KU LEUVEN

FACULTEIT INDUSTRIELE INGENIEURSWETENSCHAPPEN

TECHNOLOGIECAMPUS GENT

Accelerated edge computing with Neural Networks

Arno PLAETINCK

Promotor: Prof. Dr. Ir. Nobby Stevens

Co-promotoren: Ing. Willem Raes
Ing. Jorik De Bruycker

Masterproef ingediend tot het behalen van de graad van master of Science in de industriële wetenschappen: Industriële Ingenieurswetenschappen Elektronica-ICT Embedded Systems

Academiejaar 2019 - 2020

Dankwoord

Dank aan mezelf
Dank aan een ander

Abstract

Testabstract. Omschrijf hier je thesis.
Trefwoorden: latex, thesis, stijl

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

Lijst van symbolen en acroniemen

Symbols

λ_{ex}	Excitation wavelength
λ_{em}	Emission wavelength
$N(\ldots)$	Noise process

NOG TE DOEN OP HET EINDE VAN DE THESIS

Acronyms

ADC	Analog to Digital Converter
CCD	Charge-Coupled Device

NOG TE DOEN OP HET EINDE VAN DE THESIS

Hoofdstuk 1

Inleiding

De computerwereld maakt de laatste jaren grote stappen op vlak van Machine Learning[10]. Deze vorderingen werden gedreven door onder meer de nieuwste ontwikkelingen op vlak van computerrekenkracht en de vierde industriële revolutie[11]. Door de veelvuldige toepassingsmogelijkheden werd ML een populair en veelbesproken onderwerp. Tegenwoordig kan een bepaalde vorm van Artificiële Intelligentie (AI) in elke sector teruggevonden worden[12]. Van hartritmestoornissen in de medische sector tot commentaarherkenning in de toeristische stiel.

In deze thesis zal men trachten om een benchmark met AI op te stellen waarmee verschillende SBCs met elkaar vergeleken kunnen worden. Dit instrument moet meer inzicht verschaffen in welke mate machineleertechnieken toepasbaar zijn. Er zal vooral gekeken worden naar performantieparameters. Hoe groot is de latency die optreed? Welk verbruik en complexiteit van het netwerk gaat er hier mee gepaard? Ook tussen deze hardware-opties wordt er een afweging gemaakt. Welk toestel is voordeliger in welke situatie? Is een goedkoper toestel tot evenwaardige resultaten in staat? Er zal een oplijsting gemaakt worden van de belangrijkste parameters en met behulp van de benchmark-resultaten zal er een besluit over de SBCs genomen worden.

Hoofdstuk 2

Literatuurstudie

In dit hoofdstuk zal er eerst een afgrenzing gegeven worden waarbinnen de thesis gekozen is. Vervolgens zal een korte introductie gegeven worden tot Artificiële Intelligentie en Machine Learning: welke verschillende vormen bestaan er en wat zijn de mogelijke toepassingsdomeinen. De verschillende leermodellen worden bekeken en er wordt nagegaan hoe de onderdelen een werkend geheel vormen. Vervolgens wordt er een overzicht van de voor- en nadelen van de verschillende technieken gegeven. Er wordt een best bruikbare methode voor de toepassing in deze thesis gekozen. Bij deze keuze zullen de voor- en nadelen gewikt en gewogen worden. Verder wordt er ook de geschiedenis van Single Board Computers aangehaald. Hoe zijn deze toestellen ontstaan, hoe zijn ze geëvolueerd en in welke staat zijn de hedendaagse SBCs. Verder worden ook verschillende voorbeelden van SBCs besproken die in staat zijn om Machine Learning-technieken toe te passen. Deze zullen onderworpen worden aan een benchmark die in paragraaf 2.6 besproken zal worden.

2.1 Kadering

Een branche in de ML die steeds meer in de schijnwerpers staat, is de logistieke sector[13]. Door de steeds verder doorgedreven automatisatie van bedrijven, wordt er ook in bijvoorbeeld magazijnen geopteerd voor het optimaliseren van onder meer het leveren van de verschillende onderdelen en de veiligheid in het magazijn. Gebruik maken van zelfrijdende vorkheftrucks is een mogelijke optie in het verbeteren van de efficiëntie. Niet alleen in het magazijn maar ook op de weg is er een groeiende belangstelling naar zelfrijdende voertuigen, die ontwikkeld worden door grote bedrijven zoals Tesla en Uber. In beide cases zal er een zekere vorm van positiebepaling nodig zijn. Het is van groot belang dat deze bepaling zo accuraat mogelijk plaats vindt, met niet alleen een juiste locatie, maar ook een resultaat dat op zo kort mogelijke tijdsperiode geproduceerd wordt.

Deze nood aan *low latency* kan het verschil betekenen tussen een voertuig dat beslist dat hij moet vertragen of beslist dat hij veilig kan doorrijden maar toch een botsing veroorzaakt. De berekening van die cruciale locatiebepaling kan zowel in de *cloud*, als in de *edge*[14] gebeuren en gebeurt vooral met behulp van ML. Hierbij wordt met cloud verwezen naar het verwerken van data op een locatie ver weg van het voertuig zoals serverzalen. Met edge wordt dan weer een locatie dichtbij het voertuig bedoeld. Dit kan zowel op, als vlakbij het voertuig zijn. Doordat *cloud computing* een toegevoegde latency teweeg brengt van meerdere tientallen milliseconden, de nodige tijd om via het internet de server te bereiken, zal de keuze voor *edge computing* vallen. Indien de berekeningen in de edge plaats vinden zal de afstand tussen reële en virtuele locatie kleiner zijn, wat leidt tot een betere positiebepaling. Deze heeft dan weer tot gevolg dat inschattingen accurater plaats vinden en ongevallen vermeden kunnen worden. Welke hardware men gebruikt kan variëren van applicatie tot applicatie. De berekening zelf wordt uitgevoerd met behulp van AI doordat dit verschillende baten heeft. Deze voordelen worden besproken in de volgende paragrafen.

2.2 Artificiële Inteligentie

AI verwijst naar het simuleren van menselijk intellect in machines die geprogrammeerd worden om de menselijke redenering na te bootsen. Het wordt gezien als de studie van *intelligente agents* die zijn omgeving waarnemen en acties kunnen ondernemen om de kans van het bereiken van een bepaald doel te maximaliseren[15]. Er kan een onderscheid gemaakt worden tussen zowel sterke als zwakke AI.

- **Zwakke AI:** Dit is een vorm van AI die zich bezig houdt met onderzoek in gebieden waar handelswijzen mogelijk zijn die tekenen van intelligentie vertonen, maar niet volwaardig intelligent zijn. Hier worden de meeste vorderingen in voortgebracht, zoals handschriftherkenning of zoekalgoritmen.
- **Sterke AI:** Deze vorm van AI houdt zich bezig met onderzoek dat als doel heeft om software te creëren die zelfstandig kan redeneren en problemen aanpakken.

Het gebruiken van AI kan vele voordelen hebben[16][17]. Zo is het mogelijk om data beter en sneller te gebruiken dan de mens kan. Data kan gelezen en verwerkt worden in geautomatiseerde processen zonder de tussenkomst van een persoon en in een fractie van een seconde. Verder zal er ook, indien er meer beschikbare data zijn, een nog nauwkeuriger responsie gegeven worden. AI wordt als zwakke AI veel toegepast om repetitieve taken met relatief lage complexiteit over te nemen. Een belangrijk onderdeel is Machine Learning dat verder uitgelegd zal worden in volgende paragraaf.

2.3 Machine Learning

ML is een onderdeel van AI en is de studie en modellering van de verschillende leerprocessen dat gebruikt kan worden door verschillende computersystemen[18]. Deze systemen zijn hierdoor in staat om specifieke taken te voltooien zonder rechtstreekse instructies of regels mee te krijgen van de operator. Ze steunen in de plaats op onder meer patroonherkenning om de kans op het succesvol uit te voeren van taken te maximaliseren. Hiervoor wordt er een wiskundig model gebouwd op basis van training-data. Deze mathematische modellen en *datahandling* kan op verscheidene manieren gebeuren. Hieronder worden een aantal gebruikelijke modellen besproken. Er wordt bovendien een afweging gemaakt over welk model het interessantst is voor onze toepassing.

2.3.1 Leeralgoritmes en -technieken

Om ML technieken toe te passen moet men gebruik maken van een bepaald model dat is toegepast op trainingsdata en hierdoor nieuwe data kan verwerken om voorspellingen te maken van de te verwachte resultaten. Er bestaat een hele waaier aan mogelijke modellen. In de volgende paragrafen worden een aantal opties besproken waarna er de verschillende modellen met elkaar vergeleken worden.

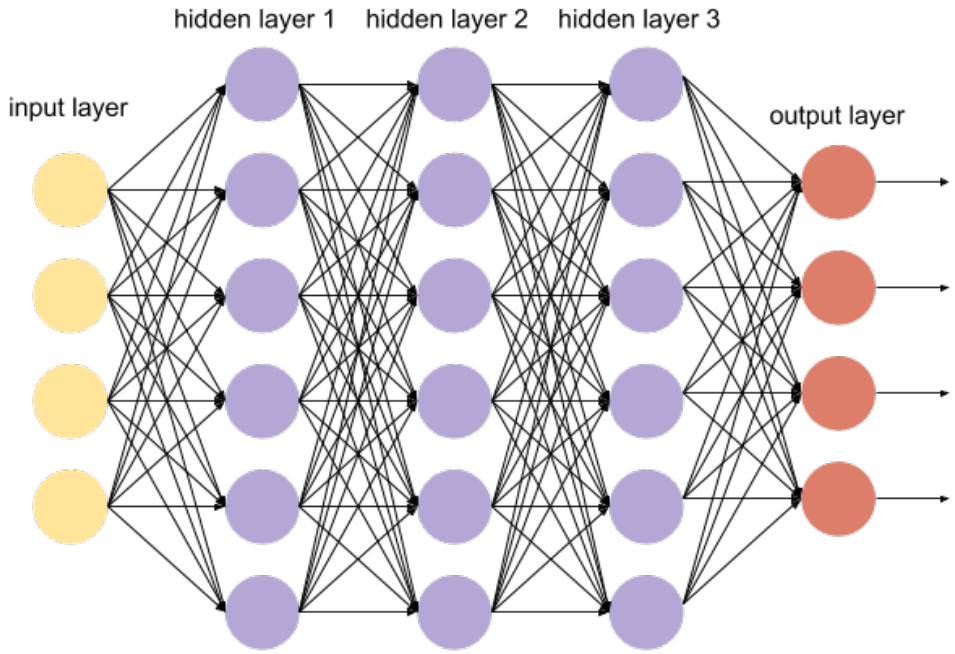
Artificiële Neurale Netwerken

Artificiële Neurale Netwerken (ANN) zijn een term dat gebruikt wordt om algoritmes te omschrijven die structurele gelijkenissen hebben met het interconnectiepatroon van hersenen. Het zijn algoritmes die in staat zijn om een bepaalde taak te leren, en zichzelf te verbeteren. In de meeste gevallen worden er amper richtlijnen of een omschrijving meegegeven. Het systeem ontdekt zelf hoe deze regels in elkaar zitten[**?**]. Hierbij blijft de interpretatie van de input wel nog een belangrijk gegeven dat wordt ingesteld door de programmeur. Een bekend voorbeeld is het herkennen van de cijfers 0 tot 9. Hierin wordt er niet aan het systeem verteld dat het cijfer 8 uit twee cirkels bestaat die verticaal tegen elkaar aansluiten. Het algoritme zal dit gaandeweg ontdekken, met behulp van de vele voorbeelden waar het gebruik van kan maken. Met behulp van veel data kan een algoritme zichzelf verfijnen en zo nauwkeuriger bepaalde cijfers herkennen.

Structuur van Neurale Netwerken Een ANN is een verzameling van nodes die met elkaar verbonden zijn zoals neuronen in de hersens van een mens. Hierbij kan elke neuron een signaal doorgeven naar het volgende neuron waar het signaal verwerkt kan worden en weer doorgegeven kan worden. Hetzelfde principe geldt ook bij Neurale Netwerken (NN) met het verschil dat er meerdere lagen van nodes te onderscheiden zijn.

Lagen Er zijn drie soorten lagen te onderscheiden: een Input Layer, Hidden Layers en een Output Layer. Elke laag is verbonden met de volgende laag door middel van connecties tussen de verschillende nodes. In figuur 2.1 is de algemene vorm van een NN te vinden.

- **Input Layer:** De eerste laag van elke NN is de Input Layer. Deze bestaat uit een aantal inputnodes. Elke inputnode krijgt de ruwe data binnen waar er een operatie op uitgevoerd wordt en vervolgens bepaalde parameters doorgeeft aan de volgende laag. De wijze waarop data geïnterpreteerd worden, vormt een belangrijk vertrekpunt voor het NN.



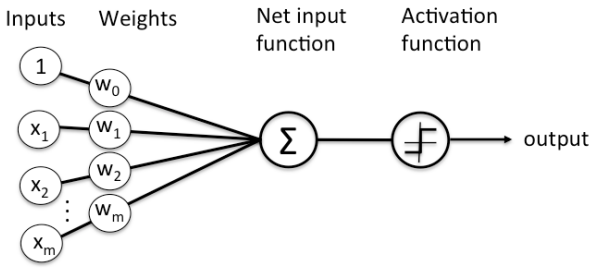
Figuur 2.1: Structuur van een Neuraal Netwerk[1].

- **Hidden Layers:** Na de inputlaag komen een aantal Hidden Layers. Het aantal Hidden Layers en de hoeveelheid nodes binnen één Hidden Layer kan variëren van applicatie tot applicatie en is sterk gerelateerd aan de complexiteit van de toepassing.
- **Output Layer:** Na de Hidden Layers is de laatste laag de Output Layer. Hier worden de laatste operaties uitgevoerd en worden de eindwaarden verkregen waar het resultaat uit afgeleid kan worden.

Nodes Een node is gelijkaardig aan zijn biologisch tegenbeeld. Het krijgt een bepaald aantal inputs, verwerkt deze en geeft een bepaald aantal outputs. Deze inputs en outputs worden van node naar node doorgegeven via verbindingen. Elke node heeft met elke node in de volgende laag een connectie. Elke verbinding draagt een bepaald gewicht. Via dit gewicht kan men de invloed van de huidige node versterken of verzwakken in de volgende node.

Activatie functie De mathematische functie die een node gebruikt voor het verwerken van inputs naar outputs heet de activatie functie. In figuur 2.2 wordt de algemene vorm van een neuron besproken. Deze neuron heeft $m+1$ inputs $(x_0\ t.e.m.\ x_m)$ en bijhorende gewichten $(w_0\ t.e.m.\ w_m)$. Gebruikelijk wordt $x_0 = +1$ genomen. Hierdoor blijven er maar m echte inputs over waardoor er voor een bepaalde output volgende functie opgesteld kan worden. Hierbij is ϕ een van de mogelijke transferfuncties die verder besproken zal worden.

$$y_k = \phi\left(\sum_{j=0}^{m} w_{kj}x_j\right) \tag{2.1}$$



Figuur 2.2: Algemene structuur van een node[2].

Types activatiefuncties De transfer functie of activatiefunctie van een neuron bevat bepaalde eigenschappen die het volledige netwerk kan verbeteren of vereenvoudigen. Hieronder zullen enkele transferfuncties besproken worden.

- **Stapfunctie:** Hier wordt er gekeken naar de verkregen waarde van de gewogen som u van $m+1$ inputs. Bedraagt deze waarde minder dan een bepaalde drempel θ, dan wordt de output gelijkgesteld aan nul, bij een hogere waarde dan weer aan 1. Dit is te zien in formule 2.3. Dit type wordt vooral gebruikt om binaire inputs te verzorgen bij de volgende laag.

$$u = \sum_{j=0}^{m} w_{kj} x_j \tag{2.2}$$

$$y = \begin{cases} 1 & \text{als } u \geq \theta \\ 0 & \text{als } u < \theta \end{cases} \tag{2.3}$$

$$f(x) = \frac{1}{1+e^{-x}} \tag{2.4}$$

- **Lineare Combinaties:** In dit geval is de output niets minder dan de gewogen som vermenigvuldigd met een constante waarbij zoals in formule 2.2 waar er nog een tweede constante wordt opgeteld.

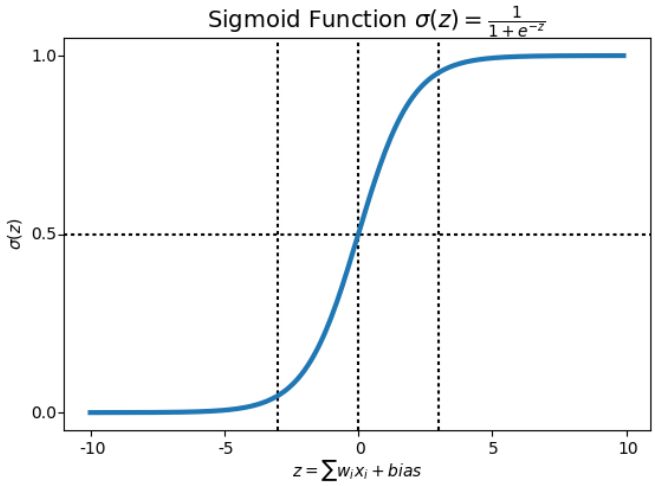
- **Sigmoid:** De Sigmoid functie[19] is een mathematische functie zoals te zien is in figuur 2.4. Het heeft de karakteristieke 'S'-vorm zoals te zien is in figuur 2.3. Door deze activatiefunctie worden inputs omgezet in een waarde tussen 0 en 1 (of -1 en 1, afhankelijk van de conventie).

- **Rectifier:** De rectifier als activatiefunctie[20] is een functie die enkel het positieve deel van zijn argument doorlaat. In vergelijking 2.5 vind je de functie weer waar x de input is van de neuron. Deze is een vector aan waarden die zowel positief als negatief kunnen zijn. Deze functie is ook gekend onder de naam *rectified linear unit (ReLU)*.

$$f(x) = x^+ = max(0,x) \tag{2.5}$$

Beslissingsboom

Het gebruik van een beslissingsboom is een leermethode die met regelmaat terugkomt in de statistiek als een voorspellend model. Men maakt gebruik van observaties rond een bepaalde uitspraak.



Figuur 2.3: De Sigmoid activatiefunctie[3].

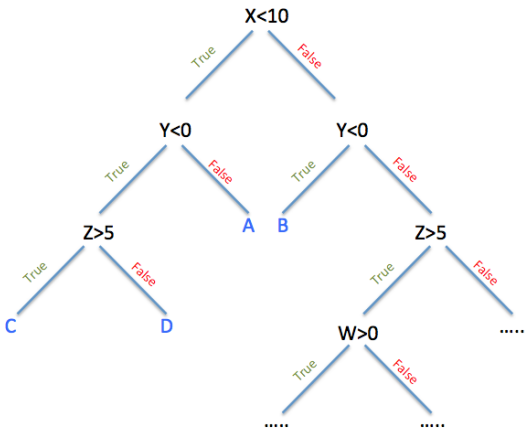
Men kwantificeert deze observaties zodat deze leiden naar een variabele outputwaarde. Indien deze waarde valt onder te verdelen in discrete klassen spreekt men over een *classificatie boom*. Neemt de gezochte variabele eerder een continue vorm aan, dan maakt men gebruik van *regressie bomen*.

Structuur van een beslissingsboom Net zoals bij de NN bestaat een beslissingsboom uit verschillende lagen en nodes. Zoals te zien is in figuur 2.4 wordt er in elke laag een onderscheid gemaakt op basis van een statement of parameter. Deze parameter kan een enkele inputwaarde zijn of een lineaire combinatie van meerdere inputwaarden.

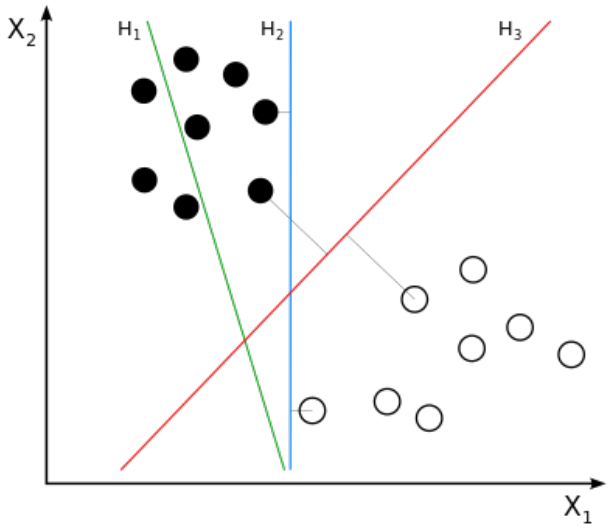
Bij de nodes kan er onderscheid gemaakt worden tussen een gewone node en een eindnode. Bij elke gewone node wordt een bepaald statement geverifieerd en wordt er naar een node overgegaan in de volgende laag op basis van dit statement. Bij een eindnode is het niet meer mogelijk om door te gaan naar een volgende laag, maar wordt er een outputwaarde gegeven.

Support Vector Machines

Support Vector Machines (SVMs) of ook wel gekend als support vector networks is een model dat veelvuldig gebruikt wordt in classificatie en regressie-analyse. SVMs verdelen objecten onder in twee verschillende klassen op basis van een aantal kenmerken. Het is dus een binaire classificeerder. Om aan de hand van kenmerken een onderverdeling te maken moeten deze kenmerken eerst omgezet worden in een vectorruimte. In de trainingsfase wordt er getracht een zo optimaal mogelijke scheiding tussen beide klassen te vinden. Deze optimale scheiding wordt ook wel een *hypervlak* genoemd en ligt op een zo groot mogelijke afstand tussen de dichtstbijgelegen objecten van beide klasses of support vectors. in figuur 2.5 kan u een tweedimensionaal voorbeeld vinden. Hierin is scheidingslijn H1 geen acceptabele scheiding omdat er objecten van de zwarte klasse fout geclassificeerd worden. H2 is acceptabel maar is nog niet optimaal aangezien er weinig foutmarge is voor een nieuw object. H3 is het hypervlak omdat de foutenmarge tussen de twee klassen zo groot mogelijk is. Deze methode is niet alleen bruikbaar in toepassingen met een lineaire scheiding.



Figuur 2.4: Algemene structuur van een beslissingsboom.[4]

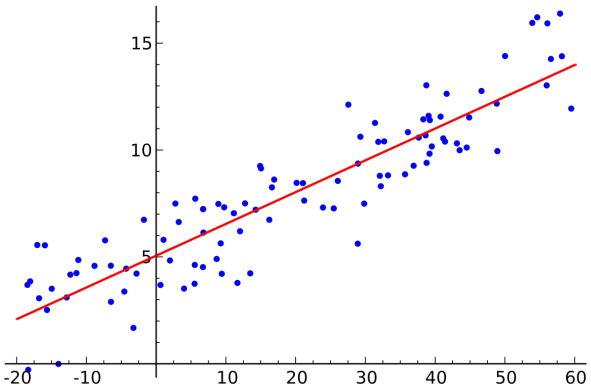


Figuur 2.5: Tweedimensionale Support Vector Machine.[5]

Ook in niet-lineaire gevallen kan men een transformatie uitvoeren om toch een lineaire scheiding te bekomen. Deze hervorming wordt ook wel de *kernel trick*[21] genoemd.

Regressie Analyse

Regressie Analyse is een techniek uit de statistiek[22], die gebruikt wordt om gegevens te analyseren met een specifiek verband. Er bestaat vaak een relatie tussen een afhankelijke variabele en één (of meerdere) onafhankelijke variabelen. De meest voorkomende vorm van regressie analyse is de lineaire regressie, waar men op zoek gaat naar de functie die het dichtst aanleunt bij de data. De functie moet wel vervullen aan specifieke criteria, zo moet de functie bijvoorbeeld een bepaalde orde hebben. Regressie analyse wordt vooral gebruikt voor het voorspellen van nieuwe data of gebeurtenissen. In figuur 2.6 kan je een voorbeeld van lineaire regressie vinden.



Figuur 2.6: Voorbeeld van Lineaire Regressie.[6]

Bayesian Netwerken

Bayesian Netwerken, ook wel probabilistische netwerken genoemd, zijn structuren waarin data op probabilistische wijze geanalyseerd kunnen worden. Dit wil zeggen dat men als output niet enkel de outputs maar ook de onzekerheid hierop krijgt. Men maakt gebruik van gerichte grafen. Hierin bestaan de knopen uit variabelen en de arcs beschrijven de conditionele afhankelijkheden tussen de verschillende knopen. Bayesian Netwerken worden vooral gebruikt om te analyseren wat de bepalende oorzaak is voor een zekere gebeurtenis.

Genetische Algoritmes

Genetische Algoritmes zijn een heuristiek geïnspireerd op het principe van natuurlijke selectie en zijn een klasse binnen evolutionaire algoritmes. Dit type algoritme kan gebruikt worden om oplossingen te vinden in optimalisatie- en zoekproblemen. Door te steunen op biologische principes zoals mutatie, selectie en kruisbestuiving worden er nieuwe *chromosomen* gegenereerd die mogelijks een betere oplossing geven voor een bepaald probleem.

2.3.2 Keuze voor een ML-methode

Om de keuze voor een bepaalde ML-techniek te verantwoorden, zal dit hoofdstuk de voor- en nadelen behandelen van de voornaamste technieken die via regressie of classificatie toegepast kunnen worden[23].

Regressie

- **Neurale Netwerken:**

  Voordelen: Neurale netwerken zijn de meest gebruikte toepassing in verschillende domeinen. NN kunnen uitstekend omgaan met onder meer beeld-, audio- en tekstdata en deze verwerken. Verder kan de architectuur ook nog gemakkelijk aangepast worden aan de toepassing door te variëren in het aantal lagen of nodes. Het gebruik van de hidden layers vermindert ook het hanteren van feature engineering.

Nadelen: Neurale netwerken zijn minder bruikbaar voor *general-purpose* algoritmes door de grote hoeveelheid data die er voor nodig zijn. In dat geval is het beter om voor beslissingsbomen te kiezen. Bovendien vragen ze veel computationeel vermogen voor het trainen van het netwerk en vragen veel expertise voor kleine aanpassingen zoals aan de architectuur of hyperparameters.

- **Regression Trees:**

  Voordelen: Beslissingsbomen met nadruk op regressie zijn in staat om niet-lineaire relaties te leren en zijn robuust voor uitschieters in de te verwerken dataset.

  Nadelen: Regression trees zijn vatbaar voor overfitting indien er te veel gebruik gemaakt wordt van branches. Bij bomen is het ook mogelijk om vertakkingen te blijven maken tot het een exacte kopie voorstelt van de trainingsdata.

- **Lineaire Regressie:**

  Voordelen: Dit is een eenvoudige methode om zowel te begrijpen als uit te leggen. Daarnaast kan er een eenvoudige bescherming tegen overfitting geïmplementeerd worden.

  Nadelen: Niet-lineaire relaties zijn een zwak punt voor lineaire regressie. Het is moeilijk om een correcte fitting te vinden voor een gegeven ingewikkelde relatie. Bovendien is het onvoldoende flexibel om complexe patronen op te vangen.

Classificatie

- **Neurale Netwerken:**

  Voordelen: NN blijven uitstekend presteren bij het classificeren van audio-, tekst- en beeldherkenning.

  Nadelen: Er is nood aan grote hoeveelheden data om het model te trainen en minder geschikt als general-purpose algoritme.

- **Classification Trees:**

  Voordelen: Verrichten zeer goed werk in praktijk. Ze zijn robuust voor uitschieters, schaalbaar voor meerdere klassen en kunnen niet-lineaire grenzen op natuurlijke wijze modelleren dankzij de hiërarchische structuur.

  Nadelen: Classification trees zijn vatbaar voor overfitting indien er te veel gebruik gemaakt wordt van branches. Bomen hebben vaak de neiging om branches aan te maken tot het een exacte kopie voorstelt van de trainingsdata.

- **Support Vector Machines:**

  Voordelen: SVMs zijn in staat om niet-lineaire beslissingsgrenzen te modelleren en hebben een sterke robuustheid tegen overfitting, vooral in hogere dimensionale vectorruimtes.

  Nadelen: SVMs zijn heel erg geheugen intensief. Ze vragen ook meer expertise in het afstemmen door het grote aanbod in mogelijke kernels. SVMs hebben de eigenschap om minder effectief te zijn bij het schalen naar grotere datasets.

- **Geregulariseerde Regressie:**

Voordelen: Outputs hebben een gemakkelijk leesbare probabilistische interpretatie. Ook kan er bescherming tegen overfitting geïmplementeerd worden en kunnen modellen eenvoudig geüpdatet worden.

Nadelen: Niet-lineaire relaties zijn een zwak punt voor lineaire regressie. Het is moeilijk om een correcte fitting te vinden voor een gegeven ingewikkelde relatie. Bovendien is het onvoldoende flexibel om complexe patronen op te vangen.

Besluit

Na een afweging gedaan te hebben van de voor- en nadelen van de voornaamste kandidaten bij zowel regressie als classificatie toepassingen zal in het kader van deze thesis voor NN gekozen worden. Er wordt vooral gesteund op het feit dat NN uitstekend werk levert in beide klassen in het analyseren van data. Bovendien zijn de voornaamste nadelen minder van toepassing in het kader waarin NN toegepast zal worden. Het is vooral van belang hoe de netwerken in de executiefase presteren. Het trainen van de verschillende netwerken met grote hoeveelheden data kan op aparte systemen gebeuren. De trainingsfase is dus minder van belang voor het doel van deze thesis. Bovendien is het niet nodig om een breed general-purpose netwerk te voorzien.

Een mogelijk alternatief voor NN is het gebruiken van Regressie Analyse. De bepalende factor hiervoor is dat het karakter van de verscheidene applicaties vaak uit lineaire relaties bestaat.

2.3.3 Leertechnieken

Er zijn verschillende mogelijkheden om een neuraal netwerk te laten leren. De drie belangrijkste methodes om een mathematische functie te verkrijgen worden hieronder opgesomd[12].

Supervised Learning

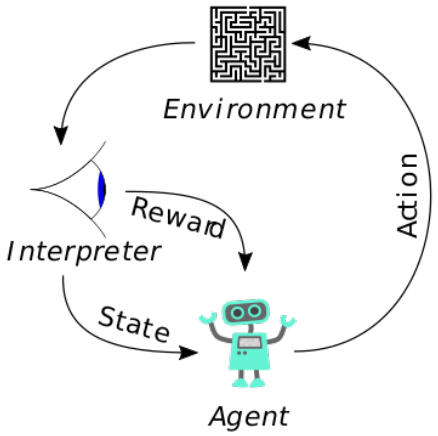
Deze techniek maakt gebruik van gepaarde datasets van inputobjecten en de te verwachten outputobjecten. Het doel is om een mathematische functie te creëren waarbij de gegenereerde outputs zo nauw mogelijk overeenkomen met de gelabelde outputs uit de datasets. Men optimaliseert deze mathematische functie door iteratief te trainen. De bijgeschaafde functie kan dan ook gebruikt worden voor nieuwe datasets zonder gelabelde output. Hierbij zal hij zelf outputwaardes genereren. Een toepassing van Supervised Learning is bijvoorbeeld het detecteren van spam met een trainingset van al gelabelde e-mails.

Unsupervised Learning

Unsupervised learning is een techniek die gebruik maakt van Hebbian Learning om onbekende patronen te herkennen in datasets. De twee meest gebruikte methodes onder Unsupervised Learning zijn principal component en cluster analysis. Principal component maakt gebruik van orthogonale transformaties om een set van mogelijke afhankelijke variabelen om te zetten in een set van lineaire onafhankelijke variabelen. In cluster analyse wordt er getracht om een groep objecten te identificeren en te verdelen in een cluster van gelijkaardige objecten. Een belangrijke toepassing van Unsupervised Learning is het clusteren van gelijkaardige documenten op basis van de inhoud van de tekst.

Reinforcement Learning

Deze leertechniek heeft betrekking tot hoe agents acties moeten ondernemen in een omgeving om een bepaald attribuut te maximaliseren. Het onderscheidt zich van Supervised en Unsupervised Learning door de onafhankelijkheid van gelabelde outputdatasets. De techniek heeft als doel om een evenwicht te vinden tussen exploratie van ongekend gebied en exploitatie van de huidige kennis. In figuur 2.7 kan je een eenvoudige routine vinden van Reinforcement Learning-algoritme. Hierbij maakt een agent een bepaalde actie gebaseerd op de staat waar hij in is. Deze actie heeft in een omgeving een zekere invloed die door een Interpreter beoordeeld wordt en een score toekent. De agent kan deze verandering daarna gebruiken om zichzelf te verbeteren en zijn acties aanpassen. Een van de vele mogelijke toepassingen van Reinforcement Learning is het aanleren van schaken door enkel mee te geven of het algoritme gewonnen of verloren heeft.



Figuur 2.7: Routine bij Reinforcement Learning[7].

2.4 Evolutie SBC

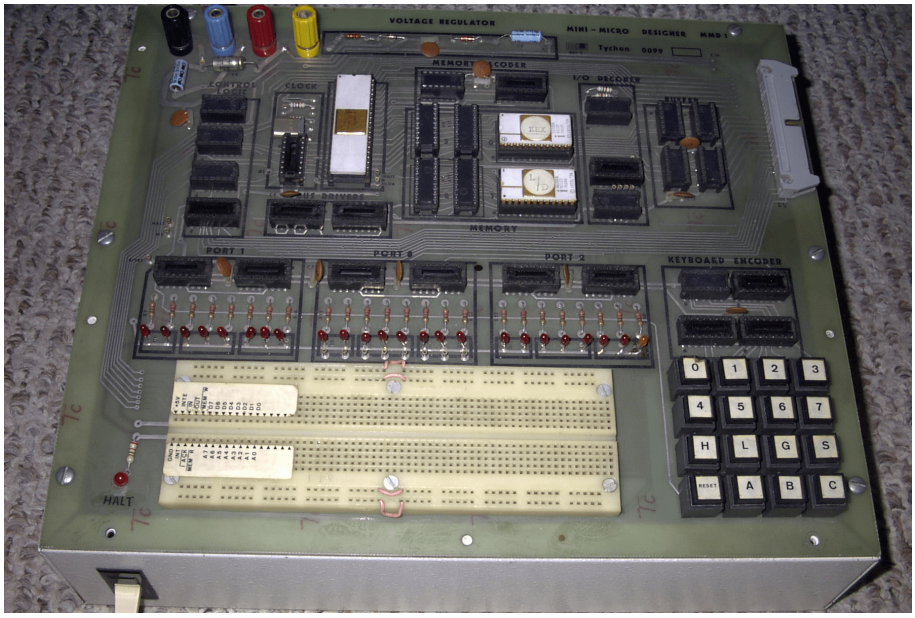
Een SBC is een volledige computer gemaakt op 1 enkele printplaat. Het bevat onderdelen zoals een microprocessor, geheugen, inputs en outputs. De eerste SBC werd ontwikkeld als een voorstel-hulpmiddel bij educatieve doelstellingen of het gebruik als een embedded computer controller. Tegenwoordig zijn ook vele (draagbare) computers geïntegreerd op één printplaat. Het grote verschil met (draagbare) computers is dat er geen nood is aan expansion slots zoals bijvoorbeeld voor RAM-geheugen of een Graphics Processing Unit (GPU).

2.4.1 Geschiedenis

De eerste echte SBC was de zogenaamde ”dyna-micro” uit figuur 2.8 die later de naam ”MMD-1”(Mini-Micro Designer 1) kreeg. Dit toestel werd uitgegeven in 1976 en werd populair doordat het werd gepresenteerd in het destijds ’BugBook’. Een andere vroege SBC was de KIM-1 (Keyboard Input Monitor 1) uit hetzelfde jaar. Beide machines werden voor ingenieurs geproduceerd en ontworpen maar vonden een breed publiek onder de hobbyisten waar het heel populair werd. Later kwamen nog andere namen zoals de Ferguson Big Board en de Nascom.

Naarmate de markt voor desktops en PC’s groeide, nam de belangstelling voor SBC in computers meer en meer af. De focus van de markt werd verlegd naar een moederbord met de belangrijkste componenten en dochterborden voor periferiecomponenten zoals seriële poorten. De voornaamste reden hiervoor was dat de componenten groot waren. Alle onderdelen op dezelfde printplaat zou zorgen voor een onpraktisch ontwerp met grote afmetingen. Deze beweging was echter tijdelijk en naarmate de vorderende technologie kleinere componenten kon leveren, werden onderdelen terug naar het mainframe verschoven. Tegenwoordig kunnen de meeste moederborden terug als SBC beschouwd worden.

In het jaar 2004 werd er in Italië een nieuwe microcontroller uitgebracht onder de naam ”Arduino”. Dit ontwerp had, naast het voordeel van compact en goedkoop te zijn, ook nog eenvoudigheid mee. Door de eenvoud werd het Arduino-platform snel populair onder techneuten van alle soorten. Twee jaar later bracht de Universiteit van Cambridge een nieuwe goedkope SBC uit. De bekende Raspberry Pi werd gelanceerd voor de prijs van $35. Het hoofddoel van dit project was een nieuw leermiddel om te programmeren maar werd door het grote aantal applicaties ook zeer populair.



Figuur 2.8: Eerste SBC: MMD-1 [8]

De laatste jaren kende een grote explosie aan nieuwe SBCs. Een hele reeks nieuwe namen verschenen. Banana Pi, Beaglebone, Intel Galileo, Google Coral Dev en Asus Tinker Board zijn maar enkele van de vele voorbeelden. Deze toestellen hebben vaak een processor gebaseerd op de x86- of ARM-series en maken gebruik van een Linux besturingssysteem zoals Debian.

2.5 Assortiment aan 'off the shelf' toestellen

In deze sectie wordt er een kort overzicht gegeven van de te gebruiken SBCs. Er werd gekozen om met vijf verschillende toestellen te werken die verspreid liggen over het spectrum. Zo zijn er zowel goedkope als kostelijke apparaten, populaire boards als minder gekende SBCs. De specificaties van de verscheidene toestellen wordt ook meegegeven.

2.5.1 Beaglebone AI

BeagleBone Ai (BB AI) is een SBC dat verder bouwt op de succesvolle BeagleBoard-series[24]. Het is een open source project met een op Linux gebaseerd aanpak. De BB AI probeert het gat tussen kleinere SBC en krachtigere industriële computers te overbruggen. Met dank aan de krachtige Texas Instruments AM5729 CPU kunnen ontwikkelaars de krachtige System On Chip (SoC) gebruiken van een hele brede waaier aan toepassingen op het AI terrein. De BB AI maakt het makkelijk om dat terrein te ontdekken en verkennen. Door gebruik te maken van onder andere embedded-vision-engine (EVE) cores die steunen op een geoptimaliseerde TIDL machine learning OpenCL API met gebruiksklare hulpmiddelen kan je terecht in alledaagse automatisatie in industriële, commerciële en thuisapplicaties.

Specificaties

- **GPU:** Niet van toepassing
- **CPU:** Texas Instruments AM5729
- **Memory:** 16 GB on-board eMMC flash
- **Storage:** 1GB RAM + micro SD-slot
- **Power:** 5 Watt
- **Prijs:** $139.37

2.5.2 Coral Dev Board

De Coral Dev Board is een development board gemaakt door het Amerikaanse technologiebedrijf Google[25]. Het board is ontworpen om het ontwikkelen van on-device ML producten te vergemakkelijken. Hiervoor heeft het een aantal belangrijke voordelen gekregen door zijn designers. Het is vooral de aangepaste Tensor Processing Unit (TPU) AI chip die hier opvalt. De TPU is een Application Specific Integrated Circuit (ASIC) speciaal ontworpen voor NN ML, en is in staat om video in hoge resolutie te analyseren aan 30 frames per second. Deze System-on-Module (SoM) is geoptimaliseerd om Tensorflow Lite te kunnen draaien aan meerdere tera operations per second (TOPS).

Specificaties

- **GPU:** Integrated GC7000 Lite Graphics
- **CPU:** NXP i.MX 8M SOC (quad Cortex-A53, Cortex-M4F) + coprocessor Google Edge TPU
- **Memory:** 8 GB on-board eMMC flash
- **Storage:** 1GB RAM LPDDR4 + micro SD-slot
- **Power:** 0.5 watts for each TOPS - 2 Watt
- **Prijs:** $149.99

2.5.3 Nvidia Jetson Nano

De Jetson Nano is een populair bord uit de Jetson Series van Nvidia[26]. Het is een kleine maar krachtige computer ontwikkeld voor embedded applicaties en low-power AI-Internet-of-Things (IoT). Deze SBC wordt ondersteund door meerdere bibliotheken in sectoren zoals deep learning, computer vision, beeld en multimedia. De hardware bevat zowel een GPU als een Central Processing Unit (CPU). De GPU bestaat uit een krachtige Maxwell architectuur die beeld kan decoderen aan 500 MP/sec. De CPU is van het type Cortex-A57 met 4 kernen.

Specificaties

- **GPU:** 128-core Maxwell met 128 CUDA-cores
- **CPU:** Quad-core ARM A57 @ 1.43 GHz
- **Memory:** 4 GB 64-bit LPDDR4 25.6 GB/s
- **Storage:** micro SD-slot
- **Power:** 5 W
- **Prijs:** $99

2.5.4 Nvidia Jetson TX2

De TX2, uit de zelfde Jetsonserie, is de high end versie van de hiervoor besproken Nano[27]. Het is het snelste en meest power-efficiëntst van de embedded AI toestellen die gebruikt zal worden in deze thesis. De TX2 verbruikt een 7.5 Watt en brengt het zware AI-rekenwerk naar de edge. Zijn bekwame GPU met 256 CUDA kernen en een duo CPU zijn in staat om de meest geavanceerde Machine Leertechnieken uit te voeren. De grote geheugenvoorzieningen zorgen bovendien dat de datasetgrootte geen beperkende factor meer kan spelen. Verder wordt er nog gezorgd voor een grote ondersteuning via een grote variatie aan hardware interfaces. Hierdoor wordt het integreren van producten aanzienlijk makkelijker.

Specificaties

- **GPU:** 256-core NVIDIA Pascal GPU architecture with 256 NVIDIA CUDA cores
- **CPU:** Dual-Core NVIDIA Denver 2 64-Bit & CPU Quad-Core ARM Cortex-A57 MPCore
- **Memory:** 8GB 128-bit LPDDR4 Memory 1866 MHx - 59.7 GB/s
- **Storage:** 32GB eMMC 5.1
- **Power:** 7,5 - 15 W
- **Prijs:** $399

2.5.5 Raspberry Pi

De laatste SBC die hier besproken is de Raspberry Pi 4[28]. Dit is een heel goedkoop en eenvoudig device. Het komt uit de heel gekende Raspberry-reeks zoals al besproken in paragraaf 2.4.1. Ondanks de enorm lage prijs heeft de vierde editie nog mooie specificaties. Het heeft een behoorlijke Cortex Quad core-CPU, degelijk geheugen met de mogelijkheid om uit te breiden m.b.v. een SD-kaart en een verrassend laag verbruik.

Specificaties

- **CPU:** Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- **Memory:** 1GB, 2GB or 4GB LPDDR4-3200 SDRAM
- **Storage:** micro SD-kaart
- **Power:** 2,8 - 5,2 W
- **Prijs:** $35

2.6 Benchmarking van ML algoritmes

Om betekenisvolle resultaten te verkrijgen is het nodig om een goed bruikbare en representatieve benchmark op te stellen. Een benchmark is een onderzoek waarbij de prestaties van programma's met elkaar vergeleken worden. Dit komt tot stand als elk programma op identieke wijze wordt onderzocht. Hoe de benchmark exact in elkaar zit is gebonden aan de kwaliteitscriteria die onderzocht worden. Om ervoor te zorgen dat de benchmark onafhankelijk is van zowel het specifiek veld als toepassing, is het nodig dat er aan een aantal karakteristieken wordt voldaan[29].

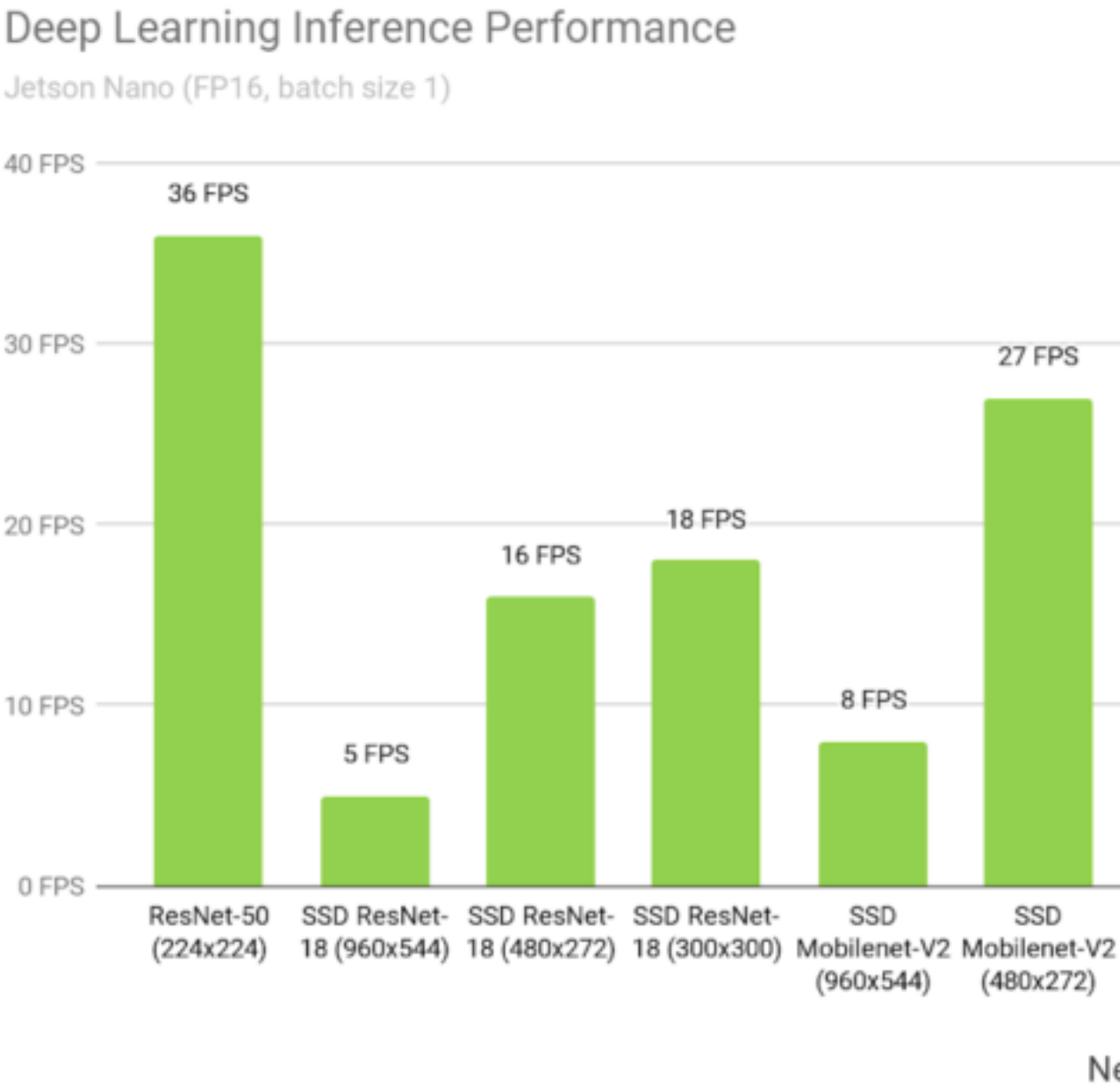
- **Vergelijkbaarheid:** Benchmarks moeten zodanig opgesteld zijn, dat het evident is wat ze vergelijken en de conclusie ondubbelzinnig is.
- **Herhaalbaarheid:** Bij het herhalen van de test onder gelijkaardige omstandigheden moeten gelijkaardige resultaten gehaald worden.
- **Goed gedefinieerde methodologie:** De werkwijze, methode en aannames moeten voldoende gedocumenteerd en gestaafd worden.
- **Configureerbaar:** Benchmarks moeten beschikken over parameters die aangepast kunnen worden naar het specifieke probleem dat wordt behandeld.

2.6.1 Bestaande benchmarks

De ontwikkelaars van de Jetson Nano vermelden op de Nvidia-website[9] verschillende Deep Learning Inference Benchmarks (DLIB) waarbij de auteur verscheidene op voorhand getrainde Deep Learning (DL) modellen toepassen op het Nano model. Deze modellen zijn gebaseerd op een brede waaier aan populaire ML frameworks zoals Tensorflow, Caffe, PyTorch en Keras. Bovendien zijn de applicaties ook gespreid over meerde toepassingen zoals beeldherkenning, objectdetectie, positiebepaling en anderen. De modellen en resultaten zijn te vinden in figuur 2.9.

Ook voor de Jetson TX2 bestaat er een benchmark zoals voorgesteld in [30]. Het gaat over een general-purpose benchmark die een aantal parameters controleert. Het gaat over variabelen zoals Frames per Second (FPS), *inference time*, GPU temperatuur en geheugen verbruik. Met inference time wordt de tijd bedoeld om een berekening te doen in GPU en CPU m.a.w. de latency veroorzaakt door de SBC. Deze gegevens worden uit de data gehaald door middel van verschillende DL modellen toe te passen op Convolutional Neural Networks (CNN). Deze modellen passen ze toe op twee verschillende datasets: Microsoft COCO dataset voor objectdetectie, een foto bank met een grote verscheidenheid aan categorieën, en de KITTI Stereo Vision databank. De KITTI-databank bestaat uit 400 foto's specifiek bedoeld als benchmark fotoset.

Er bestaan ook meer algemenere benchmarks om vergelijkingen tussen computersystemen te maken. Zo bestaat er ook de LINPACK benchmarks[31]. Dit is een programma waar de snelheid gemeten kan worden waarmee een computer in staat is om n bij n matrix van lineaire vergelijkingen op te lossen. Ondanks dat het een veelgebruikte benchmark is, zijn er wel nog bedenkingen over de werkwijze. Zo bestaat de kerntaak maar uit een enkele computationele taak die onmogelijk de algemene performantie van een systeem kan weergeven. Desondanks geeft de LINPACK benchmark een goed karakteristiek beeld van computersystemen. De meest bekende toepassing waar LINPACK in toegepast wordt is de ranking van de beste 500 supercomputers ter wereld[32]



Figuur 2.9: Modellen en resultaten van de benchmark op de Jetson Nano[9].

Hoofdstuk 3

Data verwerving

Hoofdstuk 4

Opbouw code

Hoofdstuk 5

Resultaten

Hoofdstuk 6

Conclusie

Bibliografie

[1] https://github.com/drewnoff/spark-notebook-ml-labs/tree/master/labs/DLFramework, accessed on 24.12.2019.

[2] https://pathmind.com/wiki/neural-network, accessed on 23.12.2019.

[3] https://towardsdatascience.com/multi-layer-neural-networks-with-sigmoid-function-deep-learning-for-rookies- accessed on 23.12.2019.

[4] https://medium.com/machine-learning-bites/machine-learning-decision-tree-classifier-9eb67cad263e, accessed on 24.12.2019.

[5] https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781788994170/5/ch05lvl1sec36/support-vector-machines, accessed on 20.12.2019.

[6] https://en.wikipedia.org/wiki/Regression_analysis, accessed on 19.12.2019.

[7] http://webindream.com/reinforcement-learning/, accessed on 23.12.2019.

[8] https://nl.wikipedia.org/wiki/Singleboardcomputer, accessed on 23.12.2019.

[9] https://devblogs.nvidia.com/jetson-nano-ai-computing/, accessed on 17.02.2020.

[10] M. R. Minar and J. Naher, "Recent advances in deep learning: An overview," 02 2018.

[11] J. Bloem, M. Van Doorn, S. Duivestein, D. Excoffier, R. Maas, and E. Van Ommeren, "The fourth industrial revolution," *Things Tighten*, vol. 8, 2014.

[12] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[13] L. Barreto, A. Amaral, and T. Pereira, "Industry 4.0 implications in logistics: an overview," *Procedia Manufacturing*, vol. 13, pp. 1245–1252, 2017.

[14] H. Li, K. Ota, and M. Dong, "Learning iot in edge: Deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, Jan 2018.

[15] D. I. Poole, R. G. Goebel, and A. K. Mackworth, *Computational intelligence*. Oxford University Press New York, 1998.

[16] https://www.frankwatching.com/archive/2017/02/06/artificial-intelligence-6-redenen-waarom-je-juist-nu-moet- accessed on 27.1.2020.

[17] https://www.sas.com/en_us/insights/analytics/what-is-artificial-intelligence.html, accessed on 27.1.2020.

[18] J. G. Carbonell, R. S. Michalski, and T. M. Mitchell, “An overview of machine learning,” in *Machine learning*. Springer, 1983, pp. 3–23.

[19] J. Han and C. Moraga, “The influence of the sigmoid function parameters on the speed of backpropagation learning,” in *International Workshop on Artificial Neural Networks*. Springer, 1995, pp. 195–201.

[20] V. Nair and G. E. Hinton, “Rectified linear units improve restricted boltzmann machines,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[21] F. Sun, “Kernel coherence encoders,” 2018.

[22] I. Sieben and L. Linssen, “Logistische regressie analyse: een handleiding,” *Geraadpleegd via www. ru. nl/publish/pages/525898/logistischeregressie. pdf*, 2009.

[23] https://elitedatascience.com/machine-learning-algorithms, accessed on 21.12.2019.

[24] https://beagleboard.org/ai, accessed on 23.12.2019.

[25] https://venturebeat.com/2019/10/22/googles-coral-ai-edge-hardware-launches-out-of-beta/, accessed on 23.12.2019.

[26] https://developer.nvidia.com/embedded/jetson-nano, accessed on 23.12.2019.

[27] https://developer.nvidia.com/embedded/jetson-tx2, accessed on 23.12.2019.

[28] https://www.raspberrypi.org/products/raspberry-pi-4-model-b/, accessed on 23.12.2019.

[29] L. A. Libutti, F. D. Igual, L. Pinuel, L. De Giusti, and M. Naiouf, “Benchmarking performance and power of usb accelerators for inference with mlperf.”

[30] L. P. Bordignon and A. von Wangenheim, “Benchmarking deep learning models on jetson tx2.”

[31] https://en.wikipedia.org/wiki/LINPACK_benchmarks, accessed on 27.02.2020.

[32] https://www.top500.org/project/linpack/, accessed on 27.02.2020.

Bijlage A

Een aanhangsel

sdfsffqsfsf

Bijlage B

Beschrijving van deze masterproef in de vorm van een wetenschappelijk artikel

Bijlage C

Poster

FACULTEIT INDUSTRIELE INGENIEURSWETENSCHAPPEN
TECHNOLOGIECAMPUS GENT
Gebroeders De Smetstraat 1
9000 GENT, België
tel. + 32 92 65 86 10
fax + 32 92 25 62 69
iiw.gent@kuleuven.be
www.iiw.kuleuven.be