd: de PTB-XL AFib database. Dit is een variant en een subset van de originele PTB-XL database. In tegenstelling tot de 71 klassen die de oorspronkelijke dataset bevat, heeft deze subset enkel drie klassen. Deze klassen zijn SR (sinusritme), AF (atriumfibrilleren) en VA (verzameling van andere aritmieën). In de documentatie van deze dataset wordt vermeld dat, de ECG-opnames een sample rate hebben van 500 Hz, zoals in de oorspronkelijke dataset. Dit betekent dat per seconde 500 metingen (samples) worden geregistreerd per afleiding. Verder blijkt uit de documentatie dat van de 6 428 opnames in totaal, er 2 000 tot klasse SR behoren, 1 587 tot AF en 2 841 tot VA [ptb21].

Hoofdstuk 3

Methodologie

Het doel van deze thesis is om op een creatieve manier andere combinaties van methodes te onderzoeken en te vergelijken dan de combinaties die in de literatuur besproken zijn. In dit hoofdstuk worden eerst een aantal technieken besproken, zoals preprocessing, feature-extractie en dimensionality reduction, die binnen verschillende combinaties toegepast zijn in de implementatie.

3.1 Data

Om het probleem initieel zo klein mogelijk te houden, is de PTB-XL AFib database gekozen die eerder in de literatuur besproken werd. Hierdoor is het mogelijk om het aantal klassen beperkt te houden, aangezien deze database enkel drie klassen bevat. In de toekomst kan deze uitgebreid worden door databases te nemen met meer klassen om meerdere aritmieën te classificeren en te onderscheiden van elkaar. Daarnaast is de tijdsduur van elke opname consistent waardoor preprocessing over het algemeen gemakkelijker kan verlopen. Zoals eerder vermeld zijn er 6 428 ECG-opnames (12-lead), waarvan 2 000 behoren tot klasse SR, 1 587 tot klasse AF en 2 841 tot klasse VA. Alle ECG-opnames duren exact tien seconden met een sample rate van 500 Hz. Iedere opname bestaat in totaal uit 60 000 samples (500 samples x 10 seconden x 12 afleidingen). Als gevolg bevat de volledige dataset in totaal 385 680 000 datapunten (6 428 x 60 000) [ptb21].

3.2 Preprocessing

Zoals besproken in de literatuurstudie begint preprocessing typisch met ruisfiltering en het identificeren van R-pieken om iedere hartslag als één datapunt te kunnen beschouwen. In deze thesis wordt er echter afgeweken van deze aanpak en wordt elke ECG-opname als één datapunt beschouwd, zoals in studie 2 van Sectie 2.4. Deze keuze is gemaakt doordat in de PTB-XL database **elke opname** gelabeld is door het type aritmie, terwijl in de MIT-BIH database **elke individuele hartslag** gelabeld is door het type hartritme. Vervolgens leek het interessanter om te clusteren op de volledige ECG-opnames, niet alleen om creatief af te wijken van de traditionele methodes, maar ook omdat, zoals aangetoond in Figuur 2.3, atriumfibrilleren voornamelijk getypeerd wordt door onregelmatige frequentie van de QRS-complexen in plaats van specifieke patronen binnen elke individuele hartslag.

3.2.1 Flatten

Een clustering algoritme neemt typisch tweedimensionale data als invoer, waarbij één dimensie de samples of de datapunten representeert (in deze context elke ECG-opname) en de andere dimensie de features van deze datapunten representeert (alle individuele amplitude waarden

binnen iedere ECG-opname). De ingeladen dataset is initieel een driedimensionale dataset van 6 428 x 5 000 x 12 (6 428 ECG-opnames, 5 000 samples per afleiding en twaalf afleidingen). Het doel is om de informatie uit de afleidingen te gebruiken als invoer voor het clustering algoritme. Aangezien de clustering algoritmen enkel tweedimensionale data aannemen als invoer, moet de driedimensionale dataset eerst omgezet worden naar een tweedimensionale dataset. Dit wordt gedaan door de ECG-opname uit iedere afleiding achter elkaar te "plakken". Dit resulteert in 60 000 features (5 000 samples x 12 leads) voor iedere opname. Met andere woorden worden de laatste twee dimensies van de dataset "geflattened". Het resultaat van dit proces is een tweedimensionale dataset van 6 428 x 60 000 (6 428 opnames en 60 000 features).

De resulterende dataset bevat echter een te grote hoeveelheid aan het aantal features, wat ongewenste resultaten kan opleveren. Later in deze thesis zal er in detail besproken worden hoe dit een probleem vormt voor de feature-extractie en dimensionality reduction stap. Daarom is het essentieel om het aantal features te reduceren of enkel de relevante features te selecteren om de grootte van de dataset te reduceren zonder veel belangrijke informatie te verliezen. Het doel is om het aantal features van 60 000 te verminderen naar 10 000 features, zodat de gereduceerde dataset gebruikt kan worden als invoer in de dimensionality reduction of feature-extractie stap. In deze thesis worden twee methodes hiervoor voorgesteld: *downsampling* en *sliding window*.

3.2.2 Downsampling

Typisch betekent downsampling in signaalverwerking dat de sample rate gereduceerd wordt en als gevolg de "resolutie" van het signaal afneemt. Dit wordt typisch gedaan door slechts één waarde bij te houden van elk n-de sample in de data door bepaalde filters toe te passen zoals low-pass filters [dow24]. Een techniek die binnen deze thesis en in de context van ECG-analyse interessant is, is de moving average filter, omdat deze niet enkel toegepast wordt om de grootte van de dataset te reduceren, maar voornamelijk gebruikt wordt om ruis te verwijderen. De moving average filter kan ruis uit het ECG-signaal verwijderen, terwijl scherpe signaalkenmerken behouden blijven. Daarnaast is deze filter één van de ruisfiltering technieken die in de literatuurstudie besproken werd in Sectie 2.2.1.

Wiskundig wordt de moving average filter gedefinieerd als:

$$x(i) = \frac{1}{M} \sum_{j=0}^{M-1} x(i+j)$$

waarbij $x(i)$ de nieuwe gefilterde waarde is op positie i in het signaal. M is vervolgens de grootte van het venster dat vooraf bepaald wordt en bepaalt hoeveel opeenvolgende elementen genomen worden om het gemiddelde van die elementen te berekenen [Kri21]. Dit is een iteratief proces waarbij het venster over de data "geschoven" wordt om telkens het gemiddelde te berekenen van de elementen binnen dat venster.
Wanneer enkel de moving average filter toegepast wordt op de data, wordt meestal slechts een klein aantal elementen aan de randen van de data verminderd. Echter is de hoeveelheid data die verwijderd wordt onvoldoende groot om als invoer gebruikt te worden voor de feature-extractie en dimensionality reduction stappen. Daarom worden in de implementatie van deze thesis moving average en downsampling gecombineerd om ruis uit het signaal te verwijderen, terwijl de grootte van de data voldoende gereduceerd wordt. Het belangrijkste verschil tussen enkel de moving average filter en de combinatie van moving average met downsampling is dat het venster niet telkens één plaats opschuift maar telkens M aantal plaatsen in de dataset. Dit resulteert in een significante vermindering van de grootte van de dataset. Zoals eerder vermeld is het doel van deze techniek om het aantal features van 60 000 te reduceren naar 10 000 features. In de implementatie wordt hiervoor een venstergrootte $M = 6$ gedefinieerd.

3.2.3 Sliding window

De sliding window aanpak komt niet rechtstreeks uit de literatuur en is een tweede poging om op een creatieve manier de grootte van de dataset van 60 000 features te verkleinen naar een dataset van 10 000 features. Het idee hierachter is dat bij downsampling de originele data gladgestreken wordt waardoor de exacte originele waarden niet behouden blijven. Dit is oorspronkelijk geen probleem aangezien de belangrijke informatie in de data behouden blijft. Toch wordt er geprobeerd om de volledige originele data als invoer mee te geven aan het dimensionality reduction en feature-extractie algoritme dat gebruikt wordt in de volgende stap. In tegenstelling tot de vorige aanpak wordt hier geen gebruikgemaakt van filters of wiskunde formules. Elke ECG-opname van 60 000 features wordt in volgorde opgesplitst in segmenten van 10 000 features. Vervolgens wordt ieder segment apart als invoer gebruikt voor het dimensionality reduction en feature-extractie algoritme.

3.2.4 Normalisatie

Een ECG-signaal heeft variërende amplitudewaarden. Normalisatie van deze data houdt in dat de te grote amplitudewaarden gedempt worden. Deze grote waarden ontstaan bijvoorbeeld door golven in het ECG-signaal zoals het QRS-complex. Wanneer men zonder normalisatie bijvoorbeeld deze grote golven verwijdert uit het signaal, blijven de amplitudewaarden relatief klein en omdat er geen uitschieters zijn, is de variatie binnen deze kleine waarden significant genoeg voor het model dat op die data traint. Als het model echter traint op data met grote golven, verliest het model de relevantie van die kleine amplitudewaarden, omdat de grote golven dominanter en belangrijker zijn voor het model door het verschil van deze waarden en de kleinere waarden. Daarom is het essentieel om deze waarden te normaliseren of schalen, zodat de grote golven niet volledig hun betekenis verliezen, maar de kleinere waarden ook worden beschouwd door het model. Dit is een essentiële stap voordat de dataset als invoer gebruikt kan worden in de trainingsfase van een model. In de implementatie van deze thesis gaat dit voornamelijk over het normaliseren van de amplitudewaarden naar een bereik van [-1,1] [IMS+24].

In de studie van et al. Lucas B.V. de Amorim [dACC23] worden de volgende vijf bekendste en diverse *scaling* technieken besproken:

- *Standard Scaler*
- *Min-max Scaler*
- *Maximum Absolute Scaler*
- *Robust Scaler*
- *Quantile Transformer*

In deze thesis wordt de Min-max Scaler gekozen, omdat deze techniek traditioneel gebruikt wordt om de waarden te schalen naar het bereik van [0,1] door de volgende formule toe te passen:

$$x_i' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}}$$

waarbij x_i de oorspronkelijke waarde van het datapunt is, $x_{\min}$ de minimumwaarde in de dataset, $x_{\max}$ de maximumwaarde in de dataset en x_i' de geschaalde (genormaliseerde) waarde van x_i.

Zoals eerder aangehaald, wordt een ECG-signaal vaak geschaald naar een bereik van [-1,1]. Hierdoor bestaat er een variant van deze formule voor een willekeurig bereik van [a,b]:

$$x_i' = a + \frac{(x_i - x_{\min})(b - a)}{x_{\max} - x_{\min}}$$

waarbij a en b respectievelijk de nieuwe onder- en bovenschaalgrenzen van het bereik zijn.

Deze formule wordt vaak gebruikt door machine learning onderzoekers om de waarden te schalen naar een bereik van [-1,1]. Dit resulteert uiteindelijk in de volgende specifieke formule [dACC23]:

$$x'_i = \frac{2(x_i - x_{\min})}{x_{\max} - x_{\min}} - 1$$

De `MinMaxScaler` klasse uit de *scikit-learn library* wordt in de implementatie gebruikt om de data te normaliseren. Deze klasse schaalt de waarden standaard naar een bereik van [0,1], maar door de `feature_range` parameter te definiëren kan het bereik aangepast worden naar [-1,1] [min].

3.3 Feature-extractie en dimensionality reduction

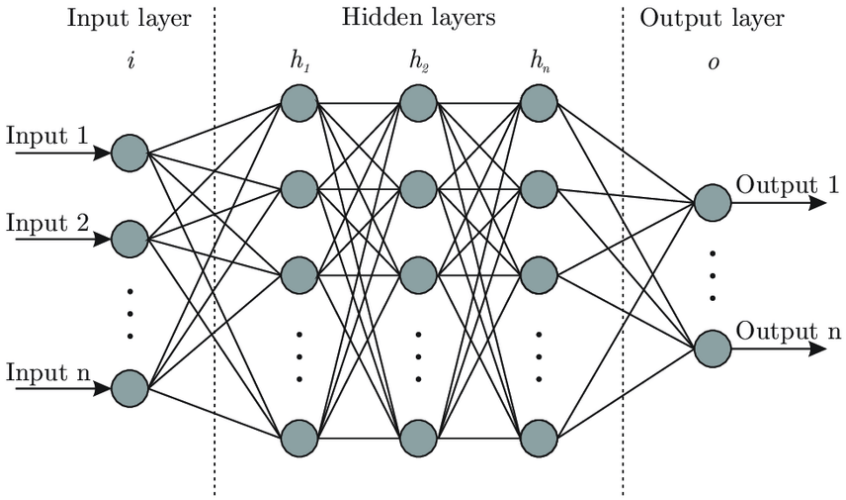
In de literatuur wordt er meestal een onderscheid gemaakt tussen de feature-extractie en dimensionality reduction stappen. Waarbij feature-extractie voornamelijk inhoudt dat er nieuwe features afgeleid worden (vaak gebaseerd op domeinkennis). Zo worden er in Sectie 2.4.3 bijvoorbeeld drie verschillende verzamelingen van features aangemaakt, gebaseerd op bepaalde karakteristieken van een ECG-signaal. Dimensionality reduction is een stap die na feature-extractie toegepast wordt in de studie en wordt gebruikt om het aantal features te reduceren, waarbij zoveel mogelijk relevante en belangrijke features behouden blijven. In deze thesis wordt (in tegenstelling tot die studie) er geprobeerd om op een creatieve manier beide feature-extractie en dimensionality reduction stappen te combineren. De techniek die hiervoor gebruikt wordt, is een deep autoencoder. Een autoencoder is een unsupervised learning techniek, aangezien het een dimensionality reduction techniek is. Een traditionele autoencoder is een unsupervised neuraal netwerk en traint op niet-gelabelde data zoals bij andere unsupervised learning technieken.

3.3.1 Neuraal netwerk

Een standaard neuraal netwerk (NN) is een machine learning techniek die de werking van het menselijke brein probeert te simuleren door de manier waarop neuronen in het menselijke brein met elkaar communiceren na te bootsen. Een NN bestaat uit meerdere lagen, ook wel *layers* genoemd, die elk een verzameling neuronen bevatten. Er bestaan drie soorten layers:

- Één input layer
- Één of meer verborgen layers (*hidden* layers)
- Één output layer

Dit wordt in Figuur 3.1 gevisualiseerd. De neuronen worden visueel voorgesteld als knopen of *nodes* die verbonden zijn met andere knopen in andere layers. Deze verbindingen worden *edges* genoemd.



Figuur 3.1: Voorbeeld schema van een *deep neural network* (DNN) [BGF17].

Iedere knoop in de input layer ontvangt een waarde uit de invoerdata. In beeldverwerking, waarbij een afbeelding de invoer is, stelt elke knoop één pixelwaarde voor uit die afbeelding. Als bijvoorbeeld een afbeelding uit 5 000 pixels bestaat, zouden er 5 000 knopen voorzien worden in de input layer.
De hidden layer is waar de verwerking van deze data plaatsvindt. Er wordt eerst een wiskundige functie gebruikt om de waarden van de opeenvolgende knopen in de huidige laag te berekenen. In Figuur 3.1 wordt dit proces schematisch gevisualiseerd.

Deze functie wordt al volgt gedefinieerd:

$$z = \sum_{i=1}^{n} (x_i \cdot w_i) + b$$

waarbij x_i de waarde is van de i-de knoop in de vorige layer die verbonden is met de knoop waar de waarde van berekend wordt in de huidige layer. En w_i het gewicht van de verbinding tussen die knopen. En de *bias* b die gezien kan worden als een extra constante, waarbij de drempelwaarde (*threshold*) van de activatiefunctie beïnvloed wordt [IBM21a] [AJDAS+22]. De berekeningen beginnen in de eerste hidden layer (h_1). De waarden van de knopen in h_1 worden berekend door de eerder besproken functie toe te passen met als invoer: de waarden van de knopen in de vorige layer (de input layer i), de gewichten van de verbindingen en de bias. Vervolgens worden activatiefuncties toegepast om het "afvuren" van de elektrische signalen vanuit de neuronen, zoals in het menselijke brein, te simuleren.

De activatiefunctie is een specifieke functie die toegepast wordt op elke knoop waar de gewogen som van berekend is via de bovenstaande functie. De activatiefunctie neemt z (de waarde van de knoop) uit de bovenstaande functie als invoer en "vuurt" een waarde af afhankelijk het type activatiefunctie dat gekozen is. Er bestaan verschillende activatiefuncties, maar in deze thesis worden de drie meest toegepaste activatiefuncties besproken:

- *ReLU (Rectified Linear Unit)*
- *Leaky ReLU*
- *Linear*

De Linear en Leaky ReLU functies worden in de implementatie van deze thesis toegepast.

De **ReLU** functie wordt als volgt gedefinieerd:

$$f(x) = \max(0, x)$$

Alternatieve notatie:

$$f(x) = \begin{cases} x & \text{als } x > 0 \\ 0 & \text{als } x \leq 0 \end{cases}$$

waarbij de waarde van de knoop z (de gewogen som) de invoer is voor de activatiefunctie $f(x)$. Als de waarde z groter is dan 0, wordt deze waarde "afgevuurd" naar de knopen in de volgende layer (in dit geval h_2).

De Leaky ReLU activatiefunctie is een variant van de standaard ReLU activatiefunctie. Zoals bij ReLU, wordt er bij een invoerwaarde groter dan 0, de invoerwaarde z zelf als uitvoer teruggeven. In tegenstelling tot de standaard ReLU functie wordt echter bij een invoerwaarde kleiner of gelijk aan 0, de waarde vermenigvuldigd door een kleine constante α. Als gevolg geeft een neuron met de waarde z die niet 0 is altijd een waarde terug die niet 0 is.

De **Leaky ReLU** activatiefunctie wordt als volgt gedefinieerd:

$$f(x) = \begin{cases} x & \text{als } x > 0 \\ \alpha x & \text{als } x \leq 0 \end{cases}$$

waarbij α typisch gelijk is aan 0,01.

De **Linear** activatiefunctie wordt als volgt gedefinieerd:

$$f(x) = ax$$

waarbij a een willekeurige constante (meestal $\alpha = 1$). Indien $a = 1$, geeft het neuron altijd exact de waarde z terug, aangezien er geen condities zijn zoals de ReLU en Leaky ReLU activatiefuncties.

Er bestaat geen vaste regel voor de keuze van de "correcte" activatiefunctie en is afhankelijk van de context waarin deze toegepast wordt. De standaard ReLU functie wordt over het algemeen het meest toegepast, terwijl de Leaky ReLU functie toegepast kan worden als het NN "dode" neuronen bevat. Dit zijn neuronen die altijd 0 als uitvoerwaarde hebben, waardoor ze niet bijdragen aan het trainingsproces van het model en uiteindelijk het finale resultaat negatief beïnvloeden. De Linear activatiefunctie wordt zelden toegepast om het NN te verbeteren. Deze functie wordt voornamelijk gebruikt waar interpretatie belangrijk is, zoals in de output layer [SSA17]. In deze thesis wordt de Leaky ReLU functie toegepast.

Verdere details over trainingsalgoritmen, zoals *backpropagation* en *gradient descent*, die tijdens het trainingsproces de gewichten aanpassen om de *loss* (het verschil tussen de gewenste en de werkelijke uitvoer) te minimaliseren, vallen buiten de scope van deze thesis [AJDAS+22].

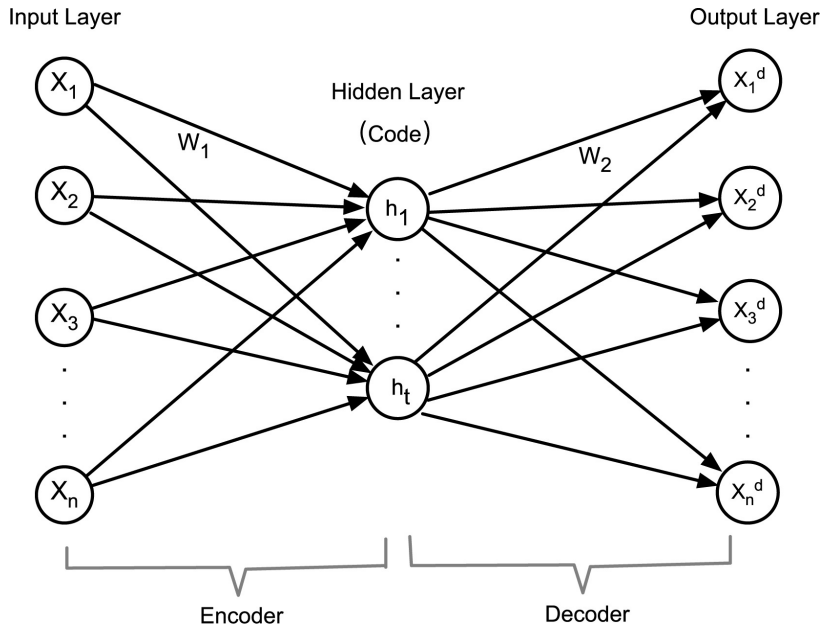
3.3.2 Autoencoder

Zoals eerder vermeld, is een autoencoder een techniek die in deze thesis toegepast wordt voor zowel feature-extractie als dimensionality reduction. Een autoencoder is een neuraal netwerk zoals beschreven in Sectie 3.3.1. Een belangrijk verschil met traditionele NN's is dat een autoencoder specifiek wordt getraind om de oorspronkelijke invoerdata te reconstrueren in de uitvoer, zonder dat deze reconstructie perfect is (intentioneel). Het model leert tijdens het trainingsproces welke aspecten en eigenschappen van de invoer essentieel zijn om zo goed mogelijk te repliceren [GBC16].

Een autoencoder bestaat typisch uit drie delen:

- Encoder
- Bottleneck, latent space of code
- Decoder

Dit wordt schematisch weergegeven in Figuur 3.2. Dit figuur toont aan dat de encoder begint bij de input layer (gelijk aan de input layer uit Sectie 3.3.1) en uiteindelijk eindigt bij de *bottleneck* (ook code of latent space genoemd). Het doel van de encoder is om de belangrijkste features te leren uit de invoerdata en is ook waar zowel dimensionality reduction als feature-extractie plaatsvinden. De output van de encoder wordt doorgegeven aan de bottleneck, die typisch een lagere dimensie (minder knopen) heeft.
Vervolgens probeert de decoder de output van de bottleneck om te zetten naar de oorspronkelijke invoerdata (dit is waar de reconstructie plaatsvindt). Daarom bevat een autoencoder evenveel knopen in de output layer als in de input layer, zoals weergegeven in Figuur 3.2. In deze thesis wordt specifiek de deep autoencoder, een variant van de traditionele autoencoder, toegepast.



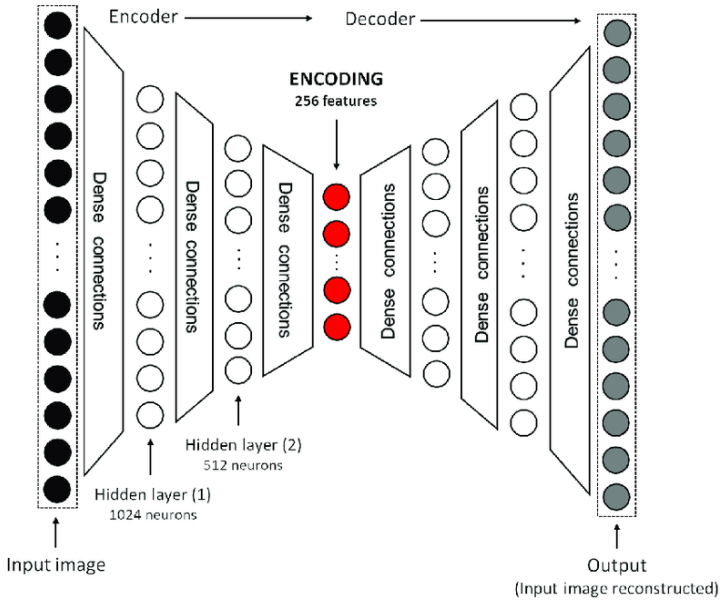
Figuur 3.2: Voorbeeld schema van traditionele autoencoder met n invoer- en output knopen en t knopen in de bottleneck (code) [LPL23].

3.3.3 Deep autoencoder

Een complexere variant van de standaard autoencoder is de deep autoencoder. Het verschil is dat een deep autoencoder een *deep neural network* (DNN) is. Wanneer de autoencoder meer dan één hidden layer heeft (dus minstens twee), wordt deze een deep autoencoder genoemd [Den14]. Zoals eerder vermeld in Sectie 3.3.1, worden de layers in het NN die tussen de input- en output layers zitten, hidden layers genoemd.

Opvallend is dat het aantal knopen in de opeenvolgende hidden layers van de encoder progressief afneemt naarmate het dichter bij de bottleneck komt. Daarnaast komt dit patroon opnieuw voor in de decoder in omgekeerde volgorde, vertrekkend vanuit de bottleneck layer. Als gevolg heeft een deep autoencoder typisch een symmetrische structuur. In Figuur 3.3 wordt een deep autoencoder schematisch weergegeven. In het schema valt het op dat de verbindingen tussen iedere layer, *dense connections* worden genoemd en betekent dat het DNN *fully connected* is.

Een fully connected NN betekent dat er verbindingen bestaan tussen alle mogelijke paren van knopen in de opeenvolgende layers.



Figuur 3.3: Voorbeeld schema van een deep autoencoder [JTJT20].

3.3.4 Training

In deze sectie wordt besproken hoe de deep autoencoder getraind en toegepast wordt in de implementatie van deze thesis. Op basis van de theorie van de vorige secties kan er geconcludeerd worden dat een encoder verantwoordelijk is voor het reduceren van het aantal dimensies. De bottleneck layer bevat vervolgens de nieuwe features die de gecomprimeerde representatie van de oorspronkelijke invoerdata voorstellen en bevat voldoende informatie voor de decoder om zo goed mogelijk de oorspronkelijke invoerdata te repliceren in de output layer van de deep autoencoder. Dit wordt ook wel een "deep autoencoder met compressie" genoemd, aangezien het totaal aantal features gereduceerd wordt en de oorspronkelijke features gecomprimeerd worden in de bottleneck layer.

Het proces om de gecomprimeerde representatie van de oorspronkelijke invoerdata te verkrijgen, bestaat uit de volgende stappen [Bro20]:

1. Trainen van de volledige deep autoencoder:
   (a) **Input layer** definiëren: 10 000 input knopen.
   (b) **Encoder layer** definiëren: respectievelijk 10 000, 10 000 en 5 000 knopen in de drie opeenvolgende hidden layers.
   (c) **Bottleneck layer** definiëren: bestaat uit 6 knopen.
   (d) **Decoder layer** definiëren: respectievelijk 5 000, 10 000 en 10 000 knopen in de drie opeenvolgende hidden layers (symmetrie).
   (e) **Output layer** definiëren: 10 000 output knopen.
2. De encoder wordt geïsoleerd uit de getrainde deep autoencoder en vervolgens opgeslagen.
3. De opgeslagen encoder wordt gebruikt om zowel trainingsdata als nieuwe data te comprimeren tot de zes meest relevante features.

In de implementatie gebruiken alle hidden layers in de deep autoencoder de Leaky ReLU activatiefunctie. Enkel de output layer gebruikt de Linear activatiefunctie. Zoals aangehaald in Sectie 3.3.1, betekent dit dat de waarden van de knopen in de output layer zonder berekeningen teruggegeven worden als eindresultaat van de deep autoencoder.

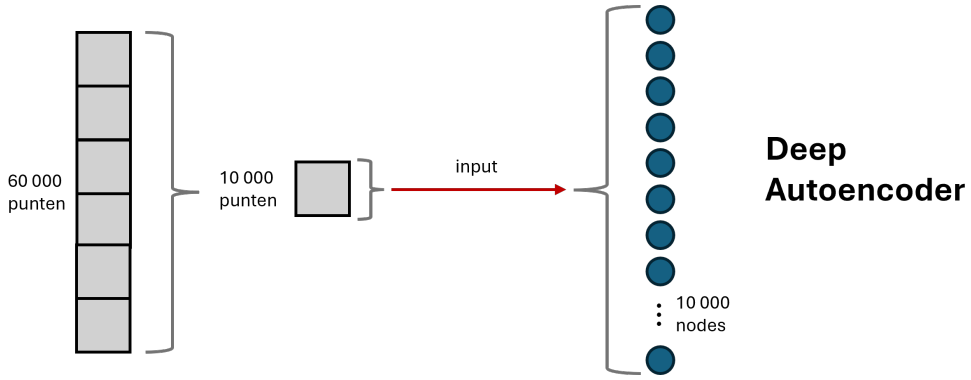
Voor de implementatie wordt er gebruikgemaakt van de *TensorFlow* library en meer specifiek de `Keras` API om de deep autoencoder te implementeren [ten]. De volgende *hyperparameters* zijn gespecificeerd voor het trainen van de deep autoencoder:

- Aantal *epochs*
- *Batch size*
- Aantal layers (encoder + decoder)
- Aantal knopen per layer
- *Optimizer*
- Activatiefuncties
- Loss functie
- *Learning rate*

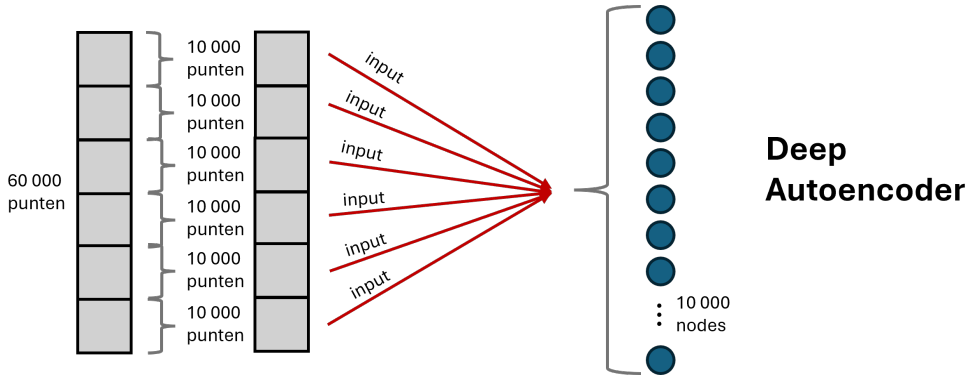
Deze hyperparameters zijn vooraf ingestelde parameters die het trainingsproces van het NN beïnvloeden. Zo geeft het aantal epochs aan hoe vaak het model de volledige trainingsdata doorloopt tijdens de training. De batch size bepaalt de hoeveelheid trainingsdata die het model per stap (batch) verwerkt binnen een epoch. Deze parameters worden *getuned* afhankelijk van de performantie van de modellen. Hierdoor bestaat er geen "correct" antwoord op wat de optimale waarden van deze hyperparameters precies moeten zijn. Hoewel deze parameters meer in detail besproken kunnen worden, vallen deze buiten de scope van deze thesis en worden daarom niet verder besproken.

3.3.5 Toepassing

Het doel is om de verwerkte ECG dataset uit de preprocessing stap in Sectie 3.2 te gebruiken als invoer voor het trainingsproces van de deep autoencoder. In Sectie 3.2 werden twee methodes besproken voor het reduceren van de grootte van de oorspronkelijke dataset: de downsampling en moving window methode. In deze sectie worden de toepassingen van deze twee methodes verder toegelicht. De reductie van de grootte van de dataset is een essentiële stap, omdat het niet mogelijk is om de volledige oorspronkelijke dataset als invoer te gebruiken voor de deep autoencoder vanwege de beperkingen van de beschikbare hardware tijdens het trainingsproces. Daarnaast is het belangrijk om een methode beschikbaar te stellen die het volledige proces schaalbaar maakt (voor grotere datasets). Beide methodes worden respectievelijk gevisualiseerd in Figuur 3.4 en 3.5.



Figuur 3.4: Schema verwerking van originele ECG data door combinatie van **down-sampling** en deep autoencoder.



Figuur 3.5: Schema verwerking van originele ECG data door combinatie van **sliding window** en deep autoencoder.

Figuur 3.4 illustreert hoe de oorspronkelijke ECG-data die uit 60 000 features bestaat, als invoer gebruikt kan worden voor de deep autoencoder die enkel 10 000 knopen bevat in de input layer door downsampling toe te passen. Het valt hier op dat 60 000 features omgevormd worden naar 10 000 features, terwijl Figuur 3.5 aantoont hoe de sliding window methode gebruikt wordt om 60 000 features op te splitsen in zes segmenten van 10 000 features, waarbij elk segment van 10 000 features als invoer gebruikt wordt voor de deep autoencoder.

3.4 Clustering

Clustering is de laatste stap in de implementatie van deze thesis. Hiervoor werden drie verschillende clustering algoritmen geselecteerd:

- K-means
- Agglomerative hierarchical clustering
- Spectral clustering

Zoals eerder aangehaald in de literatuurstudie in Sectie 2.3.1 is K-means één van de meest toegepaste en onderzochte clustering algoritmen en werd daarom geselecteerd als de eerste keuze voor deze stap. Agglomerative hierarchical clustering werd gekozen als tweede optie. Het doel van deze thesis is om diverse clustering algoritmen toe te passen om vervolgens de resultaten te vergelijken. Aangezien K-means behoort tot exclusive clustering algoritmen, werd de keuze gemaakt om een ander type clustering, hierarchical clustering, te selecteren voor de tweede optie.

Meer specifiek werd de agglomerative hierarchical clustering gekozen, omdat ruis in de ECG-data deels automatisch behandeld wordt door het algoritme. Ten slotte werd spectral clustering geselecteerd als de derde optie voor de gelijkaardige reden dat het algoritme robuust is tegen ruis in de invoerdata. Voor deze clustering algoritmen werd de scikit-learn library [sci] gebruikt. De drie clustering algoritmen uit deze library verwachten tweedimensionale data als invoer, waarbij de rijen de datapunten voorstellen en de kolommen de features. Aangezien de data verkregen uit de vorige stap bestaat uit 6 428 rijen en 6 kolommen, die al reeds genormaliseerd zijn door de normalisatie stap in Sectie 3.2.4, wordt deze tweedimensionale data gebruikt als invoer voor de drie clustering algoritmen. Deze algoritmen verwachten daarnaast als parameter het aantal gewenste clusters (K) in de invoer. Bij hierarchical clustering is dit optioneel, maar omdat het gewenste aantal clusters vooraf bekend is, wordt dit als invoer meegegeven.

De volgende vier onderzoeken werden uitgevoerd, waarvan de resultaten besproken worden in Sectie 4:

1. Clustering op volledige PTB-XL AFib dataset met $K = 3$.
2. Clustering op PTB-XL AFib dataset zonder klasse VA met $K = 2$.
3. Clustering op evenwichtige PTB-XL AFib dataset met $K = 3$
4. Clustering op evenwichtige PTB-XL AFib dataset zonder klasse VA met $K = 2$

Deze onderzoeken werden uitgevoerd voor alle drie clustering algoritmen met zowel de down-sampling als de sliding window methode in de preprocessing stap. De waarde van (K) werd gekozen op basis van het aantal klassen in de dataset. De oorspronkelijke PTB-XL AFib dataset bevat drie klassen. Maar om ook te experimenteren met een gewijzigde dataset, wordt klasse VA verwijderd in onderzoek 2 en 4. Aangezien de gewijzigde dataset als gevolg twee klassen bevat, wordt $K = 2$ gekozen. Daarnaast valt het op dat de oorspronkelijke PTB-XL AFib dataset niet volledig evenwichtig is, aangezien 2 000 samples behoren tot klasse SR, 1 587 tot klasse AF en 2 841 tot klasse VA. Daarom wordt de dataset in onderzoek 3 en 4 evenwichtig gemaakt door het aantal samples per klasse te reduceren tot 1 587, het aantal samples in klasse AF (1 587).

Andere clustering algoritmen zoals Affinity Propagation werden ook beschouwd. Deze werden echter niet in de thesis opgenomen, aangezien ze in de implementatie een groot aantal clusters genereerden die geen betekenis hebben. Er werden ook technieken toegepast om de meest optimale K te bepalen door de *elbow* methode en *Silhouette* score (deze wordt toegelicht in de volgende sectie) toe te passen. Echter werd er besloten om deze resultaten niet op te nemen in de thesis, omdat deze opnieuw clusters aanmaakten die geen betekenis hebben. Daarnaast lijkt deze methode niet relevant te zijn, aangezien er domeinkennis beschikbaar is over het verwachte aantal clusters.

3.5 Evaluatie

In hoofdstuk 4 worden de resultaten gedetailleerd besproken. Om te bepalen of de getrainde modellen gewenste resultaten opleveren is het belangrijk om de goede evaluatietechnieken te kiezen. Zoals aangehaald in het begin van deze thesis, ligt de nadruk op unsupervised learning, waarbij de modellen getraind worden op ongelabelde data. Echter voorziet de dataset die in de implementatie gebruikt wordt, echte labels voor elke ECG-opname. Hierdoor is het mogelijk om een bepaalde ***supervised clustering*** evaluatietechniek toe te passen, zoals het berekenen van *purity* scores. Purity is een metriek om de "zuiverheid" van de clusters uit het resultaat van een clustering algoritme te meten. Het geeft weer in welke mate een bepaalde klasse domineert in de clusters. Het drukt dus uit hoe goed de clusters uit een clustering algoritme een bepaalde klasse representeren.

Purity wordt gedefinieerd door de volgende formule [AA20] [vin22]:

$$\text{Purity} = \frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d|$$

waarbij N het totaal aantal datapunten voorstelt, M de verzameling van clusters, D de verzameling van echte labels, $|m \cap d|$ het aantal datapunten dat zowel in cluster m als in klasse d voorkomen.
Purity score heeft een bereik van [0,1] waarbij 1 (100%) een perfecte purity score is.

Naast de totale purity score worden ook de cluster purity scores berekend. Dit is de purity binnen elke cluster en is het percentage datapunten met het dominante echte label, ten opzichte van het totaal aantal datapunten in die cluster.

Dit wordt gedefinieerd door de volgende formule:

$$\text{Cluster Purity}(m) = \frac{\max_{d \in D} |m \cap d|}{|m|}$$

De resultaten uit de clustering algoritmen worden gevisualiseerd in Hoofdstuk 4 met behulp van een *contingency table*. Dit is een tabel waar de rijen de clusters representeren en de kolommen respectievelijk het volgende representeren:

1. Het totaal aantal datapunten per cluster.
2. Het aantal datapunten voor iedere echte klasse in de cluster.
3. De cluster purity score.

Ten slotte wordt ook de totale purity en Silhouette score weergegeven.

Een andere evaluatiemetriek die berekend wordt in Hoofdstuk 4 is de Silhouette score. In tegenstelling tot purity vereist deze techniek geen labels om de resultaten uit het clustering algoritme te evalueren. Het evalueert de kwaliteit van de clustering door zowel de intra-cluster als inter-cluster afstanden te beschouwen.

Eerst wordt de Silhouette score berekend voor ieder individueel datapunt x_i met behulp van de volgende formule [DFH19]:

$$s(x_i) = \frac{b(x_i) - a(x_i)}{\max\{a(x_i), b(x_i)\}}$$

waarbij

- $a(x_i)$, de gemiddelde afstand tussen het punt x_i en alle andere punten in dezelfde cluster (intra-cluster afstand).
- $b(x_i)$, de laagste gemiddelde afstand tussen het punt x_i en de punten in iedere andere cluster (inter-cluster afstand).

De finale stap is om de gemiddelde Silhouette score te berekenen. Deze score geeft de kwaliteit van de clustering weer. Deze score is altijd een waarde tussen [-1, 1]. Een score die -1 benadert, geeft aan dat de clustering slecht of incorrect is. Een score die 1 benadert, geeft aan dat de clusters goed gescheiden of duidelijk van elkaar te onderscheiden zijn (de datapunten liggen dichtbij hun eigen cluster en ver van andere clusters). Een score rond 0 betekent dat het onduidelijk is of punten bij de ene of bij de andere cluster horen.

De gemiddelde Silhouette score wordt berekend met de volgende formule:

$$s_k = \frac{\sum_{i=1}^{n} s(x_i)}{n}$$

waarbij $s(x_i)$ de Silhouette score is van datapunt x_i (uit de vorige formule) en n het totaal aantal datapunten. De bovenstaande formule is dus de som van de Silhouette score van ieder datapunt gedeeld door het totaal aantal datapunten.

3.6 Extra implementatie

De focus van de implementatie in deze thesis ligt op het clusteren van volledige ECG-opnames uit varianten van de PTB-XL AFib dataset. Echter wordt in deze thesis ook een extra implementatie besproken waarbij de MIT-BIH dataset wordt gebruikt in plaats van de PTB-XL AFib dataset. Hierbij worden gelijkaardige stappen uitgevoerd met enkele aanpassingen om te experimenteren met verschillende implementatiemogelijkheden, zodat de resultaten hieruit vergeleken kunnen worden met de resultaten van de vorige implementatie.

Het belangrijkste verschil tussen deze aanpak en de vorige aanpak is de granulariteit. In tegenstelling tot de vorige aanpak waar de volledige ECG-opnames gebruikt werden als invoer, wordt er in deze tweede aanpak op een lager niveau gekeken, waarbij individuele hartslagen uit de ECG-opnames als invoer gebruikt worden. Dit is eveneens een veelgebruikte aanpak in de literatuurstudie.

3.6.1 Data

Zoals eerder aangehaald, wordt in deze implementatie de MIT-BIH Arrhythmia dataset gebruikt en bestaat uit ECG-opnames van 48 patiënten, waarbij iedere opname gelabeld is beginnend bij label 100. In de implementatie wordt de eerste opname in deze dataset gebruikt, wat overeenkomt met de ECG-opname van de eerste patiënt in de dataset, gelabeld met het nummer 100 [MM01].
De opname van deze patiënt bestaat uit 2 239 normale individuele hartslagen, gelabeld met "N" (label voor normaal sinusritme), en 33 abnormale individuele hartslagen, gelabeld met "A" (symbool voor *atrial premature beat*) in deze dataset. Atrial premature beat is een type aritmie dat typisch geen ernstige gevolgen heeft en wordt daarom ook niet vermeld in Tabel 1.1. In deze implementatie wordt een poging gedaan om deze 33 abnormale hartslagen te onderscheiden van de 2239 normale hartslagen.

3.6.2 Preprocessing en clustering

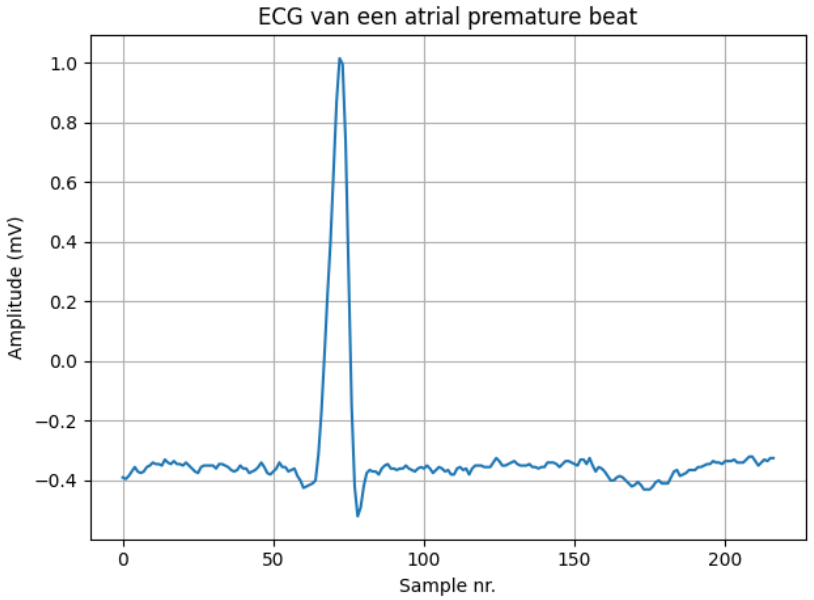
Het idee achter de preprocessing stap is grotendeels gelijkaardig aan de preprocessing stap uit de vorige implementatie. Het grootste verschil ligt in de granulariteit, waardoor deze stap op een aantal methodes afwijkt van de vorige aanpak. In de literatuurstudie wordt vermeld dat men QRS-detectie algoritmen toepast om de individuele hartslagen te detecteren in de ECG-opnames, zodat vervolgens deze hartslagen geëxtraheerd kunnen worden.

In deze implementatie is de MIT-BIH dataset bewust gekozen omdat deze annotatie bestanden bevat. In het annotatie bestand van label 100, zijn de R-pieken reeds geannoteerd samen met de exacte tijdstempels en het overeenkomende sample nummers in de ECG-opname. Hierdoor is het mogelijk de QRS-detectie stap over te slaan en direct naar de segmentatie stap over te gaan. Het is het belangrijk om te vermelden dat in de praktijk bij unsupervised learning, deze annotatie bestanden meestal niet beschikbaar zijn, waardoor QRS-detectie algoritmen wel noodzakelijk zijn. Toch wordt QRS-detectie overgeslagen en gefocust op andere preprocessing stappen, aangezien de R-piek detectie "perfect" is door het annotatie bestand en de nadruk van de onderzoeksvraag zit op het vergelijken van clustering algoritmen.

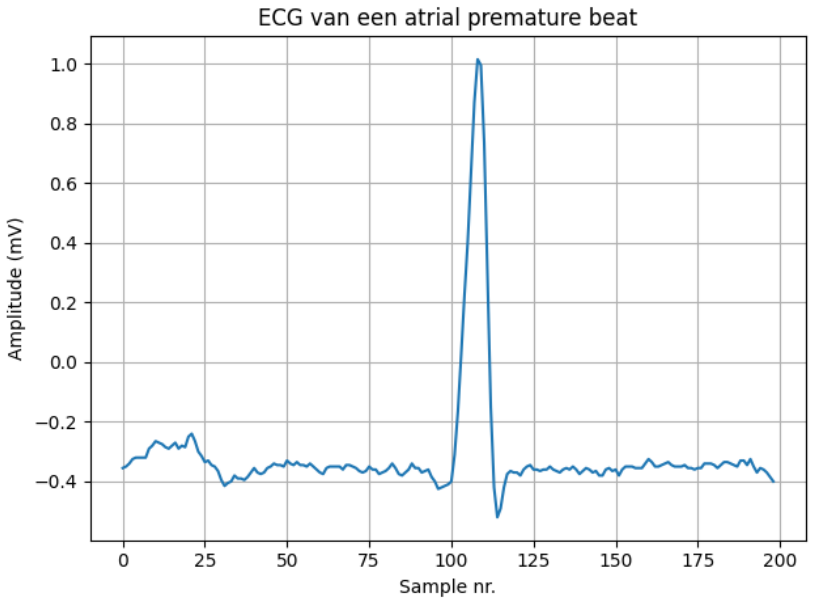
Segmentatie is een preprocessing techniek die in de vorige implementatie niet beschouwd werd, maar hier wel een cruciale stap is. In de literatuurstudie wordt segmentatie typisch uitgevoerd door een venster te definiëren rond de R-piek om het corresponderende hartslagsegment te extraheren. In deze thesis wordt (zoals in de literatuur) 200 ms voor de R-piek genomen als startpunt en 400 ms na de R-piek als het eindpunt van de hartslag. Het annotatie bestand

geeft echter enkel de tijdstempels van de R-pieken en niet van andere datapunten in de ECG-opname. Hierdoor is het niet mogelijk om de hartslagen direct te extraheren. Daarom wordt er in de implementatie eerst berekend hoeveel samples overeenkomen met deze tijdsintervallen. De dataset heeft een sample rate van 360 Hz en betekent dat er 360 datapunten worden opgenomen per seconde. Omgerekend komt dit overeen met 36 datapunten per 100 ms. Er werd eerst geprobeerd om een venstergrootte van [-200, 400] ms. te nemen volgens de studie van Filipe Canento et al. (2013) [CLSF13] met R-piek als referentiepunt. Dit komt overeen met 72 samples voor en 144 samples na de R-piek. De hartslag uit dit venster bevat dus in totaal 216 samples. In Figuur 3.6 wordt een hartslag in de ECG-opname gevisualiseerd die op deze manier gesegmenteerd werd.

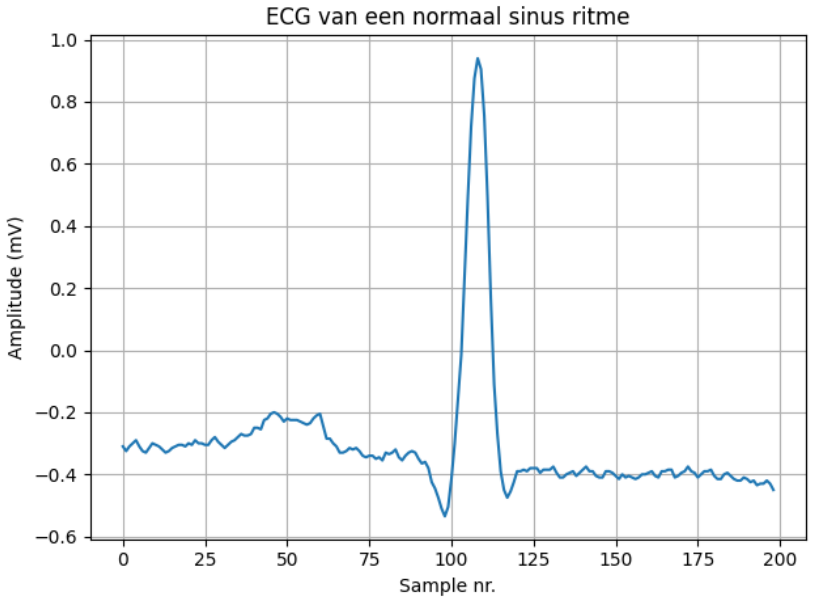
Om te valideren of de gekozen vesntergrootte geschikt is, is het belangrijk om na te gaan of de essentiële kenmerken van atrial premature beat erin voorkomen. Een atrial premature beat wordt typisch gekenmerkt door een P-golf die vroeger aanwezig is dan de P-golf van een normaal sinusritme [HY]. In Figuur 3.6 valt het op dat de P-golf niet zichtbaar is in het ECG. Daarom wordt er een tweede poging gedaan met een aangepaste venstergrootte van [-300, 250] ms. Deze grootte is opnieuw gebaseerd op de literatuur, waarbij een venstergrootte van [-300, 40] ms wordt gedefinieerd voor de segmentatie van enkel de P-golf en [-300, 250] ms voor de volledige hartslag [GIMC21]. De segmentatie van de volledige hartslag komt overeen met 108 samples voor en 90 samples na de R-piek. Het resultaat wordt weergegeven In Figuur 3.7. Het valt het op dat de P-golf inderdaad in de grafiek opgenomen wordt (ongeveer tussen samples [0, 30]). In Figuur 3.8 wordt ook het ECG van een normaal hartritme gevisualiseerd (met dezelfde venstergrootte) waar de P-golf zich ongeveer tussen samples [30, 70] bevindt. Dit bevestigt dat de P-golf inderdaad vroeger begint bij atrial premature beat dan bij een normaal sinusritme (deze grafieken zijn ruwe ECG-signalen waar de ruis nog niet gefilterd is, maar de morfologische kenmerken zijn nog steeds zichtbaar).



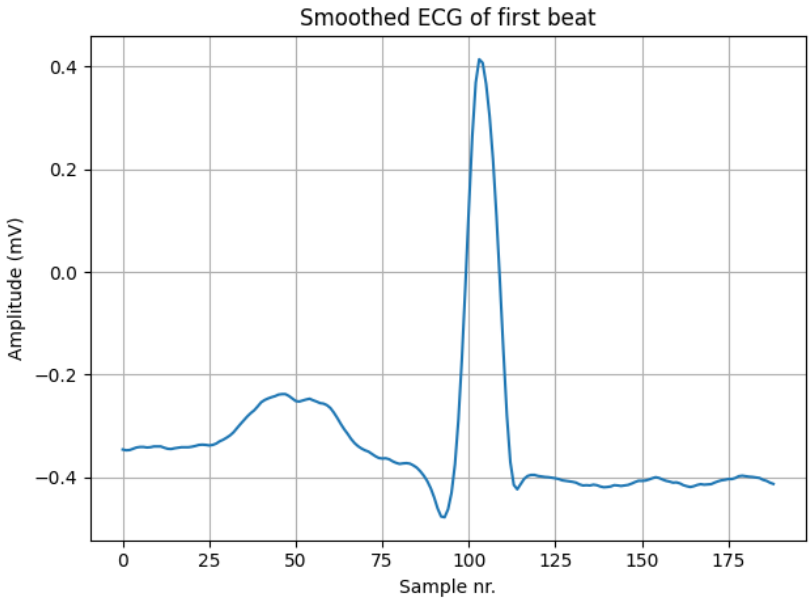
Figuur 3.6: ECG van één willekeurige hartslag met label "A"(atrial premature beat) met venster grootte [-200, 400] ms.



Figuur 3.7: ECG van één willekeurige hartslag met label "A"(atrial premature beat) met venster grootte [-300, 250] ms.



Figuur 3.8: ECG van één willekeurige hartslag met label "N"(normaal sinusritme) met venster grootte [-300, 250] ms.



Figuur 3.9: ECG uit Figuur 3.8 na ruisfiltering (venster grootte = 10).

Er wordt eerst een naïeve poging gedaan, waarbij de gesegmenteerde hartslagen als invoer meegegeven worden aan de drie clustering algoritmen die ook in de vorige implementatie gebruikt zijn (K-means, agglomerative hierarchical clustering en spectral clustering). De gesegmenteerde dataset heeft nu een dimensie van 2 270 x 198. Oorspronkelijk waren er iets meer dan 2 270 hartslagen, maar omdat de oorspronkelijke dataset een klein aantal hartslagen van klasse "V"bevat en het begin van de ECG-opname enkel een deel van een volledige hartslag heeft, worden deze verwijderd. Het aantal features (198) is het resultaat van de segmentatie stap. In tegenstelling tot de vorige implementatie is de grootte van deze dataset klein genoeg, waardoor deze direct als invoer kan meegegeven worden aan het clustering algoritme.

De tweede aanpak die op de gesegmenteerde dataset toegepast wordt, is het opnieuw trainen van een deep autoencoder om feature-extractie en dimensionality reduction toe te passen. Hoewel dimensionality reduction niet noodzakelijk is, is de feature-extractie functionaliteit van de deep autoencoder steeds waardevol binnen dit onderzoek. Zoals in de vorige implementatie wordt de data eerst genormaliseerd (via de Min-max Scaler) voordat het gebruikt wordt als trainingsdata in de deep autoencoder. Hoewel de downsampling of de sliding window methode noodzakelijk was in de vorige implementatie omdat het aantal features te groot was voor de deep autoencoder, zijn deze technieken in deze implementatie niet vereist. Desondanks wordt de moving average methode toegepast (zonder downsampling) om ruis uit het ECG-signaal te verwijderen. Het resultaat van deze functie is een ECG-signaal waarbij het aantal features licht gereduceerd is. Figuur 3.9 toont hoe het ruwe ECG-signaal uit Figuur 3.8 eruit ziet na toepassing van deze ruisfilter.

Zoals in de vorige implementatie wordt de getrainde encoder opgeslagen en gebruikt om de meest relevante features te extraheren uit de data. Waardoor het aantal features van 216 gereduceerd wordt naar 6. De gereduceerde dataset uit de encoder wordt vervolgens gebruikt als invoer voor de drie clustering algoritmen (K-means, agglomerative hierarchical clustering en spectral clustering).

Hoofdstuk 4

Implementatie en Resultaten

In dit hoofdstuk worden de implementatiedetails en resultaten besproken die zijn verkregen uit de implementaties in Hoofdstuk 3. Het is essentieel om bij het trainen van een NN de hyperparameters optimaal in te stellen. Er bestaat niet één correcte oplossing voor de waarden van de hyperparameters. Deze worden typisch bepaald via *trial and error* samen met bepaalde evaluatietechnieken en richtlijnen.

Het trainingsproces van de deep autoencoder wordt gevisualiseerd met behulp van een *training history plot* in de onderstaande secties. Dit is een grafiek waarin de training loss curve en de *validation* loss curve worden getoond. Training loss is een belangrijke evaluatiemetriek die gebruikt wordt om de fout (loss) te meten tussen de voorspelde waarden van het model en de echte waarden in de trainingsdata. Het concept van validation loss is vergelijkbaar met training loss, maar het verschil is voornamelijk dat de loss gemeten wordt tussen de voorspelde waarden en de echte waarden in de testdata (niet op trainingsdata). Het model traint enkel op trainingsdata en gebruikt testdata om het getrainde model te testen op data die niet het model nog niet "gezien" heeft.
Deze metrieken zijn essentieel voor het evalueren van een getraind model en geven aan of er sprake is van *overfitting* of *underfitting*. Bij overfitting presteert het model goed op de trainingsdata (lage training loss) maar slecht op de testdata (hoge validation loss). Dit resulteert in een model dat de trainingsdata te goed heeft geleerd, waardoor het ook ruis in de trainingsdata heeft opgenomen en daardoor slecht presteert op de testdata.
Underfitting houdt, in tegenstelling tot overfitting, in dat het model te eenvoudig is en onvoldoende getraind is om patronen te identificeren in zowel de trainings- als de testdata. [SL19].

Het doel is om zowel de training loss als de validation loss zo laag mogelijk te houden en daarnaast het verschil tussen beide losses minimaal te houden om zo overfitting en underfitting te voorkomen. In de volgende secties worden deze losses weergegeven in de training history plot om het trainingsproces van de deep autoencoder te visualiseren en te evalueren. Alle implementaties in deze thesis zijn uitgevoerd in Python-versie 3.11.9.

4.1 Deep autoencoder en downsampling

Voor elke subset van de oorspronkelijke PTB-XL AFib dataset die onderzocht werd, wordt het trainingsproces van de deep autoencoder besproken samen met de resultaten verkregen uit de drie clustering algoritmen (K-means, agglomerative hierarchical clustering en spectral clustering). In deze sectie bestaat preprocessing uit een combinatie van een deep autoencoder en downsampling (samen met andere stappen die eerder vermeld zijn, zoals flatten en normalisatie). Verder toont Tabel 4.1 de specifieke waarden van de hyperparameters aan die gebruikt werden voor het trainen van de deep autoencoder. De training history plot wordt enkel voor de volledige

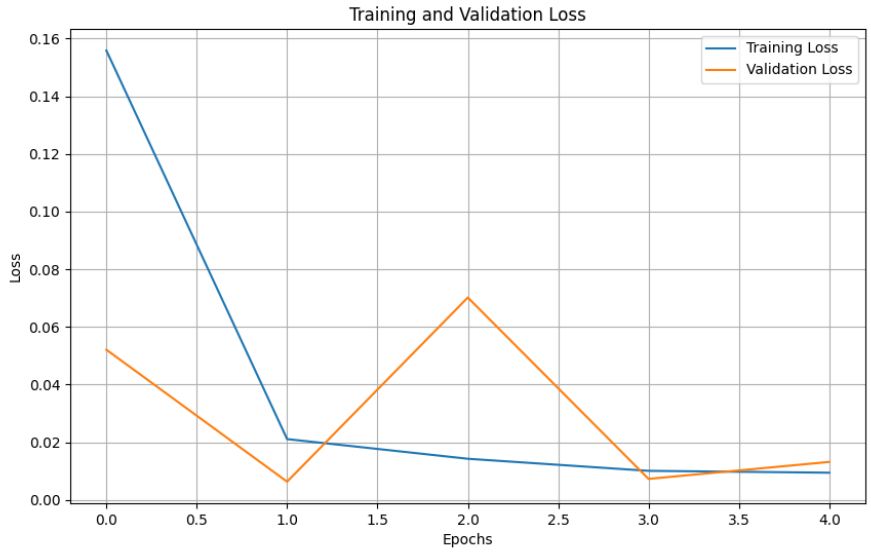
PTB-XL AFib dataset weergegeven, aangezien de trainingsprocessen bij alle subsets gelijkaardig verlopen (lage training en validation loss).

Hyperparameter	**Waarde**
Aantal epochs	100
Batch size	32
Aantal layers (input layer, encoder, bottleneck, decoder en output layer)	7
Aantal knopen per layer (respectievelijk)	Encoder: 10 000, 10 000, 5 000 Bottleneck: 6 Decoder: 5 000, 10 000, 10 000
Optimizer	Adam
Activatie functies	Leaky ReLU en linear (enkel output)
Loss functie	MSE
Learning rate	0,001 (Adam)

Tabel 4.1: Gebruikte hyperparameters bij het trainen van de deep autoencoder voor de downsampling methode.

4.1.1 SR, AF en VA

Hier wordt de oorspronkelijke dataset (PTB-XL AFib) genomen die drie klassen bevat. De grootte van de dataset wordt gereduceerd via downsampling en wordt vervolgens verwerkt door de deep autoencoder. Aan de training history plot in Figuur 4.1 is te zien dat zowel de training loss als de validation loss over het algemeen dalen vanaf epoch 0. Bij epoch 1 valt het op dat de training loss en validation loss laag en ongeveer gelijk zijn. Bij epoch 2 is er een sterke stijging van de validation loss, wat typisch kan leiden tot overfitting. Daarna daalt de validation loss opnieuw, maar het trainingsproces stopt bij epoch 4 door de `Early Stopping` functie van TensorFlow, omdat er geen verbetering is in de validation loss. Doordat de `restore_best_weights` parameter van deze functie op `True` staat, wordt het model dat in eerdere epochs het beste presteerde opgeslagen. In dit geval is dat epoch 1 met een validation loss van 0,0063 (epoch 3 heeft een validation loss van 0,0073).



Figuur 4.1: Training history plot van de deep autoencoder getraind voor de downsampling methode op de volledige PTB-XL AFib dataset.

De tabellen hieronder geven de resultaten weer uit de drie clustering algoritmen, toegepast op de gereduceerde PTB-XL AFib dataset, waarbij de preprocessing bestaat uit downsampling en een deep autoencoder. Deze dataset heeft een dimensie van 6 428 x 6.

K-Means, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	2 091 (32,53%)	640	507	**944**	0,4515 (45,15%)
1	3 972 (61,79%)	1 266	1 007	**1 699**	0,4277 (42,77%)
2	365 (5,68%)	94	73	**198**	0,5425 (54,25%)
Silhouette score: 0,28 **Purity score:** 0,442 (44,2%)					

Tabel 4.2: Contingency table van het resultaat uit **K-Means** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

Agglomerative hierarchical clustering, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	500 (7,78%)	137	100	**263**	0,526 (52,6%)
1	5 085 (79,11%)	1 639	1 274	**2 172**	0,4271 (42,71%)
2	843 (13,11%)	224	213	**406**	0,4816 (48,16%)
Silhouette score: 0,39 **Purity score:** 0,442 (44,2%)					

Tabel 4.3: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

Spectral clustering, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	3 126 (48,63%)	1 013	817	**1 296**	0,4146 (41,46%)
1	1 271 (19,77%)	390	259	**622**	0,4894 (48,94%)
2	2 031 (31,6%)	597	511	**923**	0,4545 (45,45%)
Silhouette score: 0,25 **Purity score:** 0.442 (44,2%)					

Tabel 4.4: Contingency table van het resultaat uit **spectral clustering** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

Affinity propagation vindt, zoals beschreven in Sectie 2.3.3, zelf het optimale aantal clusters. Het resultaat wordt hier niet weergegeven, aangezien dit algoritme 203 clusters teruggaf met een Silhouette score van 0,16. Voor elke cluster werd de distributie van echte labels binnen de cluster onderzocht, maar er was geen cluster met een significant hoge purity score.

4.1.2 SR en AF

Hier geven de onderstaande tabellen opnieuw de resultaten uit de clustering algoritmen weer, maar voor de PTB-XL AFib dataset zonder de samples van klasse VA. Deze subset van de oor-

spronkelijke dataset heeft een dimensie van 3 587 x 60 000. Na preprocessing (downsampling en deep autoencoder) heeft het een nieuwe dimensie van 3 587 x 6. Er wordt wegens domeinkennis gekozen voor $K = 2$.

K-Means, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	3 549 (98,94%)	**1 982**	1 567	0,5585 (55,85%)
1	38 (1,06%)	18	**20**	0,5263 (52,63%)
Silhouette score: 0,89				
Purity score: 0,5581 (55,81%)				

Tabel 4.5: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

Agglomerative hierarchical clustering, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	62 (1,72%)	29	**33**	0,5323 (53,23%)
1	3 525 (98,27%)	**1 971**	1 554	0,5591 (55,91%)
Silhouette score: 0,87				
Purity score: 0,5587 (55,87%)				

Tabel 4.6: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **2 clusters** op de PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

Spectral clustering, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	3 366 (93,84%)	**1 889**	1 477	0,5612 (56,12%)
1	221 (6,16%)	**111**	110	0,5023 (50,23%)
Silhouette score: 0,74				
Purity score: 0,5576 (55,76%)				

Tabel 4.7: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

4.1.3 Evenwichtige dataset: SR, AF en VA

De evenwichtige dataset heeft een dimensie van 4 761 x 60 000. In deze sectie worden terug alle drie (SR, AF en VA) klassen beschouwd. Na preprocessing (downsampling en deep autoencoder) heeft de dataset een nieuwe dimensie van 4 761 x 6.

K-Means, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	4 615 (96,93%)	**1 551**	1 545	1 519	0,3361 (33,61%)
1	27 (0,56%)	6	9	**12**	0,4444 (44,44%)
2	119 (2,5%)	30	33	**56**	0,4706 (47,06)

Silhouette score: 0,78
Purity score: 0,3401 (34,01%)

Tabel 4.8: Contingency table van het resultaat uit **K-Means** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

Agglomerative hierarchical clustering, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	72 (1,51%)	19	18	**35**	0,4861 (48,61%)
1	828 (17,39%)	272	**285**	271	0,3442 (34,42%)
2	3 861 (81,1%)	**1 296**	1 284	1 281	0,3357 (33,57%)

Silhouette score: 0,51
Purity score: 0,3394 (33,94%)

Tabel 4.9: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

Spectral clustering, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	3 495 (73,41%)	**1 179**	1 155	1 161	0,3373 (33,73%)
1	1 172 (24,62%)	382	**407**	383	0,3473 (34,73%)
2	94 (1,97%)	26	25	**43**	0,4574 (45,74%)

Silhouette score: 0,45
Purity score: 0,3422 (34,22%)

Tabel 4.10: Contingency table van het resultaat uit **spectral clustering** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**downsampling** methode).

4.1.4 Evenwichtige dataset: SR en AF

In deze sectie wordt de subset genomen van de evenwichtige PTB-XL AFib dataset, waarbij enkel de samples van klasse VA verwijderd worden. Deze nieuwe dataset is als gevolg een evenwichtige dataset met een dimensie van 3 174 x 60 000. Na preprocessing (downsampling en deep autoencoder) heeft het een nieuwe dimensie van 3 174 x 6.

K-Means, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	2 258 (71,14%)	**1 145**	1 113	0,5071 (50,71%)
1	916 (28,86%)	442	**474**	0,5175 (51,75%)

Silhouette score: 0,4
Purity score: 0,5101 (51,01%)

Tabel 4.11: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

Agglomerative hierarchical clustering, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	2 863 (90,2%)	**1 451**	1 412	0,5068 (50,68%)
1	311 (9,8%)	136	**175**	0,5627 (56,27%)

Silhouette score: 0,55
Purity score: 0,5123 (51,23%)

Tabel 4.12: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

Spectral clustering, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	1 119 (35,26%)	544	**575**	0,5139 (51,39%)
1	2 055 (64,74%)	**1 043**	1 012	0,5075 (50,75%)

Silhouette score: 0,38
Purity score: 0,5098 (50,98%)

Tabel 4.13: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**downsampling** methode).

4.2 Deep autoencoder en sliding window

In deze sectie bestaat preprocessing uit een combinatie van deep autoencoder en de sliding window methode (samen met andere stappen die eerder vermeld zijn zoals flatten en normalisatie). Tabel 4.14 geeft opnieuw de gebruikte hyperparameter waarden weer voor het trainen van de deep autoencoder in deze combinatie.

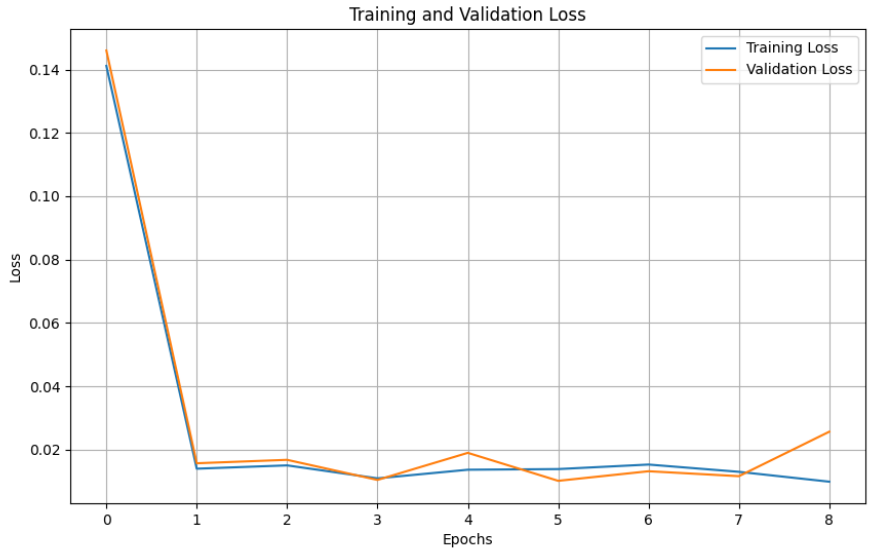
Hyperparameter	Waarde
Aantal epochs	100
Batch size	256
Aantal layers (input layer, encoder, bottleneck, decoder en output layer)	7
Aantal knopen per layer (respectievelijk)	Encoder: 10 000, 10 000, 5 000 Bottleneck: 6 Decoder: 5 000, 10 000, 10 000
Optimizer	Adam
Activatie functies	Leaky ReLU en linear (enkel output)
Loss functie	MSE
Learning rate	0,001 (Adam)

Tabel 4.14: Gebruikte hyperparameters bij het trainen van de deep autoencoder voor de sliding window methode.

De subsets van de oorspronkelijke PTB-XL AFib dataset die voor deze combinatie van preprocessing stappen gebruikt worden, komen overeen met de subsets uit de vorige sectie, daarom worden deze niet opnieuw vermeld in Sectie 4.2.1, 4.2.2, 4.2.3 en 4.2.4. Zoals bij de downsampling methode wordt hier enkel de training history plot van de volledige PTB-XL AFib dataset weergegeven, aangezien de trainingsprocessen bij alle subsets gelijkaardig verlopen (lage training en validation loss).

4.2.1 SR, AF en VA

Figuur 4.2 geeft opnieuw het trainingsproces van de deep autoencoder weer. Door de `Early Stopping` functie stopt het proces bij epoch 8. Echter staat de `restore_best_weights` parameter op `True`, waardoor het model bij epoch 5 opgeslagen wordt, aangezien dit model het beste presteerde met een validation loss van 0,0101.



Figuur 4.2: Training history plot van de deep autoencoder getraind voor de moving window methode op de volledige PTB-XL AFib dataset.

K-Means, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	90 (1,40%)	11	37	**42**	0,4667 (46,67%)
1	6 150 (95,68%)	1 947	1 489	**2 714**	0,4413 (44,13%)
2	188 (2,92%)	42	61	**85**	0,4521 (45,21%)

Silhouette score: 0,61
Purity score: 0,442 (44,2%)

Tabel 4.15: Contingency table van het resultaat uit **K-Means** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

Agglomerative hierarchical clustering, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	6 323 (98,37%)	1 982	1 545	**2 796**	0,4422 (44,22%)
1	71 (1,10%)	15	27	**29**	0,4085 (40,85%)
2	34 (0,52%)	3	15	**16**	0,4706 (47,06%)

Silhouette score: 0,69
Purity score: 0,442 (44,2%)

Tabel 4.16: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

Spectral clustering, K = 3 en PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	6 294 (97,92%)	1 979	1 532	**2 783**	0,4421 (44,21%)
1	65 (1,01%)	7	**30**	28	0,4615 (46,15%)
2	69 (1,07%)	14	25	**30**	0,4347 (43,47%)

Silhouette score: 0,66
Purity score: 0,4423 (44,23%)

Tabel 4.17: Contingency table van het resultaat uit **spectral clustering** met **3 clusters** op de PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

4.2.2 SR en AF

K-Means, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	3 368 (93,89%)	**1 887**	1 481	0,5603 (56,03%)
1	219 (6,11%)	**113**	106	0,516 (51,6%)

Silhouette score: 0,5
Purity score: 0,5576 (55,76%)

Tabel 4.18: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de PTB-XL AFib dataset **zonder VA** (**sliding window** methode).

Agglomerative hierarchical clustering, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	3 534 (98,52%)	**1 976**	1 558	0,5591 (55,91%)
1	53 (1,48%)	24	**29**	0,5472 (54,72%)
Silhouette score: 0,71				
Purity score: 0,559 (55,9%)				

Tabel 4.19: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **2 clusters** op de PTB-XL AFib **zonder VA** (**sliding window** methode).

Spectral clustering, K = 2 en PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	616 (17,17%)	**333**	283	0,5406 (54,06%)
1	2 971 (82,83%)	**1 667**	1 304	0,5611 (56,11%)
Silhouette score: 0,33				
Purity score: 0,5576 (55,76%)				

Tabel 4.20: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de PTB-XL AFib **zonder VA** (**sliding window** methode).

4.2.3 Evenwichtige dataset: SR, AF en VA

K-Means, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	4 377 (0,92%)	**1 479**	1 456	1 442	0,3379 (33,79%)
1	377 (7,92%)	108	125	**144**	0,382 (38,2%)
2	7 (0,15%)	0	**6**	1	0,8571 (85,71%)
Silhouette score: 0,5					
Purity score: 0,3422 (34,22%)					

Tabel 4.21: Contingency table van het resultaat uit **K-Means** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

Agglomerative hierarchical clustering, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	268 (5,63%)	79	91	**98**	0,3657 (36,57%)
1	377 (7,92%)	112	**157**	108	0,4164 (41,64%)
2	4 116 (86,45%)	**1 396**	1 339	1 381	0,3392 (33,92%)
Silhouette score: 0,42					
Purity score: 0,3468 (34,68%)					

Tabel 4.22: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

Spectral clustering, K = 3 en evenwichtige PTB-XL AFib

Cluster nr.	Totaal aantal datapunten	SR	AF	VA	Purity
0	3 676 (77,21%)	1 232	1 191	**1 253**	0,3409 (34,09%)
1	59 (1,24%)	12	**26**	21	0,4407 (44,07%)
2	1 026 (21,55%)	343	**370**	313	0,3606 (36,06%)

Silhouette score: 0,25
Purity score: 0,3464 (34,64%)

Tabel 4.23: Contingency table van het resultaat uit **spectral clustering** met **3 clusters** op de **evenwichtige** PTB-XL AFib dataset met alle 3 klassen (**sliding window** methode).

4.2.4 Evenwichtige dataset: SR en AF

K-Means, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	401 (12,63%)	**209**	192	0,5212 (52,12%)
1	2 773 (87,37%)	1 378	**1 395**	0,5031 (50,31%)

Silhouette score: 0,4
Purity score: 0,5054 (50,54%)

Tabel 4.24: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**sliding window** methode).

Agglomerative hierarchical clustering, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	2 735 (86,17%)	1 348	**1 387**	0,5071 (50,71%)
1	439 (13,83%)	**239**	200	0,5444 (54,44%)

Silhouette score: 0,36
Purity score: 0,5123 (51,23%)

Tabel 4.25: Contingency table van het resultaat uit **agglomerative hierarchical clustering** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**sliding window** methode).

Spectral clustering, K = 2 en evenwichtige PTB-XL AFib zonder VA

Cluster nr.	Totaal aantal datapunten	SR	AF	Purity
0	412 (12,98%)	**222**	190	0,5388 (53,88%)
1	2 762 (87,02%)	1 365	**1 397**	0,5058 (50,58%)

Silhouette score: 0,37
Purity score: 0,5101 (51,01%)

Tabel 4.26: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de **evenwichtige** PTB-XL AFib dataset **zonder VA** (**sliding window** methode).

4.3 Extra implementatie

Hier worden de resultaten van dezelfde clustering algoritmen weergegeven, die eerst zijn toegepast op de ruwe dataset zonder preprocessing en daarna op dezelfde dataset die is verwerkt met verschillende preprocessing stappen, zoals moving average en een deep autoencoder. De kolommen in de tabellen met de labels "N" en "A" geven respectievelijk het symbool weer voor een normale hartslag en een atrial premature beat en geven aan hoeveel datapunten van deze klassen in iedere cluster zitten. Voor deze implementatie wordt de MIT-BIH dataset gebruikt. De dimensie van deze dataset (label 100) is 2 270 x 198.

4.3.1 Naïeve methode

De resultaten in deze sectie zijn afkomstig van clustering algoritmen die zijn toegepast op de ruwe ECG-opname met label 100 van een patiënt uit de MIT-BIH dataset. Voor preprocessing werd enkel normalisatie toegepast.

K-Means, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	1 274 (56,12%)	**1 254**	20	0,9843 (98,43%)
1	996 (43,88%)	**983**	13	0,9869 (98,69%)

Silhouette score: 0,29
Purity score: 0,9855 (98,55%)

Tabel 4.27: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de MIT-BIH dataset zonder toegepaste preprocessing stappen op de data.

Agglomerative hierarchial clustering, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	1 528 (67,31%)	**1 500**	28	0,9817 (98,17%)
1	742 (32,69%)	**737**	5	0,9933 (99,33%)

Silhouette score: 0,28
Purity score: 0,9855 (98,55%)

Tabel 4.28: Contingency table van het resultaat uit **agglomerative hierachical clustering** met **2 clusters** op de MIT-BIH dataset zonder toegepaste preprocessing stappen op de data.

Spectral clustering, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	804 (35,42%)	**795**	9	0,9888 (98,88%)
1	1 466 (64,58%)	**1 442**	24	0,9836 (98,36%)

Silhouette score: 0,29
Purity score: 0,9855 (98,55%)

Tabel 4.29: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de MIT-BIH dataset zonder toegepaste preprocessing stappen op de data.

4.3.2 Smoothing en deep autoencoder

In tegenstelling tot de vorige methode worden in deze sectie de resultaten van de clustering algoritmen weergegeven waarbij preprocessing wel is toegepast, namelijk de moving average techniek voor smoothing en een deep autoencoder voor feature-extractie. De uiteindelijke dimensie van deze dataset is 2 270 x 6.

K-Means, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	1 750 (77,09%)	**1 722**	28	0,9840 (98,40%)
1	520 (22,91%)	**515**	5	0,9904 (99,04%)

Silhouette score: 0,39
Purity score: 0,9855 (98,55%)

Tabel 4.30: Contingency table van het resultaat uit **K-Means** met **2 clusters** op de MIT-BIH dataset waar smoothing en deep autoencoder op toegepast is.

Agglomerative hierarchial clustering, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	1 636 (72,07%)	**1 609**	27	0,9835 (98,35%)
1	634 (27,93%)	**628**	6	0,9905 (99,05%)

Silhouette score: 0,48
Purity score: 0,9855 (98,55%)

Tabel 4.31: Contingency table van het resultaat uit **agglomerative hierachical clustering** met **2 clusters** op de MIT-BIH dataset waar smoothing en deep autoencoder op toegepast is.

Spectral clustering, K = 2 en de MIT-BIH dataset

Cluster nr.	Totaal aantal datapunten	N	A	Purity
0	1 772 (78,06%)	**1 743**	29	0,9836 (98,36%)
1	498 (21,94%)	**494**	4	0,992 (99,2%)

Silhouette score: 0,51
Purity score: 0,9855 (98,55%)

Tabel 4.32: Contingency table van het resultaat uit **spectral clustering** met **2 clusters** op de MIT-BIH dataset waar smoothing en deep autoencoder op toegepast is.

Hoofdstuk 5

Discussie

Het trainingsproces van de deep autoencoders van zowel de downsampling als de sliding window methode is succesvol verlopen. Dit blijkt uit de lage training loss en validation loss van het trainingsproces van beide methodes. De grafieken van Figuren 4.1 en 4.2 tonen dit respectievelijk aan. Het verschil in de prestaties van het trainingsproces van de twee methodes zit in de tijdsduur van dit proces. De downsampling methode vergt minder tijd dan de sliding window methode door de hoeveelheid data die de sliding window methode als invoer gebruikt. Om deze reden werd de batch size vergroot van de sliding window methode om het trainingsproces te versnellen. Zoals getoond in Figuur 4.2 heeft dit geen invloed op de performantie van de deep autoencoder aangezien zowel de training loss als de validation loss laag blijven op epoch 5 ($<$ 0,02).

Daarnaast valt het op dat er 100 epochs voorzien waren voor het trainingsproces, maar in alle gevallen stopt het proces al voor 10 epochs door de `Early Stopping` functie. Om de centrale onderzoeksvraag van deze thesis te beantwoorden, is het doel van de methodologie over het algemeen om betekenisvolle resultaten te verkrijgen. Dit betekent dat de verkregen clusters uit de clustering algoritmen sterk genoeg een bepaalde klasse (een bepaalde aritmie) moeten representeren. Daarom worden in de opeenvolgende secties de cijfers uit de evaluatietechnieken besproken samen met hun exacte betekenis.

5.1 Resultaten op de volledige PTB-XL AFib dataset

In Sectie 4.1.1 en 4.2.1 worden de finale resultaten weergegeven van de combinatie van de twee methodes (downsampling en sliding window) en drie verschillende clustering algoritmen op de volledige PTB-XL AFib dataset. Zoals eerder vermeld, heeft deze dataset een dimensie van 6 428 x 60 000 waarbij 2 000 samples behoren tot klasse SR, 1 587 tot klasse AF en 2 841 tot klasse VA. Dit is een redelijk ongebalanceerde dataset, maar de verwachting is dat de clustering algoritmen correcte clusters vormen die deze drie klassen representeren.

5.1.1 Verdeling en sterkte van de clustering

Tabel 4.2 geeft het resultaat weer van het K-means algoritme met $K = 3$, toegepast op de dataset waarop de downsampling methode is toegepast.
Het valt op dat cluster 1 de dominantste cluster is met 3 972 datapunten. Deze cluster bevat 61,79% van alle datapunten, terwijl cluster 2 slechts 365 datapunten bevat, wat overeenkomt met 5,68% van het totaal aantal datapunten.
Cluster 0 bevat echter een groot aantal datapunten, namelijk 2 091 (32,53%).
Perfecte clustering betekent dat de clusters respectievelijk 1 587, 2 000 en 2 841 datapunten hebben die elk één klasse representeren. Verder is de Silhouette score van de clustering 0,28. Dit is een lage score en houdt in dat het clustering algoritme niet “zeker” is van de gecreëerde

clusters. Deze score samen met de onevenwichtige distributie van de datapunten over de clusters indiceert over het algemeen een slechte clustering. Echter geeft agglomerative hierarchical clustering in Tabel 4.3 opnieuw een slechte distributie van datapunten over de clusters terug. Zoals eerder vermeld, is een Silhouette score die 1 benadert een indicatie van sterke clustering. Hoewel de Silhouette score van 0,39 al een verbetering is ten opzichte van het resultaat van K-means, is dit steeds een Silhouette lage score. De combinatie van deze score en de distributie van de datapunten indiceert echter opnieuw een slechte clustering. In tegenstelling tot de resultaten van de vorige twee clustering algoritmen, lijkt het resultaat in Tabel 4.4 (waarin spectral clustering toegepast werd) een betere distributie van datapunten te tonen. De kleinste cluster (cluster 1) bevat 1 271 datapunten, wat bijna 20% is van het totaal aantal datapunten, gevolgd door cluster 2 met 2 031 datapunten en cluster 0 met 3 126 datapunten. Echter geeft de Silhouette score van 0,25 aan dat de clustering nog steeds zwak is.

Dezelfde vergelijking wordt gemaakt voor de sliding window methode in Sectie 4.2.1. Tabel 4.15, 4.16 en 4.17 geven opnieuw respectievelijk de resultaten van K-means, agglomerative hierarchical en spectral clustering weer, toegepast op de oorspronkelijke PTB-XL AFib dataset waarop de sliding window methode toegepast is. Initieel valt het op dat de Silhouette scores significant hoger zijn ten opzichte van de vorige resultaten op de gedownsamplede dataset. Deze scores zijn respectievelijk 0,61, 0,69 en 0,66, wat verwijst naar een matige tot goede clustering. Echter blijkt de distributie van datapunten over de clusters opvallend slechter te zijn vergeleken met de vorige resultaten waarbij er in alle drie gevallen één dominante cluster is met meer dan 95% van het totaal aantal datapunten. De overige twee clusters bevatten een zeer klein aantal datapunten. Hier lijkt de dominante cluster de resterende datapunten te representeren, waarin geen duidelijke patronen zijn gevonden.

5.1.2 Purity

Aangezien de vorige sectie enkel een overzicht gaf van de sterkte van de clustering en de distributie van de datapunten, wordt ook de purity score gemeten om te bepalen of iedere cluster effectief een specifieke klasse voldoende representeert. Bij de downsampling methode is klasse VA voor iedere cluster de dominantste klasse bij alle clustering algoritmen. Echter kan er voor geen enkele cluster geconcludeerd worden dat deze klasse VA representeert, aangezien de purity scores relatief laag blijven. De hoogste purity score wordt behaald door cluster 2 van het K-means algoritme met een score van 0,5425 (54,25%), wat net iets meer dan de helft van het totaal aantal datapunten is. De combinatie van de lage cluster purity scores, lage totale purity score, het feit dat VA de dominantste klasse is in de oorspronkelijke dataset en dat elke cluster klasse VA representeert, indiceert dat de clustering algoritmen geen duidelijke patronen hebben gevonden die aritmieën in de ECG-opnames onderscheiden van elkaar.

Bij de sliding window methode kunnen er gelijkaardige observaties gemaakt worden. Het opvallendste verschil is dat de cluster purity scores van alle clustering algoritmen (bij de sliding window methode) onder 0,48 blijven. In de resultaten van spectral clustering in Tabel 4.17 valt het vervolgens op dat klasse AF de dominantste klasse is in cluster 1, echter opnieuw met een lage purity score van 0,4615 (46,15%), dicht gevolgd door klasse VA met een purity score van 0,4308 (28/65). Hierdoor kan er niet geconcludeerd worden dat cluster 1 de klasse AF representeert.

De gemiddelde purity score is voor alle clustering algoritmen bij zowel de downsampling als de sliding window methode gelijk aan 0,442 met uitzondering van de combinatie van spectral clustering en de sliding window methode. Deze laatste is de enige combinatie, waarbij een cluster bestaat met een dominantste klasse die niet VA is. Over het algemeen kan er voor alle combinaties van clustering algoritmen en beide preprocessing methoden op de volledige oorspronkelijke PTB-XL AFib dataset worden geconcludeerd dat er geen betekenisvolle resultaten zijn behaald.

5.2 Resultaten op de PTB-XL AFib dataset zonder VA

In deze sectie worden de resultaten uit Sectie 4.1.2 en 4.2.2 besproken, waarbij opnieuw dezelfde drie clustering algoritmen toegepast zijn op een subset van de oorspronkelijke PTB-XL AFib dataset.
Deze subset bestaat uit alle datapunten uit de volledige dataset met uitzondering van de datapunten van klasse VA. Deze subset bevat 1 587 datapunten van klasse AF en 2 000 datapunten van klasse SR (in totaal 3 587 datapunten).
Aangezien deze subset twee klassen bevat, werd $K = 2$ gekozen voor alle clustering algoritmen die toegepast zijn op deze subset.

5.2.1 Verdeling en sterkte van de clustering

In dit onderzoek valt het initieel op dat de Silhouette score over het algemeen hoog is. Bij de downsampling methode geven Tabel 4.5, 4.6 en 4.7 respectievelijk een gemiddelde Silhouette score van 0,89, 0,87 en 0,74 terug. Deze hoge Silhouette scores geven aan dat de toegepaste clustering algoritmes "zeker" zijn van de clustering. Echter blijft de distributie van de datapunten over de clusters een probleem vormen, waarbij cluster 1 van het K-means algoritme slechts 38 datapunten, cluster 0 van agglomerative hierarchical clustering slechts 62 en cluster 1 van spectral clustering slechts 221 datapunten bevat. Terwijl goede clustering in deze context een cluster moet aanmaken die klasse SR representeert met een benadering van 2 000 datapunten en een andere cluster die klasse AF representeert met een benadering van 1 587 datapunten. Hoewel elke cluster in de vorige aanpak (op de volledige oorspronkelijke dataset) dezelfde klasse blijkt te representeren (klasse VA), lijkt elke cluster in deze aanpak bij K-means en agglomerative hierarchical clustering een unieke klasse te representeren. Op dit aspect lijkt de clustering toch beter geslaagd te zijn. Spectral clustering daarentegen geeft in deze aanpak wel twee clusters terug die allebei dezelfde klasse (SR) representeren.

De resultaten van de sliding window methode in Tabel 4.18, 4.19 en 4.20 geven opnieuw Silhouette scores weer die over het algemeen verhoogd zijn vergeleken met de combinatie van de sliding window methode en de volledige oorspronkelijke dataset. Deze scores zijn echter over het algemeen lager dan de Silhouette scores van de vorige resultaten (downsampling en PTB-XL AFib zonder VA). Vervolgens is de distributie van de datapunten over de clusters opnieuw suboptimaal om dezelfde redenen als eerder vermeld, aangezien klasse SR de dominante klasse is voor alle clusters van K-means en spectral clustering. In tegenstelling tot deze twee clustering algoritmen geeft agglomerative hierarchical clustering twee clusters terug die elk een verschillende klasse representeren. Echter hebben deze opnieuw weinig betekenis, aangezien cluster 1 (die klasse AF representeert) enkel 53 datapunten in totaal bevat.

5.2.2 Purity

Opvallend is dat de gemiddelde purity score significant is verhoogd, ten opzichte van de vorige resultaten op de volledige oorspronkelijke dataset, voor zowel de downsampling als de sliding window methode. De gemiddelde purity scores van ongeveer 55% geeft aan dat de clusters beter een bepaalde klasse representeren. Echter verwijst de combinatie van de distributie van de datapunten en de steeds lage purity scores naar een slechte of zwakke clustering.

Een voorbeeld hiervan is de combinatie agglomerative hierarchical clustering en de sliding window methode op deze dataset. De resultaten geven een cluster terug van 53 datapunten, waarbij de purity score bijna 55% is (Tabel 4.19). Echter bestaat deze cluster uit 24 datapunten van klasse SR en 29 datapunten van klasse AF. De datapunten van klasse AF zijn dominanter, maar het verschil zit enkel in de vijf datapunten. Door dit kleine verschil in combinatie met een kleine sample size van 53 datapunten, kan er niet geconcludeerd worden dat de cluster klasse AF volledig representeert. Daarnaast zijn de hoogste cluster purity scores, zoals cluster 0 in Tabel 4.7 en cluster 1 in Tabel 4.20 vaak afkomstig uit clusters die de "overige" datapunten bevatten. Dit kan mogelijks verklaard worden door de onevenwichtige dataset die hier gebruikt

wordt, waarbij 2 000 datapunten van klasse SR zijn. In alle gevallen hebben de kleinere clusters onvoldoende aantal datapunten en een onvoldoende hoge purity score voor de klasse AF. Hierdoor bevat de grotere cluster een veel grotere hoeveelheid datapunten van klasse SR dan van klasse AF.

5.3 Resultaten op de evenwichtige PTB-XL AFib dataset

Eerdere resultaten tonen aan dat een onevenwichtige dataset het moeilijker maakt om bepaalde inzichten te interpreteren. Daarom ligt de nadruk in deze sectie op de evenwichtige dataset, waarbij de oorspronkelijke dataset gereduceerd is tot een subset van 4 761 datapunten. Deze subset bestaat steeds uit drie klassen (SR, AF en VA). Iedere klasse wordt gerepresenteerd door 1 587 datapunten.

5.3.1 Verdeling en sterkte van de clustering

Tabel 4.8, 4.9 en 4.10 geven de resultaten weer van de drie clustering algoritmen (met $K = 3$) toegepast op de evenwichtige dataset waar de downsampling methode op is toegepast.
Op dezelfde manier geven Tabel 4.21, 4.22 en 4.23 de resultaten weer van de clustering algoritmen (met $K = 3$) op dezelfde dataset waar de sliding window methode op is toegepast.

Opvallend aan deze resultaten is dat beide methodes (downsampling en sliding window), clusters teruggeven die elk één unieke klasse representeren met twee uitzonderingen: de combinatie van K-means met de downsampling methode (Tabel 4.8) en de combinatie van spectral clustering met de sliding window methode (Tabel 4.23).
De Silhouette scores zijn echter variërend bij alle combinaties. Het K-means algoritme met de downsampling methode geeft de hoogste Silhouette score van 0,78 terug, terwijl spectral clustering met sliding window methode de laagste Silhouette score teruggeeft van 0,25.

Voor de gebruikte dataset is de perfecte distributie van datapunten over de clusters dat elke cluster exact 1 587 datapunten bevat, aangezien het een evenwichtige dataset is. Echter is er geen enkele combinatie die deze perfecte distributie van datapunten benadert. Hoewel er combinaties bestaan, waarbij twee van de drie clusters grotere hoeveelheden aan datapunten bevatten, bestaat er altijd een laatste cluster die enkel tientallen datapunten bevat. Bij de combinatie van K-means en de sliding window methode bestaat cluster 2 zelfs uit slechts 7 datapunten.

5.3.2 Purity

Purity scores blijven laag voor beide methodes (downsampling en sliding window). Voor de downsampling methode ligt de gemiddelde purity score rond de 34%, terwijl deze voor de moving window methode rond de 35% is.
In Tabel 4.24 waar K-means gecombineerd wordt met sliding window is er sprake van een 85,71% cluster purity score bij cluster 2. Dit wijst er initieel op dat cluster 2 klasse AF representeert. Echter bevat deze cluster in totaal slechts 7 datapunten, wat een te kleine sample size is om te concluderen dat de gebruikte combinatie effectief de patronen van klasse AF herkent. Bovendien blijven de cluster purity scores voor andere combinaties onder 50%. Hierdoor kan er opnieuw geconcludeerd worden dat de gebruikte combinaties op de evenwichtige dataset geen gewenste resultaten opleveren. Zoals eerder aangehaald, is het toch opvallend dat ten opzichte van de resultaten van de combinaties op de onevenwichtige dataset, dat bijna iedere cluster die gemaakt wordt één unieke klasse representeert.

5.4 Resultaten op de evenwichtige PTB-XL AFib dataset zonder VA

In deze sectie wordt een subset genomen van de evenwichtige dataset die in de vorige implementatie gebruikt werd. Deze subset is opnieuw een evenwichtige dataset, waarbij de datapunten van klasse VA verwijderd zijn. Als gevolg bevat deze dataset enkel 3 174 datapunten waarvan de helft van klasse SR en de andere helft van klasse AF zijn.
De resultaten in Tabel 4.11, 4.12 en 4.13 zijn afkomstig van de combinaties van de clustering algoritmen (met $K = 2$) en de downsampling methode. De resultaten in Tabel 4.24, 4.25 en 4.26 zijn vervolgens afkomstig van dezelfde clustering algoritmen (met $K = 2$) in combinatie met de sliding window methode.

5.4.1 Verdeling en sterkte van de clustering

Aangezien de gebruikte dataset opnieuw een evenwichtige dataset is, wordt er gehoopt op een perfecte clustering, waarbij twee clusters elk 1 587 datapunten bevatten die respectievelijk de klasse SR en klasse AF representeren. De combinatie van het gebruik van een evenwichtige dataset met binaire clustering, maakt het gemakkelijker om inzicht te krijgen over hoe goed de clustering algoritmen effectief patronen kunnen onderscheiden tussen twee aritmieën (SR en AF). Wanneer er initieel gekeken wordt naar de Silhouette score valt het op dat deze scores bij zowel de downsampling methode als bij de sliding window methode slechts matig zijn.

De beste distributie van datapunten komt uit Tabel 4.13 waarbij spectral clustering gecombineerd wordt met de downsampling methode. Dit resulteert in cluster 0 met 1 119 datapunten en cluster 1 met 2 055 datapunten. Dit betekent een afwijking van ongeveer slechts 15% ten opzichte van een perfecte clustering, waarbij iedere cluster 50% van het totaal aantal datapunten bevat. Andere combinaties van zowel de downsampling als de sliding window methode gaven slechtere distributie van datapunten terug.

5.4.2 Purity

Aangezien hier een evenwichtige dataset gebruikt wordt met twee klassen, is het logisch dat iedere cluster purity score 50% of hoger is. De hoogste individuele purity score komt uit cluster 1 in Tabel 4.12 bij de combinatie van agglomerative hierarchical clustering en downsampling. Deze cluster bevat 311 datapunten en komt overeenkomt met 9,8% van het totaal aantal datapunten samen met een cluster purity score van 56,27%. Vergeleken met de clusters van andere combinaties lijkt dit minder “random” te zijn. Echter zijn deze cijfers relatief laag om te concluderen dat deze cluster effectief de patronen van een ECG van klasse SR en een ECG van klasse AF kan onderscheiden. Aangezien de totale gemiddelde purity scores van iedere combinatie voor deze dataset rond de 50% zitten, is het duidelijk dat de resultaten nog steeds suboptimaal zijn. Bovendien verliest de cluster, die een cluster purity score van 56.27% heeft, over het algemeen zijn betekenis door de grotere cluster (cluster 0) die 90.2% van het totaal aantal datapunten bevat met een cluster purity score van 50.68%. Er kan dus geconcludeerd worden dat de combinaties van deze methodes geen betekenisvolle resultaten opleveren op de gebruikte subset.

5.5 Resultaten extra implementatie

Dezelfde evaluatietechnieken worden gebruikt voor de extra implementatie waarbij de MIT-BIH dataset gebruikt werd. In deze context betekent perfecte clustering dat twee clusters ieder het label “N” (normaal sinusritme) en label “A” (atrial premature beat) representeren. De cluster met label “N” zou in dat geval 2 237 datapunten bevatten, terwijl de cluster met label “A” 33 datapunten bevat.

In de resultaten van de naïeve methode in Sectie 4.3.1 is het duidelijk dat alle drie clustering

algoritmen een lage Silhouette score hebben (0,28 of 0,29). Bovendien zijn de cluster purity scores in alle clusters minstens 98%. Dit wijst op het feit dat iedere cluster zeer sterk gerepresenteerd wordt door een bepaald label. In het perfecte geval zou één cluster het label “N” representeren en de andere het label “A”. Echter bewijzen de resultaten het tegendeel, waarbij iedere cluster het label “N” representeert.
Bij perfecte clustering bestaat de cluster, die het normale sinusritme representeert, uit 98,55% van het totaal aantal datapunten, waarbij het overige deel (1,45%) in de andere cluster zit. Echter voldoet geen enkele cluster aan de gewenste distributie van datapunten. Dit samen met de totale gemiddelde purity score van 98,55% voor alle resultaten, geeft opnieuw aan dat de gebruikte combinatie van implementatie technieken niet het gewenste resultaat oplevert. Dit is echter te verwachten, aangezien er geen preprocessing op de data is toegepast.

Voor de resultaten in Sectie 4.3.2 worden er in tegenstelling tot de naïeve methode wel een aantal preprocessing stappen toegepast, zoals de moving average voor ruisfiltering en een deep autoencoder voor feature-extractie. De resultaten tonen een significante verbetering in de Silhouette score. K-means geeft nog steeds een relatief lage Silhouette score van 0,39 terug, maar agglomerative hierarchical clustering en spectral clustering geven een matige Silhouette score terug van respectievelijk 0,48 en 0,51. Echter wijken de finale resultaten nog steeds sterk af van de gewenste resultaten, waarbij de algemene distributie van datapunten en de purity scores steeds ver verwijderd zijn van een ideale clustering. Net zoals bij de naïeve methode representeert iedere cluster het label “N” met zeer hoge purity scores van meer dan 98%.

Hoofdstuk 6

Conclusies

6.1 Mijn bevindingen

In Hoofdstuk 5 werd het duidelijk dat de gehanteerde methodologie geen gewenste resultaten heeft opgeleverd. De onderzoeksvraag van deze thesis is: **Hoe effectief zijn clustering algoritmen bij het classificeren van verschillende typen hartritmestoornissen op basis van ECG-data?**
Uit het onderzoek blijkt dat het sterk afhankelijk is van alle stappen die voor de laatste clustering stap geïmplementeerd worden. Het is bijvoorbeeld niet mogelijk om een ruwe dataset rechtstreeks als invoer te gebruiken voor een clustering algoritme, zelfs als deze klein genoeg is. Dit werd duidelijk in de implementatiefase, waar individuele hartslagen uit een kleinere dataset rechtstreeks als invoer gebruikt worden in verschillende clustering algoritmen. Ook uit de literatuurstudie blijkt dat men vaak zeer specifieke methodes toepast op de ruwe dataset, die ondersteund worden door complexe wiskundige formules, voordat men overgaat naar de laatste clustering stap.

Hoewel het jammer is dat de resultaten niet optimaal zijn, lag de focus van deze thesis op het onderzoeken van hoe clustering algoritmen presteren op ECG-data. Hierdoor lag de nadruk op de methodologie en het vergelijken van verschillende clustering algoritmen, zodat er een antwoord gevormd kan worden op de onderzoeksvraag. Uit de resultaten lijkt het niet mogelijk te zijn om te concluderen dat clustering algoritme X beter presteert dan algoritme Y, aangezien ieder clustering algoritme anders presteerde afhankelijk van de voorafgaande dataverwerking stappen, zoals ruisfiltering, feature-extractie, dimensionality reduction, gebruikte subsets van datasets en andere preprocessing stappen.

Daarnaast valt het op dat er geen generieke oplossing bestaat die alle aritmieën kan onderscheiden. Iedere studie uit de literatuurstudie die in deze thesis beschouwd werd, neemt een bepaalde subset van de beschikbare dataset om specifieke aritmieën van elkaar te onderscheiden en past specifieke combinaties van preprocessing technieken en clustering algoritmen toe, waardoor iedere combinatie een ander resultaat geeft. Andere mogelijke interessante aanpassingen in de methodologie die door de scope van dit onderzoek niet toegepast werden, zijn:

- Op een slimme manier bepalen welke van de twaalf afleidingen in de PTB-XL AFib dataset de beste resultaten opleveren.
- Verschillende combinaties van afleidingen uitproberen.
- Andere evaluatietechnieken (zoals F-score), maar voor de resultaten in deze thesis lijken de gebruikte evaluatietechnieken voldoende te zijn.
- Grotere datasets
- Performantie gerichte aanpak: optimalisatietechnieken.

Een andere mogelijke toekomstige uitbreiding is om meerdere clustering algoritmen toe te passen. In dit onderzoek werd er bewust gekozen voor dezelfde algoritmen als die in de literatuur worden gehanteerd om de resultaten rechtstreeks te kunnen vergelijken. Echter kan er uit dit onderzoek geconcludeerd worden dat het probleem complexer is dan wat sommige studies suggereren. De uitdaging ligt voornamelijk in het verwerken van ruwe ECG-data (voordat het als invoer gebruikt wordt in de clustering algoritmen) en niet in de clustering zelf.

6.2 Persoonlijke reflectie

Het leerproces begon met het begrijpen van de basisbegrippen binnen het domein van ECG-analyse. Termen zoals "elektrocardiogram" waren mij in het begin van dit leerproces niet bekend, maar door me tijdens de literatuurstudie te verdiepen in de fundamentele begrippen en concepten in dit domein, werd het geleidelijk duidelijk wat het onderwerp inhield. Naast de theoretische betekenis van deze termen, leerde ik ook over de praktische toepassingen van een ECG. Zoals hoe het hartritme precies wordt gemeten van een patiënt met behulp van elektrodes, wat deze elektrodes precies zijn en waarom ze op bepaalde posities geplaatst worden op de patiënt, hoe een cardioloog de metingen uit deze elektrodes kan lezen en betekenis eruit kan halen.

Daarnaast werd het duidelijk waarom unsupervised learning binnen ECG-analyse een belangrijk onderwerp is, welke voordelen unsupervised learning technieken met zich meebrengen en hoe deze ingezet kunnen worden om praktische problemen in dit domein op te lossen. Uit de literatuurstudie blijkt dat er in het algemeen veel onderzoek bestaat rond het toepassen van supervised learning technieken op signaaldata. Meer specifiek werden er ook studies teruggevonden waarin dergelijke technieken toegepast worden op ECG-data om verschillende aritmieën te classificeren en te onderscheiden van elkaar. In tegenstelling tot supervised learning technieken was het moeilijker om in de literatuur toepassingen van unsupervised learning technieken op ECG-data te vinden.

Met de ondersteuning van de promotor was het mogelijk om het onderzoek te starten met een deep autoencoder, wat tijdens de implementatiefase geleid heeft tot verschillende problemen die incrementeel opgelost moesten worden. Deze problemen zijn bekende problemen die voorkomen bij het toepassen van AI technieken in het algemeen, zoals een te grote dataset of onvoldoende kwaliteit van de dataset (die gebruikt wordt om een model te trainen). Hierdoor was het mogelijk om me te verdiepen in diverse aspecten van AI en niet enkel in de specifieke toepassing van ECG-analyse. Hierdoor kreeg ik een breder en concreter inzicht in de mogelijke combinaties die konden worden toegepast binnen de methodologie van dit onderzoek. Door de grote hoeveelheid aan mogelijke combinaties lag de uitdaging niet enkel in de implementatie, maar voornamelijk in het maken van correcte keuzes. Aangezien iedere keuze binnen de methodologie verantwoord moet kunnen worden, worden enkel de keuzes waarvoor een duidelijke motivering gegeven kan worden besproken in de methodologie van deze thesis.

Tijdens de implementatiefase vormden het "debuggen" of het identificeren van de problemen in de implementatie een uitdagend onderdeel. Omdat er verschillende AI technieken toegepast werden, was het uitdagend om de exacte oorzaak van de suboptimale resultaten te identificeren. Bijvoorbeeld een deep autoencoder, die in deze masterproef getraind werd, is een neuraal netwerk met verschillende hyperparameters. In dit geval volstaat het om evaluatietechnieken toe te passen om verschillende combinaties van hyperparameters te testen, zodat de beste combinatie geselecteerd kan worden. Hieruit is het mogelijk om af te leiden dat de deep autoencoder zelf voldoende getraind is, maar het blijft uitdagender om te bepalen of de resultaten van de geëxtraheerde encoder uit de deep autoencoder, de oorzaak zijn van de ongewenste resultaten. Dergelijke uitdagingen komen ook voor bij andere stappen in het proces. De oorzaak kan bijvoorbeeld de downsampling of de sliding window methode zijn, waarbij de ruisfilter functionaliteit toch niet zoals verwacht lijkt te presteren. Daarnaast is het moeilijk om te achterhalen of er toch niet te veel informatie verloren gaat bij het reduceren van het aantal features.
Een andere vraag die gesteld zou kunnen worden is of de feature-extractie functionaliteit van

de deep autoencoder effectief de features extraheert die de belangrijke patronen omvatten die nodig zijn voor het clustering algoritme om de patronen in de ECG-data correct te onderscheiden van elkaar. Door evaluatietechnieken toe te passen op de finale resultaten werd het snel duidelijk dat de gebruikte combinaties in deze thesis geen optimale oplossingen bieden voor het onderzoek.

Tijdens het schrijven van deze thesis werden bepaalde concepten, die in de implementatie toegepast werden, duidelijker. Door de toegepaste technieken verbaal te beschrijven en verder bijkomend opzoekwerk uit te voeren die mijn keuzes motiveren, was het mogelijk om een concreet beeld te vormen van de samenhang van de volledige implementatie. In tegenstelling tot de implementatiefase, waarin voornamelijk verschillende technieken uitgeprobeerd werden om op technologisch vlak een breder beeld te krijgen, was het mogelijk om tijdens het schrijven van de thesis een breder theoretisch beeld te krijgen van de uitgevoerde implementaties. Door tijdens het schrijven opnieuw de literatuur te raadplegen, kreeg ik ook bijkomende ideeën die geïmplementeerd zouden kunnen worden, maar wegens de scope van deze thesis konden ze niet meer opgenomen worden.

Over het algemeen ben ik tevreden met de gevolgde aanpak van deze thesis. Echter zou meer aandacht voor de literatuur en bestaande technieken in het begin van de thesis een verbeterpunt kunnen zijn in de aanpak. Een verbetering zou bijvoorbeeld kunnen zijn om initieel sterk te beginnen aan de literatuurstudie en deze al op te nemen in de thesistekst, zodat er een helderder beeld gevormd kan worden van de bestaande technieken om vervolgens met sterkere motivatie nieuwe combinaties van technieken te onderzoeken.

Bibliografie

[AA20] Ali A. Amer and Hassan I. Abdalla. A set theory based similarity measure for text clustering and classification. 7(1):74, 2020.

[AD08] Mikhled Alfaouri and Khaled Daqrouq. Ecg signal denoising by wavelet transform thresholding. *American Journal of applied sciences*, 5(3):276–281, 2008.

[AJDAS+22] Hussein Abdel-Jaber, Disha Devassy, Azhar Al Salam, Lamya Hidaytallah, and Malak EL-Amir. A review of deep learning algorithms and their applications in healthcare. 15(2):71, 2022.

[AKC13] Jemal H Abawajy, Andrei V Kelarev, and Morshed Chowdhury. Multistage approach for clustering and classification of ecg data. *Computer methods and programs in biomedicine*, 112(3):720–730, 2013.

[ARS+22] Jagadeeswararao Annam, Suja Radha, Jayaprada Somala, Dr Prasad, Narayana Satyala, and Bapi Surampudi. *ECG Feature Extraction*, pages 177–195. 01 2022.

[BGF17] Facundo Bre, Juan Gimenez, and Víctor Fachinotti. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. *Energy and Buildings*, 158, 11 2017.

[Bro20] Jason Brownlee. Autoencoder feature extraction for classification. 2020.

[BVWB08] Manuel Blanco-Velasco, Binwei Weng, and Kenneth E Barner. Ecg signal denoising and baseline wander correction based on the empirical mode decomposition. *Computers in biology and medicine*, 38(1):1–13, 2008.

[Cle20] Cleveland Clinic. What's an EKG?, 2020.

[CLSF13] Filipe Canento, André Lourenço, Hugo Silva, and Ana Fred. Review and comparison of real time electrocardiogram segmentation algorithms for biometric applications. In *Proc. 6th Int. Conf. Health Inform.(HEALTHINF)*, 2013.

[dACC23] Lucas BV de Amorim, George DC Cavalcanti, and Rafael MO Cruz. The choice of scaling technique matters for classification performance. *Applied Soft Computing*, 133:109924, 2023.

[Dat25] DataScienceBase. Soft clustering vs. hard clustering | DataScienceBase, 2025.

[Den14] Li Deng. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA transactions on Signal and Information Processing*, 3:e2, 2014.

[DFH19] Duy-Tai Dinh, Tsutomu Fujinami, and Van-Nam Huynh. Estimating the optimal number of clusters in categorical data clustering by silhouette coefficient. In Jian Chen, Van Nam Huynh, Gia-Nhu Nguyen, and Xijin Tang, editors, *Knowledge and Systems Sciences*, pages 1–17, Singapore, 2019. Springer Singapore.

[dow24] What is downsampling? | IBM, June 2024.

[DRMP21] Atul Dwivedi, Himanshuram Ranjan, Advaith Menon, and Prakasam Periasamy. Noise reduction in ecg signal using combined ensemble empirical mode decomposition method with stationary wavelet transform. *Circuits, Systems, and Signal Processing*, 40, 02 2021.

[FCG08] F. H Fenton, E. M. Cherry, and L. Glass. Cardiac arrhythmia. *Scholarpedia*, 3(7):1665, 2008. revision #121399.

[FD07] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *science*, 315(5814):972–976, 2007.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[GGM+23] Cédric Gilon, Jean-Marie Grégoire, Marianne Mathieu, Stephane Carlier, and Hugues Bersini. Iridia-af, a large paroxysmal atrial fibrillation long-term electrocardiogram monitoring database. *Scientific Data*, 10, 10 2023.

[GIMC21] Guadalupe García-Isla, Luca Mainardi, and Valentina DA Corino. A detector for premature atrial and ventricular complexes. *Frontiers in Physiology*, 12:678558, 2021.

[Goo25] Google. Classification: Accuracy, recall, precision, and related metrics | machine learning, 2025.

[Har] Hartcentrum Gent. Elektrocardiogram (ECG) | hartcentrum gent. z.d. (zonder datum).

[HY] Joseph Heaton and Srikanth Yandrapalli. Premature atrial contractions. In *StatPearls*. StatPearls Publishing.

[IBM21a] IBM. What is a neural network? | IBM, 2021.

[IBM21b] IBM. What is unsupervised learning? IBM, 2021.

[IBM24] IBM. What is supervised learning? IBM, 2024.

[IMS+24] Erica Iammarino, Ilaria Marcantoni, Agnese Sbrollini, Micaela Morettini, and Laura Burattini. Normalization of electrocardiogram-derived cardiac risk indices: A scoping review of the open-access literature. *Applied Sciences*, 14(20), 2024.

[JTJT20] Manuel Jimenez, Mercedes Torres, Robert John, and Isaac Triguero. Galaxy image classification based on citizen science data: A comparative study. *IEEE Access*, PP:1–1, 03 2020.

[KFS18] Mohammad Kachuee, Shayan Fazeli, and Majid Sarrafzadeh. Ecg heartbeat classification: A deep transferable representation. In *2018 IEEE international conference on healthcare informatics (ICHI)*, pages 443–444. IEEE, 2018.

[Kri21] Sri Krishnan. 3 - biomedical signals and systems. In Sri Krishnan, editor, *Biomedical Signal Analysis for Connected Healthcare*, pages 85–127. Academic Press, 2021.

[LPL23] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176, 2023.

[MBBKH25] Rabah Mokhtari, Samir Brahim Belhouari, Khelil Kassoul, and Abderraouf Hocini. ECG heartbeat classification using progressive moving average transform. 15(1):4285, 2025.

[min] MinMaxScaler.

[MM01] George B Moody and Roger G Mark. The impact of the mit-bih arrhythmia database. *IEEE engineering in medicine and biology magazine*, 20(3):45–50, 2001.

[MSMB23] Manuel Martínez-Sellés and Manuel Marina-Breysse. Current and future use of artificial intelligence in electrocardiography. *Journal of Cardiovascular Development and Disease*, 10(4):175, 2023.

[NSHF23] Kasra Nezamabadi, Neda Sardaripour, Benyamin Haghi, and Mohamad Forouzanfar. Unsupervised ecg analysis: A review. *IEEE Reviews in Biomedical Engineering*, 16:208–224, 2023.

[oP] Unc Eshelman School of Pharmacy. UNC MEDIA | heart rhythms.

[PLT+23] Bach-Tung Pham, Phuong Thi Le, Tzu-Chiang Tai, Yi-Chiung Hsu, Yung-Hui Li, and Jia-Ching Wang. Electrocardiogram heartbeat classification for arrhythmias and myocardial infarction. *Sensors*, 23(6), 2023.

[ptb21] PTB-XL - atrial fibrillation detection, 2021.

[RKN21] Nikita Rafie, Anthony H. Kashou, and Peter A. Noseworthy. Ecg interpretation: Clinical relevance, challenges, and advances. *Hearts*, 2(4):505–513, 2021.

[RSCFCD07] JL Rodriguez-Sotelo, D Cuesta-Frau, and G Castellanos-Dominguez. An improved method for unsupervised analysis of ecg beats based on wt features and j-means clustering. In *2007 Computers in Cardiology*, pages 581–584. IEEE, 2007.

[RSDTPO+10] JL Rodríguez-Sotelo, E Delgado-Trejos, D Peluffo-Ordóñez, D Cuesta-Frau, and G Castellanos-Domínguez. Weighted-pca for unsupervised classification of cardiac arrhythmias. In *2010 Annual International Conference of the IEEE Engineering in Medicine and Biology*, pages 1906–1909. IEEE, 2010.

[SC25] Yasir Sattar and Lovely Chhabra. Electrocardiogram. In *StatPearls*. StatPearls Publishing, Treasure Island (FL), 2025. [Updated 2023 Jun 5].

[sci] 2.3. clustering.

[sl] scikit learn. 2.3. clustering.

[SL05] Leif Sörnmo and Pablo Laguna. Chapter 7 - ecg signal processing. In Leif Sörnmo and Pablo Laguna, editors, *Bioelectrical Signal Processing in Cardiac and Neurological Applications*, Biomedical Engineering, pages 453–566. Academic Press, Burlington, 2005.

[SL19] Shaeke Salman and Xiuwen Liu. Overfitting mechanism and avoidance in deep neural networks. *arXiv preprint arXiv:1901.06566*, 2019.

[SM98] Rosaria Silipo and Carlo Marchesi. Artificial neural networks for automatic ecg analysis. *IEEE transactions on signal processing*, 46(5):1417–1425, 1998.

[SSA17] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316, 2017.

[Sta23] Mayo Clinic Staff. Heart arrhythmia - symptoms and causes, 2023.

[ten] TensorFlow.

[vin22] vincydesy. Evaluation of supervised clustering (purity) from scratch, 2022.

[WHO21] WHO. Cardiovascular diseases (CVDs), 2021.

[WSB+20] Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dieter Kreiseler, Fatima I. Lunze, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset. 7(1):154, 2020.

[YSD23] İbrahim Yazici, Ibraheem Shayea, and Jafri Din. A survey of applications of artificial intelligence and machine learning in future mobile networks-enabled systems. *Engineering Science and Technology, an International Journal*, 44:101455, 2023.