KU LEUVEN

FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN

CAMPUS DE NAYER SINT-KATELIJNE-WAVER

Mobile Deep Visual Detection and Recognition

Ondertitel (facultatief)

Thijs VERCAMMEN

Promotor(en): Prof. dr. ir. Toon Goedéme

Co-promotor(en): Ing. Floris De Feyter

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

Academiejaar 2021 - 2022

Voorwoord

Het voorwoord vul je persoonlijk in met een appreciatie of dankbetuiging aan de mensen die je hebben bijgestaan tijdens het verwezenlijken van je masterproef en je hebben gesteund tijdens je studie.

Samenvatting

De (korte) samenvatting, toegankelijk voor een breed publiek, wordt in het Nederlands geschreven en bevat **maximum 3500 tekens**. Deze samenvatting moet ook verplicht opgeladen worden in KU Loket.

Abstract

Het extended abstract of de wetenschappelijke samenvatting wordt in het Engels geschreven en bevat **500 tot 1.500 woorden**. Dit abstract moet **niet** in KU Loket opgeladen worden (vanwege de beperkte beschikbare ruimte daar).

Keywords: Voeg een vijftal keywords in (bv: Latex-template, thesis, ...)

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

Lijst van symbolen

Maak een lijst van de gebruikte symbolen. Geef het symbool, naam en eenheid. Gebruik steeds SI-eenheden en gebruik de symbolen en namen zoals deze voorkomen in de hedendaagse literatuur en normen. De symbolen worden alfabetisch gerangschikt in opeenvolgende lijsten: kleine letters, hoofdletters, Griekse kleine letters, Griekse hoofdletters. Onderstaande tabel geeft het format dat kan ingevuld en uitgebreid worden. Wanneer het symbool een eerste maal in de tekst of in een formule wordt gebruikt, moet het symbool verklaard worden. Verwijder deze tekst wanneer je je thesis maakt.

b	Breedte	$[mm]$
A	Oppervlakte van de dwarsdoorsnede	$[mm^2]$
c	Lichtsnelheid	$[m/s]$

Lijst van afkortingen

CNN RoIs ReLu

Hoofdstuk 1

Situering en doelstelling

1.1 Situering

Tegenwoordig wordt deep learning steeds meer en meer gebruikt om beeldverwerking problemen op te lossen. Via neurale netwerken kunnen we met meer en betere features werken om de afbeeldingen te analiseren. Maar veel van deze modelen hebben behoorlijk wat rekenkracht en geheugen nodig om te werken. Ook is er steeds meer intresse naar real-time toepassingen waarvan het resultaat zo snel mogelijk beschikbaar moet zijn. Dit wordt moeilijk bij veel hedendaagse systemen waarbij de foto eerst genomen moet worden en vervolgens door een computer geanaliseerd moet worden, omdat hedendaagse systemen veel rekenwerk en geheugen vragen. In deze masterproef wordt er onderzocht of de computer kan weggelaten worden en de afbeelding meteen door het mobiel apparaat geanaliseerd kan worden. Dus er moet onderzocht worden hoe een bestaand model aangepast kan worden om efficiënt te werken op een mobiel apparaat. Hierbij moet vooral rekening gehouden worden met de rekenkracht en geheugen van het mobiele apparaat.

1.2 Probleemstelling

Mobiele apparaten zijn kleine toestellen met beperkt geheugen en beperkte rekenkracht. In deze masterproef wordt er onderzocht hoe het rekenwerk beperkt kan worden zodat het resultaat real-time geleverd kan worden. Er gaat ook onderzocht worden hoe alle data efficiënt kan worden opgeslagen op het toestel.

1.3 Doelstellingen

Het uiteindelijke doel van deze masterproef is er voor zorgen dat een bestaand deep learning model aangepast kan worden zodat dit real-time resultaten kan geven op een mobiel apparaat. Dit gebeurt aan de hand van de volgende stappen:

- grondig begrijpen van een deep learning herkenningssysteem

- grondig begrijpen van een deep learning detectiesysteem
- Onderzoeken welke technieken er gebruikt kunnen worden om bestaande systemen op een mobiel apparaat te implementeren.
- onderzoeken voor optimalisaties voor een herkenningssysteem
- onderzoeken voor optimalisaties voor een detectiesysteem
- gevonden technieken testen en analiseren
- werkend prototype applicatie ontwerpen voor een mobiel apparaat

Hoofdstuk 2

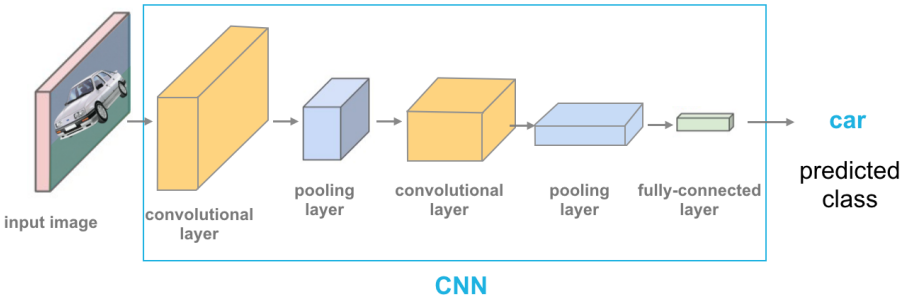
Herkenning en Detectie Algemeen

2.1 Deep learning-gebaseerde herkeningssystemen

Herkeningssystemen systemen voorspellen wat de klasse van een object is in een afbeelding. Dus het herkennen van objecten in digitale afbeeldingen zonder deze te lokaliseren of aan te duiden. Bij herkeningssystemen is er geen of weinig overlap tussen de trainingsafbeeldingen en de inputafbeeldingen. Bijvoorbeeld bij een gezichts herkenningssysteem wordt er een algemeen herkeningssystemen ontworpen dat gezichten herkent, en niet elk individueel gezicht herkent. Voor een herkenningssysteem is er een goed getraind netwerk nodig dat input afbeeldingen omzet in features. Er moet een database zijn met daarin de gegevens van de objecten die men wilt herkennen. Vervolgens hebben is er ook een methode nodig om features van het neuraal netwerk te vergelijken met de gegevens in de database om het juiste object te herkennen.

2.1.1 Herkenning

Eens dat er een getraind CNN is kunnen we verdergaan met de effectieve herkenning. Als men bepaald objecten in een afbeelding wil ontdekken gaat men met behulp van het CNN de afbeelding omzetten in een embedding. Embeddings Koehrsen (2018) zijn vector representaties die kunnen worden vergeleken in een embedding space, waar gelijkaardige objecten dichter bij elkaar liggen. De embedding van de input afbeelding wordt vergeleken met de embeddings die zich in een gallerij bevinden. De gallerij is een database/verzameling met gekende embeddings/ID's van de objecten die men wilt herkennen. Met behulp van een query kunnen we gelijkaardige objecten uit de gallerij halen om deze te gaan vergelijken in een embedding space. Een query is een embedding van de input waarvan het label niet gekend is. Gelijkaardige embeddings kunnen gezocht worden via de nearest neighbour techniek, waar we naar de klasse van de dichtsbijzijnde buur gaan kijken.

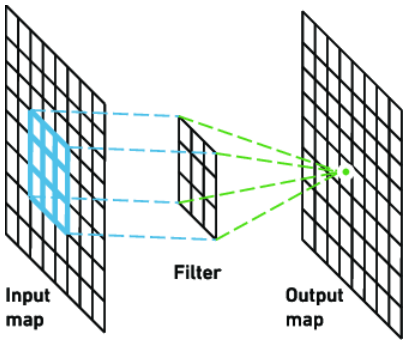


Figuur 2.1: CNN met 2 convolutie lagen en 2 pooling lagen en een fully-connected layer

2.1.2 convolutioneel neuraal netwerk (CNN)

De belankrijkste bouwsteen van een herkenningssysteem is een goed getrainde CNN beschreven door Jiang et al. (2019). Het algemeen model van een CNN is weergegeven in figuur 2.1. In tegenstelling tot fully connected netwerken wordt bij een CNN de gewichten gedeeld over verschillende locaties om zo het aantal parameters te verminderen.

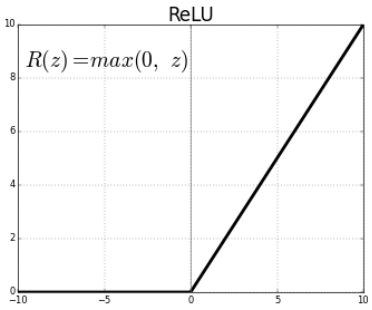
Het belangrijkste deel van een CNN zijn de convolutielagen (figuur 2.2) waarbij men een kernel/filter over de input laat gaan wat als output een feature map genereerd. Een kernel bestaat uit set van gewichten die met de input worden vermenigvuldigd, deze kernel wordt over de input afbeelding geschoven. Al de pixels binnen het veld van de kernel worden gereduceert tot een enkele waarde. CNN leren verschillende features met verschillende kernels in parallel. Waardoor de matrices met feature mappen steeds kleiner worden maar ook dieper worden. Een andere factor van een convolutie laag is de stride, deze waarde geeft aan met hoeveel pixels de kernel telkens moet doorschuiven. Een CNN bestaat uit een opeenvolging van een aantal convolutie lagen die steeds meer high-level features extraheren. Hoe meer convolutielagen een netwerk telt hoe meer features er uit de input worden gehaald, maar hoe trager het netwerk is.



Figuur 2.2: Convolutie laag waarbij een filter wordt herleid tot een output feature.

Elke convolutie laag wordt gevolgd door een niet-lineare activatie functie , de meest gebruikt functie hiervoor is de rectified linear unit (ReLu) (figuur 2.3). De ReLu wordt vaak gebruikt omdat deze eenvoudig is, kan exact 0 weergeven en ziet er lineair uit. Max(0,x) is de ReLu bewerking, dus er wordt verdergegaan met 0 of de input waarde. Zonder een niet-lineare activatie functie kan het

CNN herleid worden tot 1 convolultie laag die geen high-level features kan extraheren. Andere mogelijkheden voor Lineare activatie functies zijn: Sigmoid en Tangens hyperbolicus maar deze functies vragen meer rekenwerk.



Figuur 2.3: ReLu, waarbij het maximum wordt genomen van 0 en de input waarde.

Een volgende bouwsteen is de pooling laag waarbij het aantal samples in de feature map wordt verlaagt. De meest voorkomende methode is max-pooling waarbij er verder wordt gegaan met de maximum waarde in een bepaalde regio. Het doel van een pooling laag is om het aantal parameters te verminderen en zo ook het rekenwerk te verminderen. Er kan ook gebruik gemaakt worden van avarage pooling waarbij er verder wordt gegaan met de gemiddelde waarde van een regio. Er is ook minimal pooling waarbij er verder wordt gegaan met de minimum waarde.

Op het einde van elk CNN volgen er meestal 1 of meerdere fully connected lagen. Deze lagen connecteren elke input van één laag met elke activatie eenheid van de volgende laag. Dit zorgt voor meer parameters en meer rekenwerk waardoor deze lagen een vertragende factor vormen. De fully connected lagen gaan niet-lineare combinaties leren van de features van de convolutie lagen. De fully connected lagen zorgen voor een classificatie op basis van de features van de convolutie lagen.

2.1.3 Trainen van een CNN

Het trainen van een CNN bestaat uit het leveren van veel voorbeelden aan het netwerk. Op basis van het resultaat van deze voorbeelden worden telkens de gewichten van de kernels aangepast, zodat er steeds een beter resultaat wordt geleverd.

De loss functie geeft de error van de voorspelling weer tijdens het trainen van een neuraal netwerk. Op basis van de loss functie gaat men via de stochastic gradiënt decent de gewichten van het netwerken bijstellen zodat bij een volgende trainingsinput de loss functie een beter resultaat geeft. Bij de stochastic gradiënt decent wordt er per batch/group trainingsvoorbeelden de gewichten bijgesteld. De gradienten worden berekend door de loss af te leiden naar de gewichten via de ketting regel, en de gewichten worden bijgesteld volgens de tegengestelde gradient.

De learning rate bij het trainen van een CNN beïnvloed de grootte van de stap waarmee de gewichten worden bijgesteld. Hoe kleiner de learning rate hoe langer het trainen van een CNN duurt. Maar als de learning rate te hoog is kan het resultaat een slecht getraind netwerk zijn, omdat de

veranderingen op de gewichten te groot is om een beter resultaat te verkrijgen.

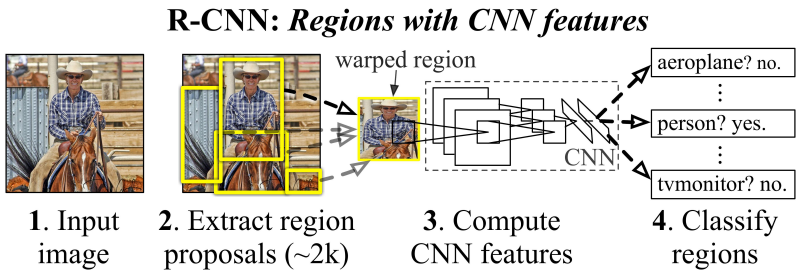
2.2 Deep learning-gebaseerde detector

Object detectie is het lokaliseren en classificeren van objecten in een afbeelding, waarbij de objecten aangeduid worden met een Bounding box. Door gebruik te maken van CNN kunnen er vrij nauwkeurige object detectoren ontworpen worden. Object detectie maakt voornamelijk gebruik van twee methodes: de single-stage detector en de two-stage detector.

2.2.1 Two-stage detector

Zoals de naam zegt bestaat deze methode uit 2 niveaus. Het eerste deel worden er Regions of Intrest (RoIs) gecreëerd, dit is het filteren van regio's waarbij de kans groot is dat deze een object bevatten. Het tweede deel classificeert en verfijnt de lokalisatie van de RoIs die in het eerste deel gecreëerd werden. Dit gebeurt door elk van de RoIs door een CNN te voeren. Region-based Convolutional Neural Network (R-CNN) Girshick (2015) is het basis principe van de two-stage detectoren weergegeven in figuur 2.4. Hierbij wordt met een region proposal algoritme regio's uit de afbeelding gefilterd waar de kans groot is dat er objecten op staan. R-CNN bestaat uit 3 stappen:

1. Via een selective search algoritme Uijlings et al. (2013) worden er ongeveer 2000 mogelijke regio's met objecten geselecteerd.
2. Elke mogelijke regio wordt omgezet naar een feature vector via een CNN.
3. Elke regio wordt vervolgens geclassificeerd met een klas-specifieke SVMs.

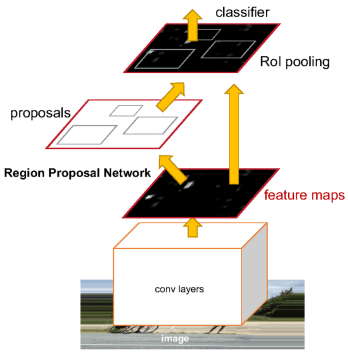


Figuur 2.4: R-CNN

R-CNN is een trage detector vermits elke RoIs door een CNN moet gaan. Deze methode is geëvolueerd tot de veel snellere methode Faster R-CNN Ren et al. (2016). Hierbij wordt de afbeelding door een CNN behandelt en vervolgens maakt men gebruik van een Region Proposal Network (RPN) dit wordt weergegeven in figuur 2.5. Het RPN gaat zoals bij R-CNN regio's uit de afbeelding filteren waar de kans groot is dat er objecten opstaan. Maar het RPN werkt sneller en levert betere resultaten dan het region proposal algoritme.

Het RPN is een deep fully convolutional network dat per input een set van regio's als output geeft. Elk van deze regio's heeft een objectness score wat een maat is voor het object t.o.v. de achtergrond in de afbeelding. Om een region proposal te genereren wordt het RPN over de feature map geschoven die gegenereerd is door het voorgaande CNN. Op elke sliding window locatie worden er meerdere regio voorspellingen gedaan. Deze voorspelling wordt gedaan door verschillende anchor boxen in de sliding window te evalueren.

Vervolgens worden RoIs omgezet naar een feature vector met vaste lengte door RoI pooling. Elk van deze features gaat door een set van fully-connected lagen die 2 lagen als output heeft. een softmax laag die de klasse voorspelt, en een bounding box regressie laag die de bounding box voorspelt.



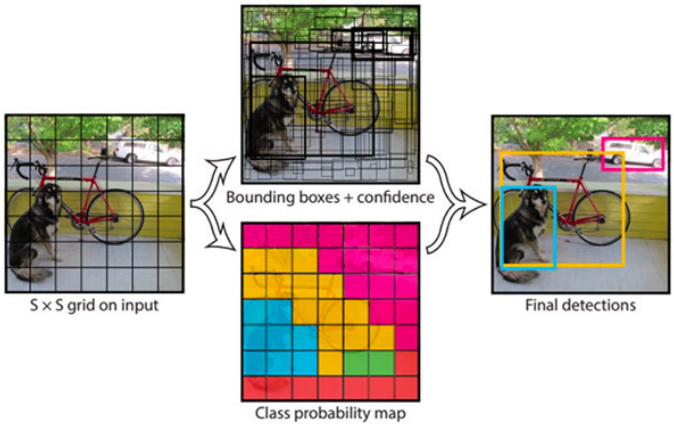
Figuur 2.5: Faster R-CNN

2.2.2 One-stage detector

Bij one-stage detectoren gebeurt object detectie in één keer met één neuraal netwerk. Dus er is geen region proposal niveau meer zoals bij de two-stage detector. Deze detector gebruiken minder geheugen en rekenkracht t.o.v. two-stage detectoren. One-stage detectoren zijn sneller dan two-stage detectoren omdat ze alles in één keer doen, maar kunnen wat in nauwkeurigheid verliezen t.o.v. two-stage detectoren. De twee meest gebruikte technieken van one-stage detectie zijn: You Only Look Once (YOLO) en Single Shot Detection (SSD).

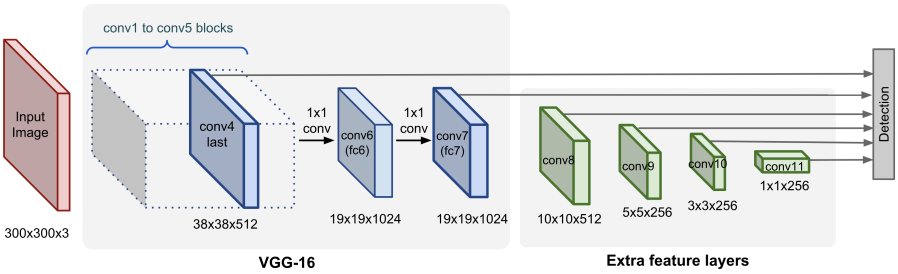
YOLO Redmon et al. (2016) verdeelt de afbeelding in een S x S rooster zoals in figuur 2.6 te zien is. De cel waarin het middelpunt van het object valt is verantwoordelijk voor de object detectie. Elke cel voorspelt K aantal bounding boxes en een score die aangeeft hoe zeker het model is dat een bepaalde bounding box een object bevat. deze score wordt bepaalt met de Intersection Of Union (IOU) tussen de voorspelde box en de ground truth box. Vervolgens is er nog een methode nodig voor de overbodige bounding boxen te verwijderen. Een eerste mogelijkheid is door enkel bounding boxen te tekenen waarvan de voorspelling boven een treshold ligt. Een andere methode is non-maxima supression deze methode zorgt ervoor dat elk object maar één bounding box heeft. Deze techniek houdt enkel de bounding box over met de beste voorspelling en onderdrukt de rest van de bounding boxen.

SSD Liu et al. (2016) is een one-stage detector (figuur 2.7) waarbij een afbeelding door verschil-



Figuur 2.6: YOLO waarbij de input is opgedeeld in een S x S rooster. En waarbij bounding box voorspellingen zijn gedaan.

lende convolutielagen gaat, wat als resultaat feature mappen op verschillende schalen oplevert. Op elke locatie van deze feature mappen van verschillende schalen wordt een vaste set van bounding boxen geëvalueerd. Voor elk van deze boxen wordt de zekerheid dat het een object bevat voorspelt. Op het einde wordt non maximum suppression gebruikt om de finale voorspelling te maken. Het netwerk van een SSD bestaat uit een basis netwerk dat gevormd wordt door een standaard classificatie netwerk zonder de fully-connected lagen. Vervolgens worden er extra convolutie lagen toegevoegd, wat het model toelaat om voorspellingen te doen op verschillende schalen.



Figuur 2.7: One-stage detector met VGG net backbone

Hoofdstuk 3

Herkenning en detectie implementatie op mobiel platform

Dit hoofdstuk zal gaan over het implementeren van deep learning herkeningssystemen en detectiesystemen op een mobiel platform. Hierbij zal het trainen van het CNN model nog steeds gebeuren op een computer. Bij het uitvoeren van een neuraal netwerk op een mobiel apparaat zal moet men rekening houden met de volgende zaken: gelimiteerde rekenkracht en beschikbaar geheugen. Ook moet er rekening gehouden worden met een beperkte batterij, want CNN's gebruiken veel bandbreedte en voeren veel berekeningen uit wat meer energie verbruikt. In dit hoofdstuk zullen we een aantal state-of-the-art CNN architecturen voor mobiele platformen bespreken.

3.1 Implementatie op mobiele platformen

Voor de implementatie van neurale netwerken op een mobiel platform kan er gebruik gemaakt worden van een framework. Deze frameworken zijn een set van tools die de programeur helpen om een machine learning model te implementeren op een mobiel toestel. De twee meest gebruikte frameworks voor mobiele implementaties zijn: TenserflowLite en Pytorch mobiel.

Tenserflow is ontworpen door Google en is een open source library voor machine learning implementaties. Door de introductie van de Keras API is Tenserflow gebruiksvriendelijk geworden. Zo heeft Tenserflow een gebruiksvriendelijk API voor eenvoudige projecten en meer uitgebreide tools voor complexe projecten. Zo is het met Tenserflow mogelijk om overzichtelijke en elegante code te schrijven. Door gebruik te maken van TensorBoard kan data op een flexibele manier gevisualiseerd worden. Tenserflow biedt ook betere ondersteuning op gebied van deployment van een productie model.

Pytorch is ontworpen door facebook en wordt zoals Tenserflow ook gebruikt voor machine learning implementaties. Dit is een python gebaseerd framework dat focust op flexibiliteit, maar deze extra flexibiliteit zorgt voor meer lijnen code. Door zijn flexibiliteit is het gemakkelijk om nieuwe functionaliteiten toe te voegen door bestaande code aan te passen of nieuwe code toe te voegen. Pytorch

maakt gebruik van externe tools zoals TensorBoard om data te visualiseren.

3.2 CNN architecturen voor mobiele platformen

3.2.1 Mobilenet

3.2.2 EfficientNet

3.2.3 TinyYOLO

3.3 Optimalisaties van neurale netwerken voor mobiele platformen

In deze paragraaf wordt er onderzocht welke optimalisaties er kunnen worden toegepast om de accuraatheid, snelheid en gebruikt geheugen te verbeteren. Maar het optimaliseren van een bepaalde factor zal vaak negatieve gevolgen hebben voor een andere factor, dit zal meestal de accuraatheid zijn. Dus er zal een goede belans gevonden moeten worden tussen de optimalisatie en de negatieve gevolgen op de andere factoren.

3.3.1 Pruning

Pruning is de eerste stap van de Deep comression methode voorgesteld door Han et al. (2016). Bij het trainen van een CNN hebben bepaalde gewichten een grotere invloed op het resultaat. Andere gewichten hebben weinig tot geen invloed op het resultaat. Maar alle gewichten worden steeds berekend ongeacht hun invloed op het resultaat. Bij pruning worden de gewichten met een kleine invloed op het resultaat verwijderd. Waardoor er geen berekeningen meer moeten uitgevoerd worden voor de verwijderde gewichten. Eerst wordt het CNN op een normale manier getraind zonder pruning. Vervolgens worden al de kleine gewichten onder een bepaalde treshold verwijderd. volgens Han et al. (2016) wordt voor VGG-16 het aantal parameters met factor 13 verminderd, voor AlexNet met een factor 9. Deze methode heeft zeer weinig tot geen effect op de accuraatheid.

3.3.2 Parameter quantisatie

Het quantiseren en delen van gewichten is een tweede stap van de deep compression methode. Een CNN bestaat uit miljoenen gewichten, en de waarde van elke van deze gewichten moeten op het systeem worden opgeslagen. De default representatie van een waarde wordt opgeslagen als een floating point nummer wat 4 bytes in beslag neemt. Dus voor mijoenen parameters hebben de gewichten veel schijfruimte nodig. Een mogelijke oplossing hiervoor is quantiseren van gewichten, waarbij de getal representatie van de gewichten wordt verandert naar fixed point. Hierbij worden het waarden van gewichten beperkt tot een set van beschikbare waardes. Waarbij de waardes ëënmalig worden opgeslagen en al de gewichten refereren naar een waarde van de vaste set met

waardes. Hoe kleiner de set met waardes is hoe minder geheugen er in beslag wordt genomen, maar een kleinere set van waardes zorgt ook voor een mindere accuratie. Dus de grootte van de set moet goed worden gekozen zodat er niet te veel geheugen wordt gebruikt met een accepteerbare daling in accuratie. Han et al. (2016) past vervolgens Huffman encoding toe die een compressie uitvoerd op de gequantiseerde parameters.

3.3.3 Convolutionele filter compresies

een andere methode voorgesteld is Compressed Convolutional Filters. Hierbij wordt de grootte van de kernel verkleind om het aantal parameters en rekenwerk te verminderen. Maar door de kernels te verkleinen daalt de accuraatheid van het CNN.

3.3.4 Matrix factorisatie

3.3.5 Vermijd fully connected lagen

Fully connected lagen zijn een basis component van neurale netwerken. Maar fully connected lagen genereren veel parameters, dus gebruiken ook veel geheugen. Ook voeren fully connected lagen veel berekening uit waardoor zij ook een vertragende factor zijn. Dus voor mobiele implementaties is het beter om geen of weinig fully connected lagen te gebruiken.

3.3.6 Wijzigen van de Kernel

Door met meer kernels te werken kan men meer informatie uit de data halen, maar dan worden er ook meer feature mappen gegenereerd. Deze feature mappen beschikken over veel informatie maar nemen meer geheugen in beslag. Een groter aantal feature mappen is meer data dus ook meer berekeningen en meer berekeningen maakt het systeem trager. Dus door het aantal kernels te verminderen worden er ook minder feature mappen gegenereerd. Dit zorgt voor een snelheids winst en meer vrij geheugen, maar er is dan wel een verlies aan informatie.

De grootte van de kernel kan ook woorden aangepast zo kan men i.p.v. een 3x3 kernel met een 2x2 kernel werken. Door de kernelgrootte te verkleinen moeten er minder berekeningen worden uitgevoerd wat zorgt voor een snelheidswinst en meer vrij geheugen. Maar de door de kernelgrootte te verkleinen is er terug een verlies aan informatie.

3.3.7 Pooling laag optimalisatie

De pooling laag zorgt voor een vermindering in de dimensie van de feature map. deze vermindering van dimensie zorgt ervoor dat er minder parameters zijn, maar minder parameters betekent ook verlies aan informatie. Dus door wijzigingen aan te brengen aan de pooling laag kan de hoeveelheid data en rekenwerk verbeterd worden.

De pooling laag kan naar voor worden geschoven in het CNN waardoor de dimensie van de feature map sneller kleiner wordt. Dit heeft als gevolg dat er met minder parameters verdergegaan moet worden, waardoor het model sneller wordt. Maar dit zorgt er ook voor dat bepaalde informatie sneller verloren gaat wat zorgt voor een lagere accuratie. Een andere vorm van pooling optimalisatie is gewoonweg meer pooling lagen toevoegen waardoor de dimensie van de feature mappen vaker verkleint wordt. Maar dit heeft ook een negatieve invloed op de accuraatheid.

Hoofdstuk 4

Specificaties

4.1 Algemene richtlijnen

Hoofdstuk 5

Richtlijnen voor formules

Hoofdstuk 6

Richtlijnen voor referenties

Bibliografie

Girshick, R. (2015). Fast R-CNN. *arXiv:1504.08083 [cs]*. arXiv: 1504.08083.

Han, S., Mao, H., and Dally, W. J. (2016). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv:1510.00149 [cs]*. arXiv: 1510.00149.

Jiang, X., Hadid, A., Pang, Y., Granger, E., and Feng, X. (2019). *Deep Learning in Object Detection and Recognition, edited by Xiaoyue Jiang, Abdenour Hadid, Yanwei Pang, Eric Granger, Xiaoyi Feng.* Springer Singapore : Imprint: Springer, Singapore, 1st ed. 2019. edition.

Koehrsen, W. (2018). Neural Network Embeddings Explained.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. (2016). SSD: Single Shot MultiBox Detector. volume 9905, pages 21–37.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788. ISSN: 1063-6919.

Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497 [cs]*. arXiv: 1506.01497.

Uijlings, J. R., R, van de Sande, K. E., A, Gevers, T., Smeulders, A. W., and M (2013). Selective Search for Object Recognition. *International Journal of Computer Vision*, 104(2):154–171. Num Pages: 154-171 Place: New York, Netherlands Publisher: Springer Nature B.V.

Bijlage A

Uitleg over de appendices

Bijlagen worden bij voorkeur enkel elektronisch ter beschikking gesteld. Indien essentieel kunnen in overleg met de promotor bijlagen in de scriptie opgenomen worden of als apart boekdeel voorzien worden.

Er wordt wel steeds een lijst met vermelding van alle bijlagen opgenomen in de scriptie. Bijlagen worden genummerd het een drukletter A, B, C,...

Voorbeelden van bijlagen:

Bijlage A: Detailtekeningen van de proefopstelling

Bijlage B: Meetgegevens (op USB)

FACULTEIT INDUSTRIËLE INGENIEURSWETENSCHAPPEN
CAMPUS DE NAYER SINT-KATELIJNE-WAVER
J. De Nayerlaan 5
2860 SINT-KATELIJNE-WAVER, België
tel. + 32 15 31 69 44
iiw.denayer@kuleuven.be
www.iiw.kuleuven.be