Max Pooling vs Average Pooling

Inhoudsopgave:

Inleiding

Bij CNN's is Pooling een essentieel proces die meestal wordt gebruikt om de dimensionaliteit van de input gegevens te verkleinen, en daarmee de computionele efficientie te verbeteren. In de categorie van Pooling zijn Max Pooling en Average Pooling de meest gebruikte technieken. Hoewel beide poolingstechnieken worden gebruikt om de input te verkleinen hebben ze verschillende effecten op de prestatie van het Object Detection model. In dit onderzoek wil ik het verschil tussen Max Pooling en Average Pooling èn hun effecten op de prestaties van een Object Detection model onderzoeken.

Hoofdvraag

Hoe beïnvloedt de keuze tussen Max Pooling en Average Pooling de prestaties van Object Detection-modellen en speelt de achtergrondkleur van de afbeelding een rol?

Mobilenet V1

Mobilenet V1 is een CNN architectuur. Wat mobilenet anders maakt dan andere CNN architecturen is dat er gebruik wordt gemaakt van depthwise separable convolutions in plaats van standaard convoluties die vaak bij andere architecturen gebruikt worden. Depthwise separable convolutions zijn een speciaal type convolutielaag die in CNN's worden gebruikt om de verwerking van featuremaps efficiënter te maken. In tegenstelling tot standaardconvoluties waarbij een enkele kernel wordt gebruikt om alle featuremaps tegelijkertijd te convoleren. Maakt een depthwise seperable convolution gebruik van twee opeenvolgende lagen. De eerste laag is een depthwise convolution op elke inviduele featuremap. De tweede laag is een pointwise convolution om de featuremaps te combineren. Hierdoor is het mogelijk om lichtere modellen te bouwen. Mobilenet V1 is ontworpen om alleen afbeeldingen te verwerken met de dimensies van 224 x 224 x 3. Ook heeft Mobilenet V1 globale hyperparameters, namelijk een breedtevermenigvuldiger en een resolutievermenigvuldiger. Hiermee kan je de snelheid en de grootte van het model aanpassen zonder onnodig precisie te verliezen. De architectuur bestaat uit 15 standaard convoluties en 15 depth seperatable convoluties die elkaar afwisselen. Daarnaast zijn er ook Average Pooling, Fully Connected en Softmax layers.

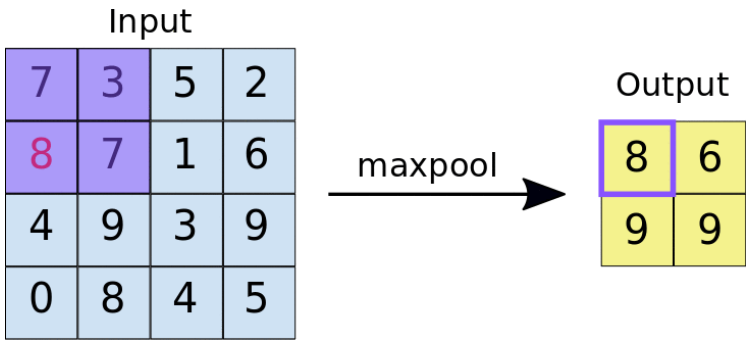
TYPE	STRIDE	KERNEL SHAPE	INPUT SIZE
Conv	2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw	1	3 × 3 × 32	112 × 112 × 32
Conv	1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw	2	3 × 3 × 64	112 × 112 × 64
Conv	1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw	1	3 × 3 × 128	56 × 56 × 128
Conv	1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw	2	3 × 3 × 128	56 × 56 × 128
Conv	1	1 × 1 × 128 × 256	56 × 56 × 128

Conv dw	1	3 × 3 × 256	28 × 28 × 256
Conv	1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw	2	3 × 3 × 256	28 × 28 × 256
Conv	1	1 × 1 × 256 × 512	14 × 14 × 256
Conv dw	1	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw	1	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw	1	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw	1	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw	1	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw	2	3 × 3 × 512	14 × 14 × 512
Conv	1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw	2	3 × 3 × 1024	7 × 7 × 1024
Conv	1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool	1	Pool 7 × 7	7 × 7 × 1024
Fully Connected	1	1024 × 1000	1 × 1 × 1024
Softmax	1	Classifier	1 × 1 × 1000

Pooling

Afbeeldingen bevatten erg veel features. Dit is een probleem wanneer die afbeeldingen als input voor een CNN worden gebruikt. Als er teveel features zijn ontstaat er namelijk overfitting waardoor de accuraatheid van de CNN verminderd wordt. Om dit probleem op te lossen kunnen we pooling toepassen om de features te generaliseren. Hierdoor heeft de CNN een verminderde berekeningstijd.

Pooling werkt als volgt: een tweedimensionaal rechthoekig gebied glijdt over een afbeelding en kiest daaruit een element, bij bijvoorbeeld Max Pooling pakt hij dan het grootste element in het rechthoekig gebied.



Voorbeeld pooling

Het is mogelijk om een eigen pool size mee te geven, dat is dan de grootte van de rechthoek waarmee hij over het plaatje glijdt. Er zijn verschillende Pooling layers om uit te kiezen die verschillende elementen pakken, en dus ook verschillende resultaten geven. Ook is het mogelijk om ze te combineren

Max Pooling

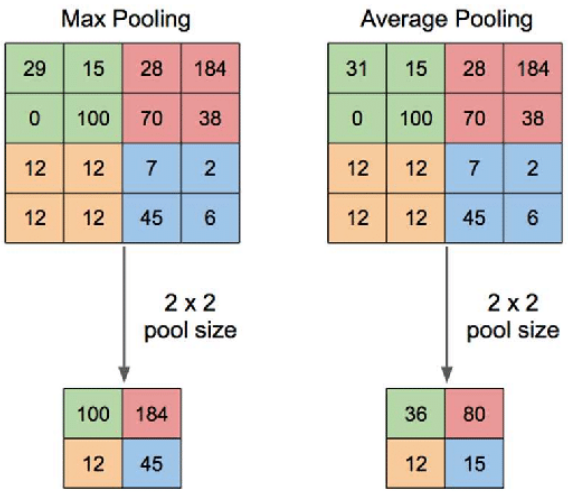
Max Pooling is een manier van poolen die de maximale element in een zelf aangegeven regio selecteerd. De output van een Max Pooling layer is dan de afbeelding met de belangrijkste features van de afbeelding.

```
tf.keras.layers.MaxPooling2D(
pool_size=(2, 2), strides=None, padding="valid", data_format=None, **kwargs
```

Average Pooling

Average Pooling is een manier van poolen die het gemiddelde van de elementen in een rechthoekig gebied berekend. De output van een Average Pooling layer is dan een afbeelding met een gemiddelde van features.

```
tf.keras.layers.AveragePooling2D(
    pool_size=(2, 2), strides=None, padding="valid", data_format=None, **kwargs)
```



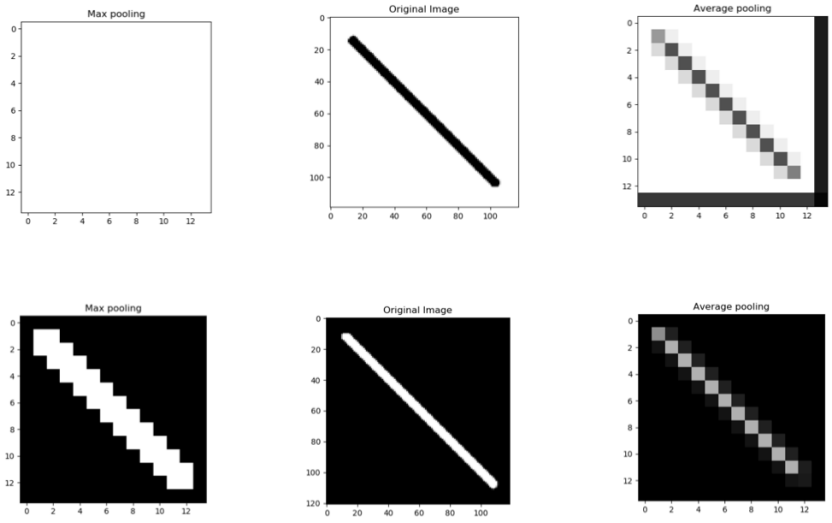
Verschil output MaxPooling en Averagepooling

Vergelijking soorten Pooling

Max Pooling en Average Pooling hebben verschillende sterke en zwakke punten, of ze je CNN beter maken kan ook afhangen van welke dataset je gebruikt en welke taak je wil uitvoeren.

Max Pooling is beter in het behouden van belangrijke features en kan ook translationele invariantie veroorzaken, wat betekend dat de exacte positie van het object in je input minder belangrijk is. Waardoor het ook niet uitmaakt of een afbeelding bijvoorbeeld geroteerd of verschoven is. Ook is het handig voor classificatietaken, en is het vaak sneller dan Average Pooling aangezien een max vinden een snellere operatie is dan een gemiddelde berekenen. Maar helaas kan het ook gebeuren dat hij kleine belangrijke details weglaat.

Average Pooling is beter in het behouden van de algehele structuur van de input en verminderd ook ruis. Ook helpt het met overfitting verminderen en als de input veel ruis of variatie bevat.



Hierboven is goed te zien dat de voor- en achtergrond een erg groot verschil kunnen maken bij het resultaat van de twee poolingmethodes. Bij Max Pooling met een lichte achtergrond en een donkere voorgrond valt de lijn zelfs helemaal weg. Maar bij een donkere achtergrond doet Max Pooling het juist super goed. De pool size kan ook een drastisch verschil maken.

Verschil Max Pooling en Average Pooling op een afbeelding

Code

Voor de volledige code kan worden gekeken naar: https://github.com/zoevdpol/vision_eindopdracht

Eerst schreef ik een functie die csv's uitleest zodat ik dat kon doen voor de train en test data.

```
def CsvReader():
    #csv train data laden
    TrainCsv = pd.read_csv(
        "train_labels.csv",
        names=["filename", "width", "height", "class", "xmin",
            "ymin", "xmax", "ymax"])
    #csv test data laden
    TestCsv = pd.read_csv(
        "test_labels.csv",
        names=["filename", "width", "height", "class", "xmin",
            "ymin", "xmax", "ymax"])
    return TrainCsv, TestCsv
```

Hierna heb ik een functie gemaakt die de train foto’s cropped op basis van de meegegeven boundingboxes en daarna alle fotos 224 x 224 maakt aangezien het Mobilenet model die ik gebruik deze imagesize moet hebben. Dit heb ik gedaan door borders toe te voegen aan de zijkant of bovenkant en onderkant, zodat de afbeeldingen naar 224 x 244 geresized konden worden zonder de foto’s te warpen / stretchen. Om dat te doen moest ik wel eerst kijken als de foto langer was in de breedte of in de hoogte, of als de afbeeldingen al vierkant was. Ik heb ook een soortgelijke functie gemaakt voor de traindata die alleen de afbeeldingen resized naar 224 x 224 met de borders.

```
def PreprocessImage(ImagesList, TrainCoordList):
    kleinefotos = []
    borderarrayx = []
    borderarrayy = []
    #borderarrays maken
    for i in range(0, 224):
        borderarrayx.append(np.array([0, 0, 0]))
    borderarrayx = np.array(borderarrayx)
    #alle foto's de grootte maken van hun boundingbox
    for i in range(0, len(ImagesList)):
        xmin = int(TrainCoordList[i][0])
        ymin = int(TrainCoordList[i][1])
        xmax = int(TrainCoordList[i][2])
        ymax = int(TrainCoordList[i][3])
        height = ymax - ymin
        width = xmax - xmin
        #als de foto groter is in de hoogte dan in de breedte
        if(height > width):
            x = int((width/height) * 224)
            y = 224
            borderamount = int((224 - x)/2)
            croppedimage = ImagesList[i][ymin:ymax, xmin:xmax]
            croppedimage = resize(croppedimage, (y, x, 3))
            foto = croppedimage
            borderarrayy = []
            for s in range(0, borderamount):
                borderarrayy.append(np.array([0, 0, 0]))
            borderarrayy = np.array(borderarrayy)
            foto = foto.tolist()
            for j in range(0, 224):
                for item in borderarrayy:
                    foto[j].insert(0, item)
                    foto[j].append(item)
                if (len(foto[j])==223):
                    foto[j].append(borderarrayy[0])
                foto[j] = np.array(foto[j])
            foto = np.array(foto)
            kleinefotos.append(foto)
        #als de foto groter is in de breedte dan de hoogte
        elif (width != height):
            x = 224
            y = int((height/width) * 224
            borderamount = int((224 - y) /2)
            croppedimage = ImagesList[i][ymin:ymax, xmin:xmax]
            croppedimage = resize(croppedimage, (y, x, 3))
            foto = croppedimage
            foto = foto.tolist()
            for j in range(0, int(borderamount)):
```

```
                foto.insert(0, borderarrayx)
                foto.append(borderarrayx)
                if(len(foto) == 223):
                    foto.append(borderarrayx)
            foto = np.array(foto)
            kleinefotos.append(foto)
        #als de foto al vierkant is
        else:
            #als de foto al vierkant is kan hij direct geresized worden
            croppedimage = ImagesList[i][ymin:ymax, xmin:xmax]
            croppedimage = resize(croppedimage, (224, 224, 3))
            foto = np.array(croppedimage)
            kleinefotos.append(foto)
    kleinefotos = np.array(kleinefotos)
    kleinefotos[5][0]
    return kleinefotos
```

Hierna heb ik de uitgelezen csv's omgezet naar lists. TrainFilelist is een list die alle filenames bevat en TrainCoordList is een list die alle Coordinaten bevat. Vanuit TrainFileList heb ik ImagesList gemaakt, in ImagesList staan alle afbeeldingen in vorm van arrays. ImagesList doe ik dan in PreprocessImages zodat hij de afbeeldingen cropped op basis van de meegegeven boundingboxes en de afbeelding daarna geresized naar 224 x 224 x 3 met behulp van borders zodat de afbeeldingen niet gewarpt worden. ImagesList is dan mijn traindata met TrainCoordList en als mijn Labels. Ik heb een Mobilenet model gebruikt. hier wordt uitgelegd wat de input hiervan is en wat de Mobilenet architectuur is. Zonder de GlobalAveragePooling klopt de shape niet. Tijdens mijn proef voeg ik dan MaxPooling en AveragePooling toe om te kijken wat voor verschil dit maakt.

```
model = MobileNet(input_shape=(224,224,3), include_top=False, alpha = 1.0)
for layers in model.layers:
    layers.trainable = False
x = model.layers[-1].output
x = Conv2D(4, kernel_size=4, name="coords")(x)
x = GlobalAveragePooling2D()(x)
x = Reshape((4,))(x)
model = Model(inputs=model.inputs, outputs=x)
model.summary()
model.compile(optimizer="Adam", loss = "mse", metrics = [IoU])
stop = EarlyStopping(monitor='IoU', patience=10, mode="max" )
reduce_lr = ReduceLROnPlateau(monitor='IoU',factor=0.2,patience=10, min_lr=1e-7, verbose=1, mode="max" )
model.fit(nieuwe_input, TrainCoordList, epochs=200,callbacks=[stop,reduce_lr], verbose = 2)
```

Hierna test ik de CNN uit door een test afbeelding in een predict functie te zetten en laat ik de afbeelding met de boundingboxes zien.

Onderzoeksmethode

Om het verschil te zien van de prestaties bij Max Pooling en Average Pooling ga ik het testen. Dit ga ik doen door Max Pooling en Average Pooling toe te passen bij drie verschillende afbeeldingen.

Test data:

Er is een wasbeer uitgezocht die ik daarna op verschillende achtergronden heb geplaatst. Dit is gedaan om de theorie van hierboven te testen, waarin wordt gezegd dat de kleur van de achtergrond veel verschil maakt bij de Poolingmethodes. De eerste afbeelding is een een wasbeer met een zwarte achtergrond. De tweede afbeelding is dezelfde afbeelding van een wasbeer maar dan met een witte achtergrond. De derde afbeelding is een afbeelding van een wasbeer met een gemengde achtergrond. Op elke afbeelding ga ik drie keer Max Pooling en drie keer Average Pooling toepassen, dit doe ik om toeval te vermijden.

De drie zelfgemaakte test afbeeldingen

Proefbeschrijving

Bij het testen gebruiken ik Tensorflow Keras voor de Max Pooling functie en Average Pooling functie. Ook gebruik ik een Mobilenet model waar al layers inzitten. Bij het onderzoek doe ik per test twee keer een Max Pooling of Average Pooling, beide met een pool size van 2 x 2.

```
x = AveragePooling2D(pool_size = (2,2))(x)
x = AveragePooling2D(pool_size = (2,2))(x)
```

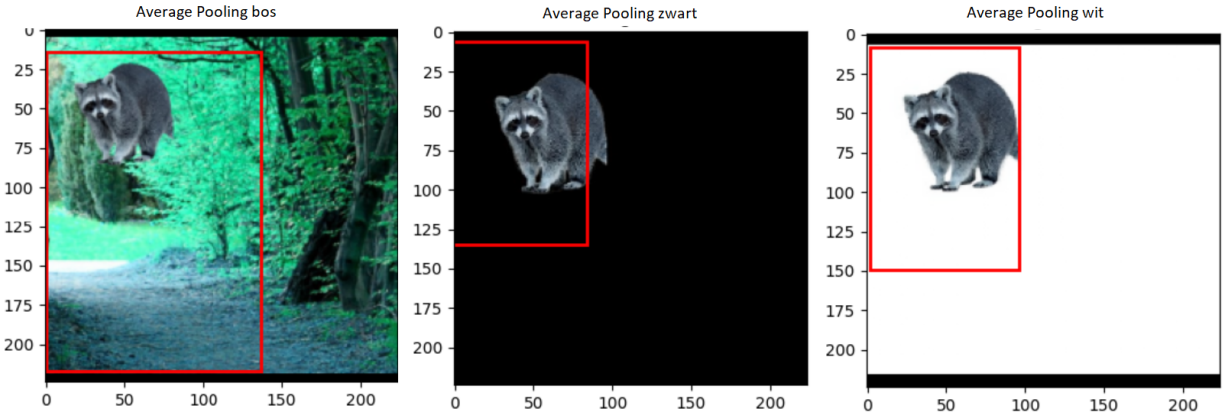
Bij elke testronde heeft het model 250 epochs.

```
stop = EarlyStopping(monitor='IoU', patience=10, mode="max" )
reduce_lr = ReduceLROnPlateau(monitor='IoU',factor=0.2,patience=10, min_lr=1e-7,
            verbose=1, mode="max" )
model.fit(nieuwe_input, TrainCoordList, epochs=250,callbacks=[stop,reduce_lr],
          verbose = 2)
```

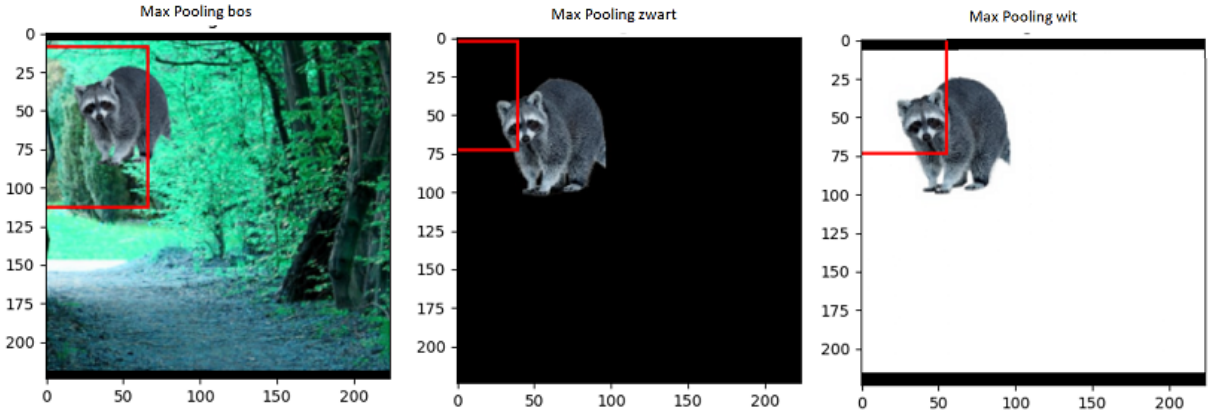
Om het resultaat te kunnen tonen laat ik de afbeelding van de wasbeer met de boundingboxes zien, ook print ik de coordinaten van de boundingboxes erbij. Zoals bij de onderzoeksmethode genoemd doe ik beide Poolingmethodes drie keer per afbeelding om toeval te verminderen.

Resultaten

Elke test is driemaal uitgevoerd en het resultaat was consistent bij elke uitvoering, daarom is er één resultaat per test geselecteerd voor verdere vergelijking.



Average Pooling resultaten



Max Pooling resultaten

Conclusie

De conclusies die hieronder staan beschreven gelden niet voor alle datasets, modellen of code. De conclusies zijn uitsluitend voor dit onderzoek. Een andere dataset, model of code kan een enorm verschil maken.

Uit de resultaten blijkt dat mijn model met Average Pooling onder de omstandigheden van mijn dataset, model en code beter de wasbeer detecteert, dan mijn model met Max Pooling. Er is ook duidelijk te zien dat bij een afbeelding met een witte achtergrond en een donkere voorgrond, Average Pooling beter werkt dan Max Pooling, zoals eerder in theorie over Pooling is beschreven. Het is opvallend dat Max Pooling niet duidelijk de randen ziet van de afbeelding en de boundingbox buiten de afbeelding blijft doortekenen, terwijl dit bij Average Pooling wel beter gaat. Ook houdt Max Pooling niet goed rekening met de gemaakte borders op de image, dit doet Average Pooling wel.

Reflectie onderzoek

De resultaten van mijn objectdetectiemodel en onderzoek kunnen worden beïnvloed door zowel de kwaliteit van mijn code als door de dataset die ik heb gebruikt. Mijn traindataset bevat slechts 179 afbeeldingen, wat in sommige gevallen weinig kan zijn, vooral als de kwaliteit van de afbeeldingen in de traindataset niet optimaal is. Om de kwaliteit van mijn traindataset te verbeteren, heb ik alle afbeeldingen bijgesneden op basis van de meegegeven bounding boxes. Vervolgens heb ik gekeken naar de hoogte-breedteverhouding van de afbeelding, en op basis daarvan borders toegevoegd aan de bovenkant+onderkant/linkerkant+rechterkant, om de afbeelding vierkant en 224 x 224 te maken. Ik hoop hiermee de prestaties van mijn objectdetectiemodel te verbeteren. Ook realiseer ik me dat mijn testdata voor mijn onderzoek niet gevarieerd was aangezien alle wasberen in de linkerbovenhoek staan in de afbeelding.

Literatuurlijst

LINK	DATUM	EVENTUELE INFO
https://www.tensorflow.org/tutorials/load_data/csv	22/02/2023	loading data out of csv file with tensorflow
https://medium.com/themlblog/splitting-csv-into-train-and-test-data-1407a063dd74	22/02/2023	
https://www.intechopen.com/chapters/69955	22/02/2023	
https://keras.io/examples/vision/object_detection_using_vision_transformer/	23/02/2023	
https://towardsdatascience.com/custom-object-detection-using-tensorflow-from-scratch-e61da2e10087	23/02/2023	basis object detetction
https://medium.com/analytics-vidhya/create-custom-object-detection-without-using-tensorflow-api-230159a58207	23/02/2023	basis object detection
MaxPool vs AvgPool (opengenus.org)	20/03/2023	uitleg over maxpooling en averagepooling en het verschil qua resultaten

Reflectie:

Hierboven staat mijn onderzoeksverslag. Hieronder staat een volledige reflectie van mijn onderzoek. Ik doe het in hetzelfde document zodat je op de woorden met een streepje eronder kan drukken en met behulp daarvan springen naar de stuk tekst waar ik naar refereer. Normaal zou ik dan dit stuk hieronder apart inleveren. Hieronder staat de reflectie op mijn code / onderzoek. Ook vertel ik over hoe ik heb gewerkt en reflecteer ik daarop in mijn werkwijze. Daarna loop ik alle beoordelingscriteria

langs er vertel ik per punt wat ik daaraan heb gedaan in de bijlage. Tot slot staat mijn planning en weekupdates die ik al ingeleverd heb

Reflectie onderzoek

De resultaten van mijn objectdetectiemodel en onderzoek kunnen worden beinvloed door zowel de kwaliteit van mijn code als de dataset die ik heb gebruikt. Mijn traindataset bevat maar 179 afbeeldingen, wat in sommige gevallen weinig kan zijn, vooral als de kwaliteit van de afbeeldingen in de traindataset niet optimaal is. Om de kwaliteit van mijn traindataset te verbeteren, heb ik alle afbeeldingen bijgesneden op basis van de meegegeven bounding boxes. Vervolgens heb ik gekeken naar de hoogte-breedteverhouding van de afbeelding en op basis daarvan borders toegevoegd aan de bovenkant+onderkant/linkerkant+rechterkant om de afbeelding vierkant en 224 x 224 te maken. Ik hoopte dat hierdoor de prestaties van mijn objectdetectiemodel zouden verbeteren. Ook realiseer ik me dat mijn testdata voor mijn onderzoek niet gevarieerd was aangezien alle wasberen in de linkerbovenhoek staan in de afbeelding.

Werkwijze + Reflectie

In deze hoofdstuk zal ik terugblikken op ervaringen tijdens de opdracht en vertellen hoe ik heb gewerkt.

Ik ben ruim optijd begonnen met onderzoek doen naar CNN’s en Object Detection omdat ik veel stress had over hoe ik de opdracht moest doen en wat er perse verwacht werd. Ook was me niet helemaal wat je wel en niet mocht gebruiken. Daarom kostte het opstarten heel veel tijd, ook moest ik meerdere keren opnieuw beginnen en verschillende methodes toepassen. Daarom vond ik het wel jammer dat er pas verder tijdens de opdracht werd verteld hoe we het moesten doen en wat we mochten gebruiken. Ook werd er in week 2 van de eindopdracht code gestuurd die we konden gebruiken, toen ik allang mijn eigen code had geschreven.

Bijlage (uitleg beoordelingscriteria)

Als er verwezen wordt naar een hoofdstuk in dit verslag (streepje onder word) is het mogelijk om op het word te drukken om direct naar dat hoofdstuk toe te gaan

De student kan vision-algoritmes impementeren ten behoeve van een geselecteerd Vision taak:

GitHub link: https://github.com/zoevdpol/vision_eindopdracht +

Code

Ik heb Object Detection met behulp van een Mobilenet model, traindata en testdata. Ook heb ik de foto’s voorbewerkt door middel van een zelf gemaakte functie.

De student kan een opdracht analyseren en op basis hiervan het benodigde werk (waaronder het implementeren) verdelen en plannen:

In hoofdstuk(ken): Planning + Weekupdates

Ik heb voordat ik ben begonnen met de eindopdracht veel onderzoek gedaan naar CNN en object detection, daardoor had ik er ook voor gezorgd dat ik een dataset koos die al boundingboxes in een CSV had staan, zodat ik daar later geen problemen mee zou hebben. Na veel onderzoek te doen heb ik een Work Breakdown Structure gemaakt, hierin kon ik duidelijk zien wat voor stappen ik moest gaan nemen en hoe lang dat ongeveer zou gaan duren. Ik vond het alleen wel jammer dat ik optijd begon en dat Marius dan vorige week met allemaal methodes komt en een hele jupyter notebook met hoe je het moest doen terwijl ik dat helemaal zelf moest gaan uitzoeken.

De student kan omgaan met software oplossingen voor versiebeheer (GIT) en kan deze software inzetten voor het ontwikkelen van code:

GitHub link: https://github.com/zoevdpol/vision_eindopdracht

Ik heb mijn code, planning, weekupdates en modules allemaal in een repository staan die ik regelmatig heb geupdate met commits in verschillende branches.

De student kan een experiment opzetten en uitvoeren om aan te tonen hoe de geïmplementeerde vision-methode werkt, hoe effectief deze methode en implementatie is en waar grenzen liggen van de gekozen methode en implementatie:

Onderzoeksmethode + Proefbeschrijving

Resultaten + Conclusie

De student kan een rapport schrijven om het gekozen en uitgevoerde experiment te beschrijven, de resultaten te beschrijven en conclusies te trekken:

Rapport schrijven: dit verslag zelf is het bewijs daarvan

Uitgevoerde experiment beschrijven : Onderzoeksmethode + Proefbeschrijving

Resultaten beschrijven en conclusies trekken: Resultaten + Conclusie

De student kan een zelf-opgezet experiment uitvoeren en de resultaten op gestructureerde wijze verzamelen:

Ik heb voordat ik het onderzoek begon een onderzoeksmethode en proefbeschrijving geschreven. Dit heb ik gedaan zodat het duidelijk was hoe ik precies mijn onderzoek wou gaan

uitvoeren. Ook heb ik alle resultaten genoteerd en verzameld en daruit een conclusie getrokken.

Onderzoeksmethode+ Proefbeschrijving + Resultaten+ Conclusie

Planning

Planning 1:

Datum: 17/02/2023

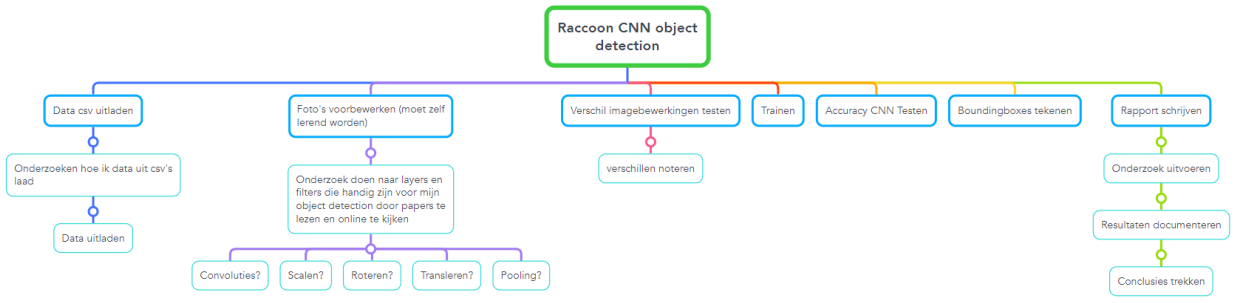
Ik ben van plan om vandaag (vrijdag 17 februari) nog module 6 af te maken, daar ben ik nu namelijk halverwege mee, als ik die dan af heb heb ik alle modules en opdracht 7 paper samenvatten al af en kan ik maandag gaan nadenken over de stappen die ik moet nemen om mijn object detection te gaan maken. Zodat ik de dag erna al een goed start kan gaan maken.

Planning 2:

Datum: 22/02/2023

Waar ben ik nu?:

Ik heb gisteren (22/02/2023) mijn opdrachten van Vision afgekregen. Daarom ben ik vandaag begonnen aan de eindopdracht. Ik ben begonnen met een WBS (work breakdown structure) te maken zodat ik in de basis weet wat ik moet doen. Het is me alleen nog niet helemaal duidelijk hoe ik alles ga doen aangezien ik niet precies weet hoe dit werkt en er nog wel wat verwarring over heb. Daarna ben ik gaan kijken hoe de tfrecords werkten, ik kon er niet veel informatie over vinden die ik begreep dus ben overgestapt naar een andere dataset van wasberen omdat die een csv heeft inplaats van tfrecords.



Work Breakdown Structure

Ik ben van plan om van links naar rechts te werken.

Weekupdates

Weekupdate week 3:

Datum: 26/02/2023

Afgelopen week heb ik alle modules en opdrachten afgerond en ben ik begonnen aan de eindopdracht. Helaas merkte ik al snel dat ik niet genoeg informatie had over tfrecords dus ben overgestapt naar een dataset met een csv. Ik kan de foto's uitladen door middel van de csv, gezorgd dat alle foto's dezelfde grootte zijn en ben gaan proberen de goeie boundingboxes bij de trainingsplaatjes te tekenen om te kijken als dat goed werkte, helaas merkte ik vrijdag middag dat ze boundingboxes niet helemaal kloppen dus ik ben van plan om dat morgen uit te gaan zoeken en dan meteen verder te gaan met wat ik in mijn planning heb staan. Ook heb ik meer een idee gekregen hoe mijn eindopdracht er uiteindelijk gaat uitzien. Ik moet alleen veel onderzoek gaan doen naar layers en dergelijke.

Weekupdate week 4:

Datum: 12/03/2023

Mijn laptop is sinds vrijdag niet meer werkend dus gebruik nu een leenlaptop van school, gelukkig had ik mijn code op github (naast een paar makkelijke regels code), dit zorgde wel ervoor dat ik wat tijd ben verloren aan alles op de leenlaptop zetten waardoor ik minder tijd had voor de opracht. Ik heb nu mijn wasbeer object detetction “werkend” alsin dat hij voorspellingen doet en daaruit bounding boxes tekend maar dat ze nog heel slecht / niet correct zijn, dus ik ga komende week goed onderzoeken wat voor layers ik kan toevoegen. Hoop aan de eind van de week iets te hebben wat best goed werkt zodat ik de week erna heb voor het verslag en eventuele uitloop.

Weekupdate week 5:

Datum: 19/03/2023

Ik heb nu een beter werkende object detection met behulp van mobilenet, dit heb ik gedaan door alle train foto's te croppen op basis van de boundingboxes, en daarna een border omheen te doen zodat ze allemaal 224 x 224 zijn zonder dat ze gewarpt worden. Het resultaat is daardoor dus ook veel beter, deze week ga ik met layers enzo experimenteren en mijn verslag daarover schrijven.

Weekupdate week 6:

Datum: 26/03/2023

Mijn opdracht is af, ik moet bij mijn verslag alleen nog even kijken voor spelfouten en dan kan alles ingeleverd worden.