Is het mogelijk om een PID controller te vervangen door een AI?

Onderzoekopzet

Mitchel Reints - 1040953@hr.nl

Hogeschool Rotterdam - Elektrotechniek - PEE51

9 juni 2025

Samenvatting

In dit onderzoek wordt er gekeken hoe een PID regelaar vervangen kan worden door een neuraal netwerkmodel. Het doel van het onderzoek is om te kijken of het mogelijk is om een neuraal netwerkmodel te bouwen dat de PID-regelaar kan simuleren. Dit wordt gedaan door middel van een aantal experimenten waarbij de prestaties van het model worden vergeleken met die van de PID-regelaar. De resultaten van deze experimenten worden gepresenteerd in dit rapport. Daarnaast wordt er gekeken naar de werking van een neuraal netwerkmodel in Tensorflow. Hierbij wordt er een feedforward neuraal netwerkmodel gebouwd. Dit model wordt gebruikt om de PID-regelaar te simuleren.

1 Inleiding

Hogeschool Rotterdam beschikt over de cursus Digitale Systemen (DIS10) over een opstelling waarbij een pingpongbal op een ingestelde hoogte wordt gehouden. Deze opstelling bestaat uit een verticale buis met onderaan een ventilator. Door de luchtdruk van de ventilator wordt de bal omhoog geblazen. Met behulp van een microcontroller wordt de ventilator aangestuurd, zodat de bal op een gewenste hoogte blijft zweven. De regeling van de ventilatorsnelheid gebeurt momenteel met behulp van een PID-regelaar (Proportioneel-Integrerend-Differentieel), waarbij de afwijking tussen de gewenste en werkelijke hoogte continu wordt bijgestuurd.

Hoewel deze opstelling functioneert zoals bedoeld, is het instellen van de PID-parameters vaak een handmatig proces. Het vergt tijd om het systeem goed af te stemmen om het uiteindelijk stabiel te laten reageren op hoogteveranderingen. Daarnaast kunnen veranderingen in de omgeving (zoals temperatuur of luchtvochtigheid) de prestaties van de PID-regelaar beïnvloeden.

De begeleidende docent heeft de vraag gesteld of deze traditionele regelmethode vervangen kan worden door een benadering gebaseerd op machine learning. Machine learning biedt de mogelijkheid om op basis van data te leren hoe het systeem zich moet gedragen. In dit project wordt onderzocht of het mogelijk is om een machine learning model te ontwikkelen dat het gedrag van de PID-regelaar kan nabootsen of zelfs verbeteren.

Dit onderzoek is relevant omdat het laat zien hoe klassieke regeltechniek vervangen kan worden door moderne, zelflerende systeem. Daarnaast sluit het aan bij actuele ontwikkelingen binnen de techniek en biedt het studenten inzicht in zowel embedded systemen als kunstmatige intelligentie. Uiteindelijk is het doel om een demonstratieopstelling te creëren die gebruikt kan worden tijdens bijvoorbeeld open dagen, om zo op een toegankelijke manier beide technieken te tonen.

Het doel van dit onderzoek is om te onderzoeken of een neuraal netwerk het gedrag van een PID-regelaar effectief kan nabootsen of verbeteren in een systeem waarbij een pingpongbal op een ingestelde hoogte wordt gehouden. Hierbij wordt een model ontwikkeld, getraind en geëvalueerd in een gesimuleerde omgeving. Uiteindelijk moet het model geschikt zijn voor implementatie op een embedded platform, met als doel een stabiele en nauwkeurige hoogtecontrole te realiseren. Het onderzoek richt zich op zowel de technische haalbaarheid als de vergelijking met de bestaande PID-regelaar.

Voor dit onderzoek zijn er een aantal onderzoeksvragen geformuleerd die uiteindelijk de hoofdvraag van dit onderszoek moeten beantwoorden:

1. Hoe werkt een PID-regelaar en hoe wordt deze toegepast in de huidige opstelling?
2. Welke soorten neurale netwerken zijn geschikt voor regeltoepassingen in embedded systemen?
3. Hoe kan trainingsdata verzameld worden om het gedrag van de PID-regelaar na te bootsen?
4. Hoe presteert het getrainde neurale netwerk vergeleken met de traditionele PID-regelaar?
5. Wat zijn de beperkingen van het gebruik van een neuraal netwerk in een embedded omgeving?

aan het einden van het onderzoek zullen deze deelvragen worden beantwoord en onderbouwd, zodat de hoofdvraag kan worden beantwoord: *Is het mogelijk om een PID-regelaar te vervangen door een AI?*

Voor dit onderzoek wordt gewerkt met de **NXP FRDM-MCXN947 Development board** als microcontrollerplatform. Hoewel de oorspronkelijke DIS10-opstelling gebaseerd is op de **Texas Instruments SimpleLink Wi-Fi CC3230S Development kit**, is er door de opdrachtgever gekozen het onderzoek uit te voeren op een ander platform. Dit verschil in hardware heeft geen invloed op dit onderzoek, omdat de PID ook geintergreerd kan worden op de NXP FRDM-MCXN947 Development board.

In dit rapport wordt eerst de benodigde theorie besproken (hoofdstuk 2), gevolgd door de gebruikte methoden (hoofdstuk 3) waarin we dit onderzoek doen. Daarna worden de resultaten gepresenteerd (hoofdstuk 4) en besproken (hoofdstuk 5). Het rapport sluit af met conclusies (Hoofdstuk 6) en aanbevelingen (hoofdstuk 7).

Een neuraal netwerk* wordt getraind met data.

2 Theorie

2.1 Werking van een PID-regelaar

Een PID-regelaar (Proportioneel-Integrerend-Differentiërend) is een veelgebruikte terugkoppelingsregelaar in de regeltechniek, die als doel heeft een systeem naar een gewenste waarde (setpoint) te sturen door continu het verschil (de fout) tussen die gewenste waarde en de gemeten systeemoutput te corrigeren. De PID-regelaar bestaat uit drie componenten:

- **Proportioneel (P):** Deze component reageert op de huidige fout. Hoe groter de fout, hoe sterker de aansturing. Dit zorgt voor een directe correctie, maar kan leiden tot een blijvende

*Een *neuraal netwerk* is een model geïnspireerd op het menselijk brein.

fout (steady-state error) als deze component alleen wordt gebruikt.

- **Integrerend (I):** Deze component reageert op de opgetelde (geïntegreerde) fout in de tijd. Hierdoor kan de regelaar kleine resterende fouten wegwerken en wordt het systeem naar het exacte setpoint geduwd. Te veel integratie kan echter leiden tot traagheid of instabiliteit.
- **Differentiërend (D):** Deze component reageert op de snelheid waarmee de fout verandert. Dit helpt om snelle veranderingen te dempen en voorkomt overshoot door het systeem te vertragen voordat het het setpoint bereikt. De D-term maakt het systeem reactiever en stabieler bij plotselinge veranderingen.

De regeloutput $u(t)$ van een PID-regelaar wordt meestal als volgt beschreven:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)\, d\tau + K_d \cdot \frac{de(t)}{dt} \qquad (1)$$

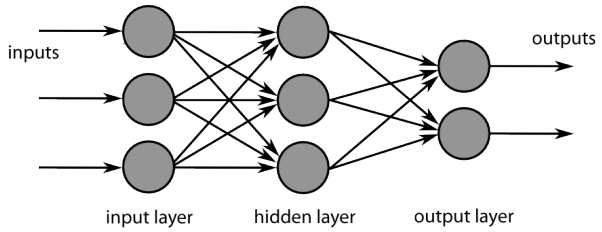
waarbij:

$e(t) = r(t) - y(t)$ (de fout tussen de referentie en gemeten waarde)
$K_p =$ proportionele versterkingsfactor
$K_i =$ integrerende versterkingsfactor
$K_d =$ differentiërende versterkingsfactor

Een goed afgestemde PID-regelaar zorgt voor een snelle respons, minimale overshoot en een stabiele benadering van het setpoint zonder blijvende fout. Hoewel we in dit onderzoek niet in detail ingaan op de afstemming van PID-parameters, is het belangrijk te vermelden dat dit een cruciaal onderdeel is van het gebruik van PID-regelaars in de praktijk. De afstemming wordt meestal handmatig gedaan en dit is meestal een iteratief proces waarbij de waarden van K_p, K_i en K_d worden aangepast op basis van de prestaties van het systeem. Er zijn verschillende methoden voor PID-afstemming, zoals de Ziegler-Nichols-methode, maar deze vallen buiten de scope van dit onderzoek. In dit onderzoek richten we ons op het vervangen van de PID-regelaar door een neuraal netwerk, waarbij we de nadruk leggen op hrt gedrag van de PID regelaar en de mogelijkheid om dit gedrag te repliceren met een neuraal netwerk.

2.2 Neurale netwerken - basisprincipes

Een neuraal netwerk is een wiskundig model dat is geïnspireerd op de werking van het menselijk brein. Het bestaat uit een verzameling onderling verbonden knopen, ook wel neuronen genoemd, die georganiseerd zijn in lagen: een inputlaag, één of meerdere verborgen lagen en een outputlaag. Elke verbinding tussen neuronen heeft een gewicht dat bepaalt hoe sterk een input wordt doorgegeven. Tijdens het doorlopen van het netwerk wordt elke input vermenigvuldigd met het bijbehorende gewicht, opgeteld met een bias en vervolgens door een activatiefunctie gehaald. Deze activatiefunctie bepaalt of en in welke mate het neuron wordt geactiveerd. Het leerproces van een neuraal netwerk gebeurt aan de hand van trainingsdata. Op basis van de fout tussen de voorspelde output en de werkelijke waarde worden de gewichten aangepast met behulp van een algoritme zoals bijvoorbeeld backpropagation in combinatie met gradient descent. Door herhaaldelijk trainen op veel voorbeelden leert het netwerk patronen en relaties herkennen in de data. Neurale netwerken zijn bijzonder krachtig in het modelleren van complexe, niet-lineaire systemen, en worden daarom veel gebruikt in toepassingen zoals beeldherkenning, spraakherkenning en regeltechniek.



Figuur 1: Illustratie van een neuraal netwerk

2.3 vergelijking PID versus NN-regelaars

Zowel PID-regelaars als neurale netwerken kunnen worden ingezet om systemen te regelen, maar hun werking, toepassingsgebied en eigenschappen verschillen sterk.

2.3.1 Verschillen

- **Regelprincipe:** Een PID-regelaar is gebaseerd op een vaste wiskundige formule die werkt met de fout tussen gewenste en gemeten waarden. Een neuraal netwerk leert juist op basis van voorbeelddata een model van het systeemgedrag.
- **Aanpasbaarheid:** PID-regelaars hebben vaste parameters K_p, K_i, K_d die handmatig of via tuning bepaald worden. Neurale netwerken leren automatisch complexe relaties tijdens het trainen, zonder expliciet geprogrammeerde regels.
- **Complexiteit:** PID-regelaars zijn relatief eenvoudig te implementeren en begrijpen. Neurale netwerken zijn complexer en vereisen meer rekenkracht en data.
- **Flexibiliteit:** PID is minder geschikt voor sterk niet-lineaire systemen of systemen met veel vertraging. Neurale netwerken kunnen deze situaties beter aan, mits voldoende training.
- **Transparantie:** PID-regelaars zijn goed uitlegbaar. Neurale netwerken zijn vaak 'black boxes', waarbij het moeilijk is om te achterhalen waarom een bepaalde beslissing wordt genomen.

2.3.2 Overeenkomsten

- Beide regelmethoden kunnen gebruikt worden voor het aansturen van dynamische systemen.
- Beide maken gebruik van foutsignalen (in neurale netwerken impliciet tijdens het trainen).
- Beide kunnen met feedback werken om de systeemoutput te corrigeren.
- Beide vereisen afstemming of training om goed te presteren in een specifieke toepassing.

In veel gevallen kan een neuraal netwerk zelfs worden getraind om het gedrag van een PID-regelaar na te bootsen, wat de basis vormt voor het vervangen of verbeteren van klassieke regelsystemen met AI-gebaseerde methoden.

2.4 keuze van netwerkarchitecturen

Bij het kiezen van een netwerkarchitectuur voor het vervangen van een PID-regelaar door een neuraal netwerk, zijn er verschillende opties beschikbaar, afhankelijk van de aard van het systeem en de beschikbare data. Twee veelgebruikte architecturen zijn feedforward netwerken en recurrente netwerken. Ik heb hierbij gekeken naar de volgende aspecten:

- **Feedforward netwerken:** Deze netwerken zijn geschikt voor statische mapping van invoer naar uitvoer, waarbij de informatie slechts in één richting stroomt. Ze zijn eenvoudig te implementeren en kunnen goed presteren bij systemen met relatief eenvoudige dynamiek. Ze zijn echter minder geschikt voor systemen met tijdsafhankelijke gedragspatronen.
- **Recurrente netwerken (RNN):** Deze netwerken zijn ontworpen om tijdsafhankelijke data te verwerken door informatie van eerdere tijdstappen te onthouden. Ze zijn nuttig voor systemen waarbij de huidige output afhankelijk is van eerdere inputs, zoals bij dynamische systemen. RNN's kunnen echter complexer zijn om te trainen en vereisen meer rekenkracht.
- **Convolutionele netwerken (CNN):** Hoewel CNN's voornamelijk worden gebruikt voor beeldverwerking, kunnen ze ook nuttig zijn voor systemen met ruimtelijke of temporele patronen. Ze zijn minder gebruikelijk voor regeltoepassingen, maar kunnen nuttig zijn als het systeem complexe patronen vertoont.
- **Long Short-Term Memory (LSTM):** Dit is een type RNN dat speciaal is ontworpen om lange-termijn afhankelijkheden te onthouden. LSTM's zijn zeer geschikt voor systemen met complexe dynamiek en tijdsafhankelijke gedragspatronen, maar ze zijn ook complexer en vereisen meer data om goed te presteren.
- **Transformer netwerken:** Deze netwerken zijn ontworpen voor het verwerken van sequentiële data en zijn zeer effectief gebleken in natuurlijke taalverwerking. Ze kunnen ook worden toegepast op regeltoepassingen, vooral wanneer er veel tijdsafhankelijke data beschikbaar is. Ze zijn echter complex en vereisen aanzienlijke rekenkracht.

De keuze van de netwerkarchitectuur hangt af van de specifieke kenmerken van het systeem, de beschikbare data en de vereiste prestaties. Voor eenvoudige systemen kan een feedforward netwerk voldoende zijn, terwijl voor complexere systemen met tijdsafhankelijke gedragspatronen een RNN of LSTM meer geschikt kan zijn. Het is belangrijk om de architectuur te kiezen die het beste past bij de aard van het systeem en de doelstellingen van de regeling.

2.5 Vergelijking van prestaties

De prestaties van een neuraal netwerk in vergelijking met een PID-regelaar kunnen op verschillende manieren worden beoordeeld, afhankelijk van de specifieke toepassing en de doelstellingen van de regeling. Enkele belangrijke prestatie-indicatoren zijn:

- **Nauwkeurigheid:** Hoe goed het neuraal netwerk de gewenste output kan voorspellen in vergelijking met de PID-regelaar. Dit kan worden gemeten aan de hand van de fout tussen de voorspelde en werkelijke waarden.
- **Stabiliteit:** Hoe stabiel het systeem is bij het bereiken van de gewenste output. Een PID-regelaar kan soms oscillaties vertonen, terwijl een goed getraind neuraal netwerk deze kan verminderen of elimineren.
- **Snelheid:** Hoe snel het neuraal netwerk kan reageren op veranderingen in de input. Dit is vooral belangrijk in real-time toepassingen waar snelle aanpassingen nodig zijn.
- **Robuustheid:** Hoe goed het neuraal netwerk presteert onder verschillende omstandigheden, zoals verstoringen of veranderingen in de systeemparameters. Een PID-regelaar kan soms gevoelig zijn voor veranderingen in de omgeving, terwijl een neuraal netwerk mogelijk beter kan generaliseren.
- **Complexiteit:** Hoe complex het neuraal netwerk is in vergelijking met de PID-regelaar. Een PID-regelaar is relatief eenvoudig te implementeren en te begrijpen, terwijl een neuraal netwerk meer rekenkracht en data vereist.
- **Trainings- en implementatietijd:** Hoe lang het duurt om het neuraal netwerk te trainen en te implementeren in vergelijking met de PID-regelaar. Een PID-regelaar kan snel worden ingesteld, terwijl een neuraal netwerk mogelijk meer tijd nodig heeft voor training en afstemming.

De vergelijking van prestaties tussen een neuraal netwerk en een PID-regelaar hangt af van de specifieke toepassing en de doelstellingen van de regeling. In sommige gevallen kan een neuraal netwerk betere prestaties leveren, terwijl in andere gevallen een PID-regelaar voldoende kan zijn. Het is belangrijk om de juiste prestatie-indicatoren te kiezen en de resultaten zorgvuldig te analyseren om een weloverwogen beslissing te nemen.

2.6 Implementatie op embedded systemen

De implementatie van een neuraal netwerk op embedded systemen, zoals microcontrollers of FPGA's, brengt specifieke uitdagingen en overwegingen met zich mee. Deze systemen hebben vaak beperkte rekenkracht, geheugen en energieverbruik in vergelijking met traditionele computers. Daarom is het belangrijk om de volgende aspecten in overweging te nemen bij de implementatie van een neuraal netwerk op embedded systemen:

- **Modeloptimalisatie:** Het neuraal netwerk moet worden geoptimaliseerd voor de beperkte rekenkracht en geheugen van het embedded systeem. Dit kan onder meer het verminderen van het aantal parameters, het gebruik van quantisatie of het toepassen van modelcompressie omvatten.
- **Efficiënte inferentie:** De inferentie (voorspellingsfase) van het neuraal netwerk moet efficiënt worden uitgevoerd. Dit kan worden bereikt door gebruik te maken van speciale hardwareversnellers, zoals Tensor Processing Units (TPU's) of Field Programmable Gate Arrays (FPGA's), die zijn ontworpen voor het uitvoeren van neurale netwerken.
- **Real-time prestaties:** Het neuraal netwerk moet in staat zijn om real-time voorspellingen te doen, wat betekent dat de inferentietijd snel genoeg moet zijn om te voldoen aan de vereisten van de toepassing. Dit kan worden bereikt door het netwerk te optimaliseren voor snelheid en door gebruik te maken van efficiënte algoritmen.
- **Compatibiliteit met hardware:** Het neuraal netwerk moet compatibel zijn met de specifieke hardware van het embedded systeem. Dit kan onder meer het gebruik van specifieke bibliotheken of frameworks omvatten die zijn geoptimaliseerd voor de gekozen hardware.
- **Ondersteuning voor machine learning frameworks:** Het is belangrijk om te kiezen voor machine learning frameworks die ondersteuning bieden voor embedded systemen, zoals TensorFlow Lite, Keras of PyTorch Mobile. Deze frameworks zijn ontworpen om neurale netwerken te optimaliseren en te implementeren op beperkte hardware.

Door rekening te houden met deze aspecten kan een neuraal netwerk effectief worden geïmplementeerd op embedded systemen, waardoor het mogelijk wordt om de voordelen van machine learning te benutten in toepassingen met beperkte middelen. Dit opent de deur naar nieuwe mogelijkheden voor automatisering, controle en optimalisatie in diverse industrieën en toepassingen.

3 Methode

Een van de eerste stappen in dit onderzoek is het verkrijgen van trainingsdata voor het neurale netwerk. Hiervoor zijn er een aantal methoden en technieken nodig om de data te verzamelen en het model te trainen.

3.1 Simulatieomgeving en uitgangspunten

Een van de uitdagingen bij dit onderzoek is

Een van de uitdagingen bij het onderzoeken van de mogelijkheid om een PID-regelaar te vervangen door een AI is het creëren van een geschikte simulatieomgeving. Deze omgeving moet de dynamiek van het systeem nauwkeurig nabootsen en voldoende gegevens genereren voor het trainen van het neurale netwerk. De uitgangspunten voor de simulatie zijn als volgt:

- De simulatie moet de werking van de huidige PID-regelaar nabootsen, inclusief de dynamiek van de pingpongbal en de ventilator.
- De simulatie moet in staat zijn om verschillende scenario's te genereren, zoals veranderingen in de gewenste hoogte, verstoringen en variaties in de omgeving.
- De simulatie moet voldoende gegevens genereren om het neurale netwerk te trainen, inclusief invoer- en uitvoerwaarden die representatief zijn voor de werkelijke situatie.
- De simulatie moet eenvoudig aan te passen zijn, zodat verschillende parameters van het systeem kunnen worden getest en geoptimaliseerd.
- De simulatie moet compatibel zijn met de NXP FRDM-MCXN947 Development board, zodat het getrainde model later eenvoudig kan worden geïmplementeerd op de microcontroller.
- De simulatie moet gebruik maken van een programmeertaal en bibliotheken die geschikt zijn voor zowel de simulatie als de implementatie op de microcontroller, zoals Python met TensorFlow of Keras.

3.2 verzameling en trainingsdata

Een van de uitdagingen van dit onderzoek is het verzalemeln van voldoende en representatieve trainingsdata voor het neurale netwerk. De trainingsdata moet de dynamiek van het systeem nauwkeurig weergeven en variaties bevatten die het model in staat stellen om te generaliseren naar nieuwe situaties. De volgende stappen worden ondernomen om de trainingsdata te verzamelen:

- **Simulatie van de opstelling:** De simulatieomgeving wordt gebruikt om het gedrag van de PID-regelaar en de pingpongbal te modelleren. Hierbij worden verschillende scenario's gecreëerd, zoals veranderingen in de gewenste hoogte, verstoringen en variaties in de omgeving.
- **realtime input en output data genereren van de opstelling:** Voor elk scenario worden invoerwaarden (zoals de gewenste hoogte en eventuele verstoringen) en uitvoerwaarden (zoals de PWM waarde) gegenereerd. Deze waarden vormen de basis voor de trainingsdata.

3.3 Netwerkarchitectuur en configuratie

Voor de

3.4 Trainingsstrategie en hyperparameters tuning

3.5 Validatiemethoden en evaluatiecriteria

4 Resultaten

4.1 Resultaten van de getrainde modellen

4.2 vergelijking met de PID-regelaar

4.3 Effect van variatie in data/robuustheid

4.4 Observatie tijdens simulatie

5 Discussie

5.1 Interpetatie van de resultaten

5.2 Beperkingen van het model

5.3 Lessen en inzichten

5.4 Reflectie op de aanpak

6 Conclusie en Aanbevelingen

6.1 Beantwoording van de hoofdvraag

6.2 Samenvatting van de belangrijke bevindingen

6.3 Aanbevelingen voor implementatie

6.4 Suggesties voor vervolgonderzoek

Referenties

[1] Mike Bowker and Phil Williams. Helsinki and west european security. *International Affairs*, 61(4):607–618, 1985.

Begrippenlijst

PID-regelaar Een regelsysteem dat gebruikmaakt van proportionele, integrale en afgeleide termen.

Neuraal netwerk Een algoritme geïnspireerd op het menselijk brein dat patronen leert herkennen.

Backpropagation Methode om foutmarges in een neuraal netwerk terug te voeren en gewichten aan te passen.

For more details, refer to [1].

A Meetresultaten

In deze bijlage worden de meetresultaten weergegeven die zijn verzameld tijdens de experimenten.

B Tabel met resultaten

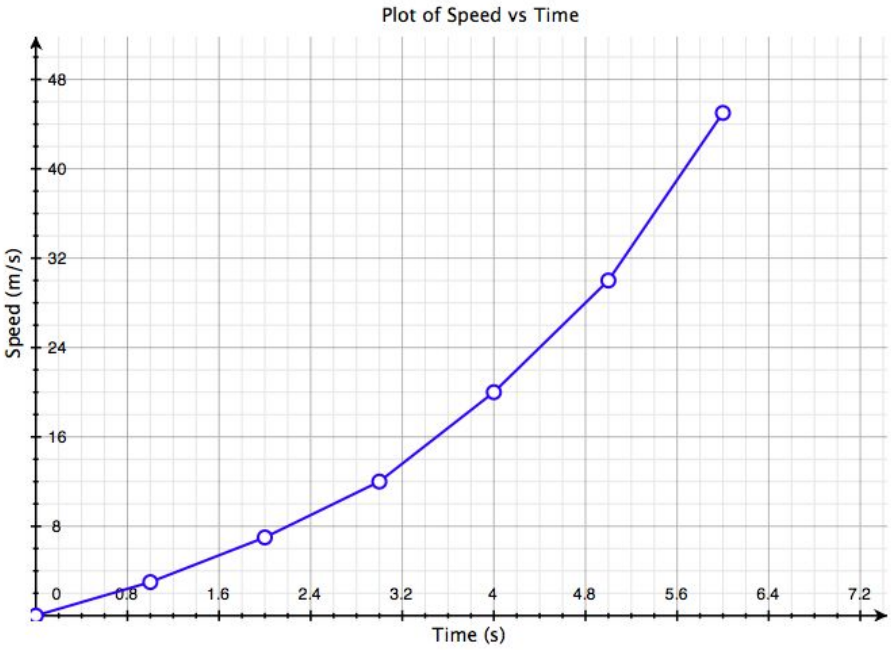
Hieronder staat een overzichtstabel van de gemeten waarden.

Tijd (s)	Spanning (V)	Stroom (A)
0.0	3.30	0.10
1.0	3.28	0.11
2.0	3.27	0.12

Tabel 1: Gemeten waarden tijdens test 1

C Grafiek

Hier zie je een voorbeeld van een grafiek (je moet je eigen figuur invoegen):



Figuur 2: Spanning over tijd

D Broncode

E Arduino-script

```
    def build_feedforward_model(input_layer_size, hidden_layer_size, output_layer_size):
        """Bouw een neuraal netwerkmodel volgens TensorFlow-conventies."""
        model = keras.Sequential(name="Feedfoward_Model")
        model.add(layers.Input(shape=(input_layer_size,), name="Input_Layer"))
        model.add(layers.Dense(hidden_layer_size, activation='sigmoid', name="Hidden_Layer"))
        model.add(layers.Dense(hidden_layer_size_2, activation='sigmoid', name="Hidden_Layer_2"))
        model.add(layers.Dense(output_layer_size, activation='softmax', name="Output_Layer"))
        return model
```