Bachelor Informatica

University of Amsterdam

Herkenning van weggelakte tekst in Wob-documenten: een onderzoek naar effectieve methodes

Roderick Majoor

27 januari 2023

Supervisor(s): Dr. M.J. Marx

Signed:

Samenvatting

Wob-documenten bevatten weggelakte stukken waardoor soms veel informatie mist op een pagina. Het is interessant om te weten hoeveel weggelakte stukken een document bevat om er zo achter te komen hoeveel informatie daadwerkelijk in documenten te vinden is. In deze scriptie worden methodes onderzocht om weggelakte stukken in een document automatisch te herkennen met een computer. Er worden twee methodes met elkaar vergeleken op basis van accuraatheid en uitvoertijd.

Inhoudsopgave

HOOFDSTUK 1

Introductie

1.1 Wet Openbaarheid van Bestuur

De *Wet openbaarheid van bestuur* (Wob)[1] was de wet die tot 1 mei 2022 de openbaarmaking van informatie door de Nederlandse overheid regelde. De wet is vervangen door de *Wet open overheid* (Woo)[2] maar in de media wordt vaak nog gerefereerd naar de oude wet.

Burgers kunnen Wob/Woo-verzoeken indienen bij de overheid om inzicht te krijgen in handelingen van de overheid. Hieruit kan bijvoorbeeld blijken waarom de overheid bepaalde keuzes heeft gemaakt om tot een beslissing te komen. Deze Wob-documenten bevatten vaak gevoelige informatie zoals persoonsgegevens. Deze gevoelige informatie wordt uit deze documenten 'weggelakt' voordat deze vrijgegeven worden.

De keuze wat wel en niet weg te lakken ligt bij de overheidsinstantie waar het Wob-verzoek wordt ingediend. Dit zou ertoe kunnen leiden dat bepaalde instanties meer of minder weglakken dan andere. Uit onderzoek van RTL Nieuws is gebleken dat er ook onterecht stukken uit Wob-documenten wordt weggelakt (Regt en Strijker 2021). Voor bijvoorbeeld journalisten is het interessant om te weten hoeveel weggelakte stukken er in documenten zit zodat duidelijk is hoeveel informatie er daadwerkelijk is vrijgegeven in een document.

1.2 Document Segmentatie

Document segmentatie is een vorm van afbeelding segmentatie waarbij de afbeelding een document is. Het doel van document segmentatie is het opdelen van het document in verschillende klassen (Yang e.a. 2017). Zo kan een document bijvoorbeeld worden opgesplitst in titels, paragrafen en afbeeldingen. Figuur 1.1 geeft een voorbeeld van document segmentatie, waarbij artikelen worden opgedeeld in meerdere klassen. Segmentatie is een belangrijk stap bij het analyseren van documenten. In dit onderzoek zullen we documenten segmenteren om de weggelakte stukken te vinden.

[1] https://www.rijksoverheid.nl/onderwerpen/wet-openbaarheid-van-bestuur-wob
[2] https://www.rijksoverheid.nl/onderwerpen/wet-open-overheid-woo

Figuur 1.1: Voorbeeld van document segmentatie waarbij artikelen worden opgesplitst in verschillende delen.[4]

1.3 Etische Aspecten

In dit onderzoek proberen we erachter te komen hoe weggelakte stukken in Wob-documenten het best kunnen worden herkend. Daarnaast zullen we per pagina een analyse maken over de hoeveelheid informatie die is weggelakt. Mogelijk kunnen de benoemde methodes in dit onderzoek kunnen worden gebruikt om te laten zien bij welke documenten (onnodig) veel wordt weggelakt. Dit zou mogelijk kunnen helpen om een verbeterde transparantie in informatieopenbaring te bewerkstelligen voor burgers. Dit zou ethisch gezien een goede stap zijn.

1.4 Einddoel

Het doel van dit onderzoek is om een segmentatiemethode te vinden om zwart en grijs weggelakte stukken in Wob-documenten te herkennen. Aan de hand daarvan kan per pagina van een document worden bekeken hoeveel weggelakte stukken er te vinden zijn, hoeveel procent van de tekstoppervlakte is weggelakt en hoeveel procent van de karakters is weggelakt. Daarnaast kan dan een inschatting worden gemaakt over het aantal weggelakte karakters op de pagina. Op die manier ontstaat er een duidelijk beeld over hoeveel informatie daadwerkelijk is vrijgegeven in een document en kan er worden bekeken welke instanties veel of weinig weglakken. Figuur 1.2 geeft een voorbeeld van herkende weggelakte stukken met groene omlijning. Accuraatheid van de methodes wordt bepaald aan de hand van de *panoptic quality metric*. Wob-documenten bestaan soms uit duizenden pagina's. Daarom proberen we ook de uitvoertijd van de methode zo kort mogelijk te maken. We zullen in dit onderzoek twee methodes testen waarbij de eerste methode zelf gemaakt is en de tweede methode een bestaande implementatie is die wordt toegepast op ons probleem. De eerste methode is regel-gebaseerd en de tweede methode is een machine learning techniek.

[4] https://clgiles.ist.psu.edu/pubs/CVPR2017-connets.pdf

Retouradres Postbus 2003, 1000 CA Amsterdam

Blenheim Advocaten
t.a.v. [redacted]
Postbus 10302
1001 EH AMSTERDAM

Datum 12 augustus 2020
Ons kenmerk NW20-11439-UIT-20-19839
Behandeld door [redacted]

Onderwerp Uw verzoek in het kader van de Wet openbaarheid van bestuur d.d. 20 juli 2020

Geachte heer/mevrouw [redacted],

Hierbij bevestig ik dat ik op **11 augustus 2020** uw brief van **20 juli 2020** heb ontvangen. Met deze brief verzoekt u om **alle informatie over Zwarte Pad 19-A van de afgelopen 20 jaar** op grond van de Wet openbaarheid van bestuur (hierna: Wob). Het registratienummer van uw brief is **NW20-11439-IN-20-09031.**

In de Wob is bepaald dat binnen vier weken na ontvangst van het verzoek beslist dient te worden op het verzoek, maar dat verdaging van de beslistermijn met vier weken mogelijk is. Het is niet mogelijk om binnen vier weken op uw verzoek te beslissen. De reden hiervoor is dat er uitgebreid archiefonderzoek nodig is. Op grond van artikel 6 van de Wob verleng ik de beslistermijn van uw verzoek dan ook met vier weken. U ontvangt derhalve uiterlijk op **6 oktober 2020** van mij antwoord op uw verzoek.

Ik vertrouw erop u hiermee voldoende te hebben geïnformeerd. U kunt contact opnemen met [redacted] indien u nog vragen heeft over de status en de termijn van afhandeling van uw verzoek.

Met vriendelijke groet,

[redacted]
[redacted]
stadsdeel Nieuw-West
gemeente Amsterdam

Figuur 1.2: Voorbeeld van herkende weggelakte stukken omlijnd in groen. Aan de hand van de gevonden stukken kan worden bepaald hoeveel procent van de tekstoppervlakte is weggelakt en er kan een inschatting worden gemaakt over het aantal weggelakte karakters.

HOOFDSTUK 2

Theoretische Achtergrond

Dit hoofdstuk beschrijft de gebruikte methodes en de meest belangrijke achtergrondinformatie die helpen om het onderzoek te begrijpen.

2.1 Gerelateerd Werk

In dit onderzoek gaan we op zoek naar weggelakte stukken in documenten. We zullen dit doen op basis van segmentatietechnieken die worden toegepast op PDF pagina's. Echter beschrijven eerdere onderzoeken op dit gebied een andere methode voor het oplossen van ons probleem. Bland, Iyer en Levchenko (2022) beschrijven een methode die gebruik maakt van de data van PDF bestanden om weggelakte stukken te vinden. PDF data bevat de elementen op de PDF zoals tekst en tekeningen met daarbij informatie over de locatie op de PDF van deze elementen. Zij gebruiken een algoritme om op basis van die informatie rechthoeken in de tekst te vinden die als weggelakte stukken worden herkend. Eenzelfde soort methode wordt bijvoorbeeld gebruikt in de *x-ray tool*[1].

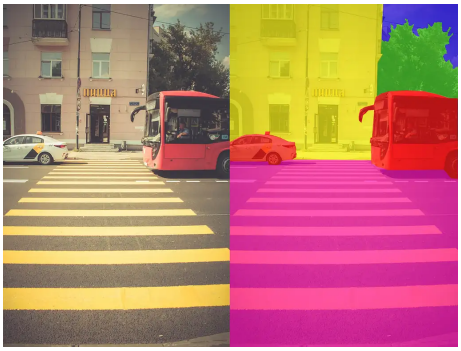
Deze methodes zijn erg snel in vergelijking met onze methode die werkt aan de hand van afbeelding operaties. Echter ontdekten we een aantal imperfecties in hun methode. Het grootste probleem is dat deze methode niet werkt op PDF bestanden op welke *PDF rasterization* is toegepast. Dit wil zeggen dat de pagina's van een PDF bestand zijn geconverteerd naar afbeeldingen of zijn ingescand waardoor de tekst niet meer uit te lezen is in de data van de PDF. Onze dataset bevat veel van dit soort documenten, waardoor het nodig is een methode te maken die ook werkt op deze documenten. Onze methodes werken wel op deze documenten.

2.2 Segmentatie

2.2.1 Semantic Segmentation

Semantic segmentation is de vorm van segmentatie waarbij alle pixels in een afbeelding in een bepaalde semantische klasse worden toegewezen. Dit betekent dat pixels met dezelfde *features* worden toegewezen tot dezelfde klasse (Mo e.a. 2022). In Figuur 2.1 is een voorbeeld te zien van een semantic segmentation op een afbeelding. Hierbij is de afbeelding opgedeeld in vijf verschillende klassen waar de pixels aan worden toegewezen. Bij deze vorm van segmentatie wordt er geen onderscheid gemaakt tussen objecten in dezelfde klasse. Zo worden de bus en de auto op de afbeelding bijvoorbeeld in dezelfde klasse geplaatst.

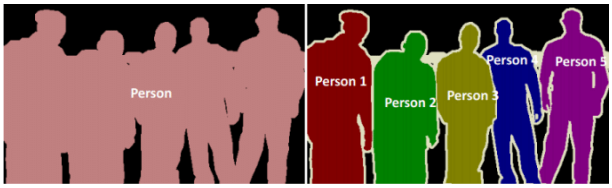
[1] https://github.com/freelawproject/x-ray
[2] https://upload.wikimedia.org/wikipedia/commons/d/d4/Image_segmentation.png



Figuur 2.1: De originele foto (links) en de foto na semantic segmentation (rechts).[2]

2.2.2 Instance Segmentation

Instance segmentation is een vorm van segmentatie waarbij wordt gezocht naar alle afzonderlijke objecten in een afbeelding. Deze segmentatie heeft als doel onderscheid te kunnen maken tussen meerdere instanties van hetzelfde object (Hafiz en Bhat 2020). Op Figuur 2.2 is het verschil tussen semantic segmentation en instance segmentation te zien. Bij semantic segmentatin worden meerdere personen in dezelfde klasse geplaatst terwijl bij instance segmentation de verschillende personen van elkaar worden onderscheiden.

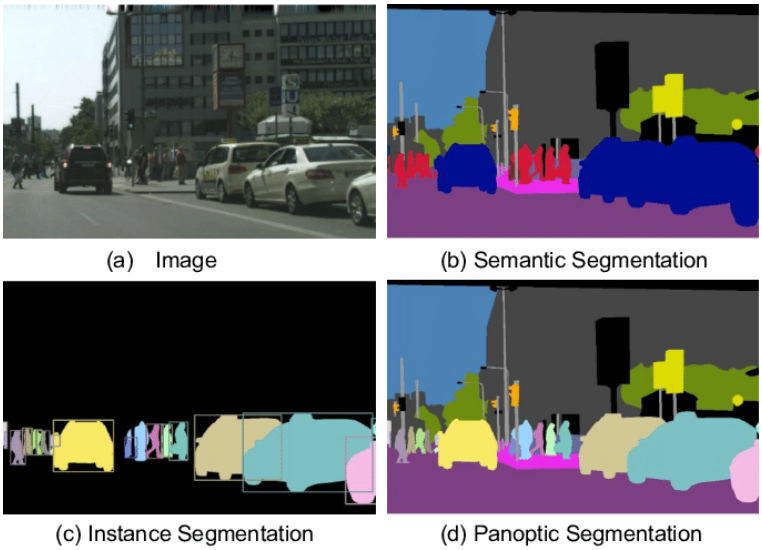


Figuur 2.2: Semantic segmentation (links) en instance segmentation (rechts).[3]

2.2.3 Panoptic Segmentation

Panoptic segmentation kan worden gezien als een combinatie van semantic segmentation en instance segmentation. Bij deze methode krijgt elke pixel in de afbeelding een *semantic label* en een *instance id* toegewezen. Pixels met hetzelfde label en id behoren dan tot hetzelfde object (Kirillov e.a. 2018). In Figuur 2.3 zijn de verschillen tussen de tot dusver genoemde vormen van segmentatie te zien.

[3]https://www.researchgate.net/profile/Vinorth-Varatharasan/publication/339328277/figure/fig1/AS:864554888204291@1583137
segmentation-left-and-Instance-segmentation-right-8.ppm



Figuur 2.3: De originele afbeelding (a), de afbeelding na semantic segmentation (b) en de afbeelding na instance segmentation (c). De combinatie tussen (b) en (c) geeft de uiteindelijke panoptic segmentation (d). [5]

2.3 Segmentatie Technieken

In dit onderzoek maken we gebruik van twee methodes. De eerste is een regel-gebaseerde methode en de tweede is een machine learning methode. Het voordeel van een regel-gebaseerde aanpak is dat er geen training nodig is en de methode simpel kan worden gemaakt om te werken voor zowel zwarte als grijze lak. Bij de machine learning methode zou het nodig zijn het model te trainen met beide soorten lak. Een groot voordeel van een machine learning aanpak is wel dat deze beter worden naarmate er meer wordt getraind, waardoor deze met veel training ook veel verschillende soorten documenten kan segmenteren.

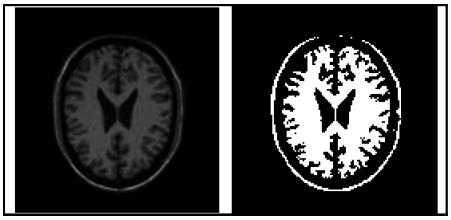
Hieronder worden de technieken die voor beide methodes worden gebruikt toegelicht.

2.3.1 Regel-gebaseerd

Threshold

Thresholding is een van de meest simpele segmentatietechnieken. Bij een simpele treshold methode wordt voor een grijswaarde foto een bepaalde waarde tussen 0 (zwart) en 255 (wit) gebruikt als threshold. Als de grijswaarde van een pixel in de foto kleiner is dan de threshold waarde, wordt deze op 0 gezet. Als deze groter is dan de threshold waarde, wordt deze op 255 gezet. Nadat dit voor elke pixel is gebeurd blijft er een foto over met twee klassen: zwarte pixels en witte pixels (Zanaty 2016). Een voorbeeld van een threshold is te zien in Figuur 2.4.

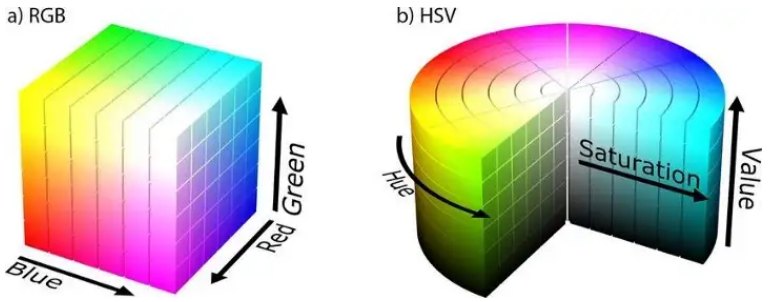
[5] https://www.researchgate.net/figure/b-semantic-segmentation-c-instance-segmentation-and-d-panoptic-segmentation-for_fig5_342409316



Figuur 2.4: De originele foto (links) en de foto na de eenvoudige threshold (rechts).[6]

Kleurdetectie

Het is mogelijk om kleuren te vinden in afbeeldingen. Deze techniek kan gebruikt worden voor het segmenteren van afbeeldingen op basis van kleuren van objecten op de afbeelding. Voor deze segmentatie wordt gebruik gemaakt van de HSV (Hue, Saturation, Value) kleurenruimte. Met deze kleurrenruimte is het makkelijker om een kleur te selecteren door een range te bepalen in de Hue. Daardoor is dit model geschikter voor het segmenteren van afbeeldingen dan de RGB kleurenruimte (Mohd Ali 2013). In Figuur 2.5 is het verschil tussen de modellen te zien.



Figuur 2.5: De RGB kleurenruimte (links) en de HSV kleurenruimte (rechts).[7]

Morphological Operations

Morphological operations[8] zijn operaties die gebruikt kunnen worden om afbeeldingen te bewerken. Deze operaties bekijken voor elke pixel in een afbeelding de *neighbourhoud* van de pixel en bepalen aan de hand daarvan welke waarde de pixel wordt. De *structuring element* of *kernel* van de operatie is een matrix die aangeeft welke pixels in de neighbouhood zitten (Soille 2013). We focussen op twee soorten operaties.

De eerste operatie is *erosion*. Deze operatie kan worden gebruikt om randen van een object in een foto te verwijderen. Afhankelijk van de grootte van de kernel kan worden bepaald hoeveel van de rand van het object wordt verwijderd. Deze operatie kan goed worden gebruikt voor het verwijderen van *noise*.

De tweede operatie is *dilation*. Dit is eigenlijk precies het tegenovergestelde van erosion. In dit geval worden de randen van een object dus groter. Erosion en dilation worden vaak na elkaar gebruikt om zo eerst noise te verwijderen en daarna de originele objecten op de foto weer te vergroten naar het origineel (Bockholt, Cavalcanti en Mello 2011).

Deze operaties kunnen worden gebruikt om kleine objecten zoals tekst van een afbeelding te verwijderen terwijl grotere objecten blijven staan. Op deze manier kan het ingezet worden om een afbeelding te segmenteren. Een voorbeeld van deze operaties is te zien in Figuur 2.6.

[6] https://www.researchgate.net/profile/E-Zanaty/publication/294682473/figure/fig1/AS:329691148898318@1455615903814/Segm by-thresholding-technique-threshold04-Taken-from-Sahoo-et-al-12.png

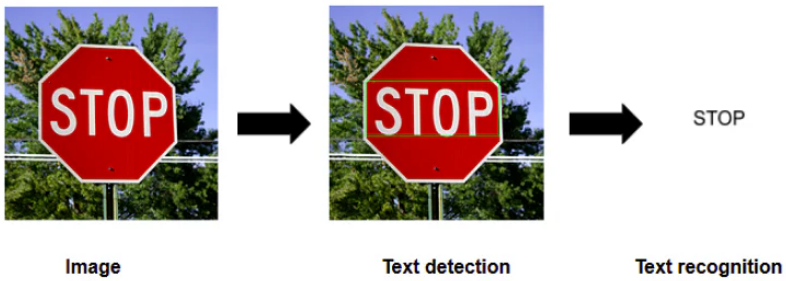
[7] https://miro.medium.com/max/720/1*W30TLUP9avQwyyLfwu7WYA.webp

[8] https://docs.opencv.org/4.x/d9/d61/tutorial_py_morphological_ops.html

Figuur 2.6: De originele foto (links), de foto na erosion (midden) en de foto na dilation (rechts).

Optical Character Recognition

Optical Character Recognition (OCR) is het proces waarbij foto's met tekst worden omgezet naar tekst die voor een computer te begrijpen is. Hierbij is het mogelijk om zowel geprinte, geschreven als getypte tekst te herkennen en deze om te zetten. Naast de tekst zelf wordt er bij het proces ook informatie gegeven over de plaats waar de tekst te vinden is op de foto. Daarmee is het mogelijk om een document te verdelen tussen tekstgebieden en niet-tekstgebieden (Thakare e.a. 2018). In Figuur 2.7 is te zien hoe OCR in zijn werking gaat. OCR kan worden gebruikt om een afbeelding te segmenteren op basis van tekst- en niet tekstgebieden.



Figuur 2.7: De originele foto (links), de foto met herkende tekst (midden) en de omgezette tekst na OCR (rechts).[10]

2.3.2 Machine Learning

Mask Region-Based Convolutional Neural Network

Het gebruik van *mask region-based convolutional neural networks* (Mask R-CNN) is de *state of the art* voor segmentatie van afbeeldingen en objectherkenning. He e.a. (2017) beschrijven deze techniek, die *machine learning* gebruikt om te voorspellen waar objecten zich in afbeeldingen bevinden. De *backbone* van de Mask R-CNN is ResNet101[11] en een *Feature Pyramid Network* (FPN) welke gebruikt worden om features van de afbeelding te herkennen. De Mask R-CNN werkt in twee fases.

De eerste fase is een *regional proposal network* (RPN). Deze RPN maakt regio's van verschillende groottes op de afbeelding en bepaalt welke van die regio's de meeste kans hebben om mogelijke objecten te hebben. De uiteindelijke voorspellingen van de RPN zijn de *regions of interest* (ROIs) en worden doorgestuurd naar de volgende fase.

De tweede fase gebruikt een dieper neural network dan in de eerste fase. Waar in de eerste fase ROIs alleen nog in een voorgrond of achtergrond klasse werden gescheiden, worden in deze

[10] https://www.projectpro.io/article/how-to-train-tesseract-ocr-python/561
[11] https://www.mathworks.com/help/deeplearning/ref/resnet101.html;jsessionid=0196e6990a04b1b893c3c1ac786b

fase de ROIs uit de eerste fase gebruikt voor het classificeren van objecten in de afbeelding. Dit betekent dat de ROIs in deze fase in de juiste klasse worden geplaatst (e.g. persoon, auto, boom etc.). Daarnaast wordt in deze fase de *bounding box* van het object in de ROI beter bepaald zodat de bounding box uiteindelijk het object correct omringd. De laatste en tevens ook unieke stap in Mask R-CNN is het maken van *pixel-level binary masks* voor ieder object. Hierdoor wordt bepaald welke pixels allemaal tot een object behoren (He e.a. 2017).

De Mask R-CNN wordt getraind door het afbeeldingen te geven met daarbij informatie over de locatie van de ground truth objecten in de afbeelding. Hiervoor is het dus nodig om een trainingset te maken met ingetekende ground truth van de locatie van weggelakte stukken op de pagina.

HOOFDSTUK 3

Methode

Voor dit onderzoek wordt de snelheid en accuraatheid van twee verschillende methodes met elkaar vergeleken. In de eerste methode maken we zelf een werkwijze met behulp van Tesseract en OpenCV om de weggelakte stukken tekst te vinden. In de rest van dit verslag zal naar deze methode worden gerefereerd als de *Regel-gebaseerde methode*. Bij de tweede methode wordt gebruik gemaakt van de Mask R-CNN om de weggelakte stukken tekst te vinden. Naar deze methode zal dan ook worden gerefereerd als de *Machine Learning methode*.

3.1 Hardware

Voor de experimenten wordt gebruik gemaakt van een Intel® Core™ i5-8265U 1.6GHz Processor met 8 GB DDR4 Memory. Voor het trainen van de Mask R-CNN implementatie wordt gebruik gemaakt van een Tesla T4 GPU.

3.2 Software

Voor dit onderzoek wordt Python versie 3.8 gebruikt. De belangrijkste libraries die gebruikt worden zijn OpenCV[1] en Python-Tesseract[2]. OpenCV wordt gebruikt om documenten om te zetten naar afbeeldingen en hier alle benodigde computer vision technieken op toe te passen. Python-Tesseract is een wrapper voor de Google Tesseract OCR engine in Python. Voor de Mask R-CNN gebruiken we de implementatie van Matterport[3].

3.3 Dataset

De dataset die voor dit onderzoek wordt gebruikt zijn gepubliceerde Wob-documenten. Deze documenten zijn onder te verdelen in besluitbriefen en vrijgegeven documenten. Bij besluitbriefen is vaak alleen persoonlijke informatie weggelakt. Dit betekent dus dat er met name kleinere stukjes tekst zijn weggehaald. Bij vrijgegeven documenten kan er zowel veel als weinig tekst weggelakt zijn. In sommige gevallen is de hele pagina of een heel document weggelakt en in andere gevallen is er niets weggelakt.

Naast het verschil in hoeveelheid weggelakte tekst is er ook een verschil in de manier van lakken. Een aantal documenten zijn automatisch gelakt met behulp van bijvoorbeeld een tool als Zylab[4]. Deze automatisch gelakte stukken zijn vaak rechthoeken met een code die aangeeft waarom een stuk tekst is weggelakt. Er zijn echter ook documenten die met de hand zijn weggelakt of waar

[1] https://opencv.org/
[2] https://pypi.org/project/pytesseract/
[3] https://github.com/matterport/Mask_RCNN
[4] https://www.zylab.com/

door tekst is heen gekrast met een pen. Ook zijn weggelakte stukken niet allemaal dezelfde kleur en verschillen ze van vorm. De data komt uit drie verschillende datasets. De *Zylab dataset* bevat 46 afbeeldingen die allemaal met de Zylab tool weggelakt zijn. De *Corpus1 dataset* bestaat uit 55 afbeeldingen en bevat een combinatie van verschillende weggelakte stukken. Dit zijn dus zowel zwarte als grijze blokken en bevatten soms wel en soms geen code. Ook bevat deze set documenten die met pen zijn weggekrast. De *Corpus2 dataset* bevat 45 afbeeldingen en heeft alleen zwarte weggelakte stukken. Veruit de meeste documenten zijn op deze manier gelakt. Een voorbeeld van de verschillende lak in de datasets is te zien in Figuur 3.1.

(a)

(b)

(c)

Figuur 3.1: Een voorbeeld van weggelakte tekst uit de Zylab dataset (a) en de Corpus2 dataset (b). De Corpus1 dataset bestaat uit een combinatie van verschillende lak en bevat ook met pen weggekraste tekst (c).

3.4 Ground Truth Values

De dataset van Wob documenten bevat zelf geen ground truth values. Het is dus nodig om deze zelf te maken. Hiervoor gebruiken we de VGG Image Annotator[5]. In deze tool annoteren we ground truth values op de afbeeldingen die we later kunnen gebruiken voor evaluatie. Er wordt gekozen voor een strenge beoordeling van de ground truth values. Dit wil zeggen dat weggelakte stukken die overlap hebben maar wel onderscheidbaar zijn door bijvoorbeeld een code, als aparte stukken worden gezet in de ground truth values. Een voorbeeld hiervan is te zien in Figuur 3.2. Daarnaast worden weggelakte stukken waar witruimte tussen zit ook als aparte stukken aangegeven. Dit kan ertoe leiden dat er soms meerdere weggelakte stukken worden geannoteerd als een enkel stuk. Hier is voor gekozen omdat het vaak onmogelijk is een goede onderscheiding te maken tussen alle verschillende stukken als deze aan elkaar grenzen.

Er is voor gekozen om weggelakte stukken zo goed mogelijk te omringen. De meeste weggelakte stukken zijn rechthoeken en kunnen dus in de ground truth values als rechthoek worden aangegeven. Echter zijn bijvoorbeeld de met de hand gelakte stukken niet precies rechthoekig. Voor deze stukken is dan ook gekozen om een *polygon* als ground truth te gebruiken om het stuk zo goed mogelijk te omringen.

Doordat deze annotaties van de ground truth values met de hand is gedaan, is het mogelijk dat de bounding boxes niet exact pixel voor pixel kloppen. Echter zal deze afwijking zo klein zijn dat het niet teveel van invloed zal zijn op het onderzoek.

[5]https://www.robots.ox.ac.uk/ vgg/software/via/

Figuur 3.2: Een voorbeeld van overlappende weggelakte stukken (links) met daarbij de ground truth values ingetekend (rechts). Er zijn 5 verschillende stukjes geannoteerd als ground truth.

3.5 Manier van Evaluatie

We onderscheiden onze evaluatie in twee soorten. De eerste is een echte evaluatie van de voorspellingen. Deze geeft dus aan hoe goed onze methodes daadwerkelijk werken. Dit doen we aan de hand van de PQ score die hieronder wordt beschreven. De tweede evaluatie geeft informatie over de hoeveelheid lak op een pagina als output van het programma. De correctheid van deze maten zullen we controleren aan de hand van de *Root Mean Square Error* (RMSE) welke wordt beschreven vanaf Sectie 3.5.2.

3.5.1 Panoptic Quality

Om te bepalen of onze methodes goede voorspellingen maken, zullen we gebruik maken van de *panoptic quality* (PQ) *metric*. Dit is een manier om te bepalen in hoeverre een voorspelling van een segmentatie overeenkomt met de daadwerkelijke *ground truth* (Kirillov e.a. 2018). PQ werkt in twee stappen.

De eerste stap is het matchen van de voorspelde segmenten en de ground truth segmenten. Segmenten worden als match beschouwd op het moment dat de *intersection over union* (IoU) tussen de segmenten groter is dan 0.5. De IoU geeft aan in hoeverre twee segmenten met elkaar overeen komen door de doorsnede te delen door de vereniging. In Figuur 3.3 is visueel duidelijk gemaakt hoe deze berekening werkt en Figuur 3.4 geeft voorbeelden van verschillende IoU waarden.

De tweede stap is het berekenen van de PQ waarde. Hiervoor worden de voorspelde en ground truth segmenten verdeeld in drie sets: *true positives* (TP), *false positives* (FP) en *false negatives* (FN). De TP set zijn de voorspelde en ground truth segmenten die als match zijn beschouwd in de eerste stap. De FP set zijn voorspelde segmenten die geen match hebben en de FN set zijn de ground truth segmenten zonder match. In Figuur 3.5 is te zien hoe dit werkt.

Na het bepalen van de drie sets wordt de waarde van PQ bepaald door:

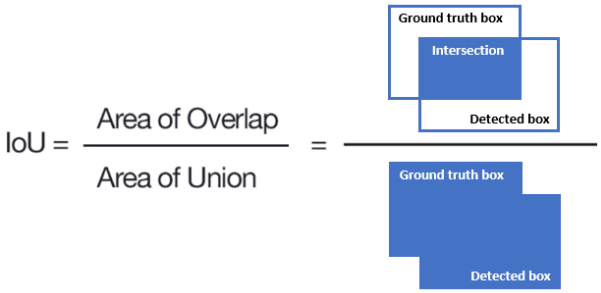
$$\mathrm{PQ} = \frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|} \tag{3.1}$$

Hierbij is $\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}$ het gemiddelde van de IoU waardes van de segmenten die als match zijn gezien. $\frac{1}{2}|FP| + \frac{1}{2}|FN|$ wordt in de noemer toegevoegd om segmenten zonder match te straffen. PQ kan ook gezien worden als de vermenigvuldiging van een *segmentation quality* (SQ) term en een *recognition quality* (RQ) term:

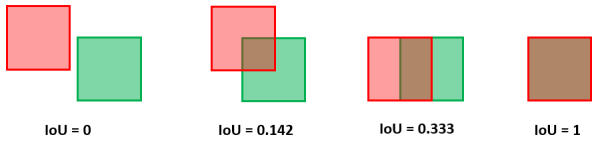
$$\mathrm{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}} \tag{3.2}$$

Hierbij kan de RQ term worden herkend als de F_1 score[6] die veel gebruikt wordt voor het bepalen van de kwaliteit van een model.

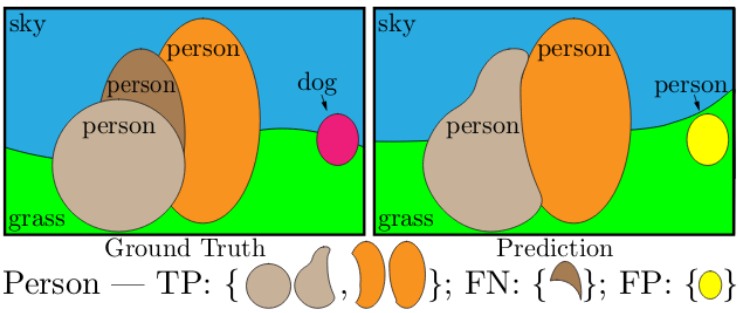
[6]https://deepai.org/machine-learning-glossary-and-terms/f-score: :text=The%20F%2Dscore%2C%20also%20called,positive'%20or%20'nega



Figuur 3.3: Berekening van de IoU tussen segmenten van een ground truth en een voorspelling.[8]



Figuur 3.4: Voorbeelden van verschillende ground truths met voorspellingen en hun IoU waarde.[10]



Figuur 3.5: Voorbeeld van de verdeling van segmenten in de TP, FP en FN sets. Hier wordt gekeken naar een *person* klasse. Segmenten die in zowel de ground truth als voorspelling worden gezien zijn aangegeven met dezelfde kleur en worden gezien als TP. Segmenten die wel aanwezig zijn in de ground truth maar niet in de voorspelling worden gezien als FN. Segmenten die wel in de voorspelling aanwezig zijn maar niet in de ground truth worden gezien als FP. [12]

[8] https://www.baeldung.com/cs/object-detection-intersection-vs-union
[10] https://www.baeldung.com/cs/object-detection-intersection-vs-union
[12] https://kharshit.github.io/blog/2019/10/18/introduction-to-panoptic-segmentation-tutorial

3.5.2 Root Mean Square Error

We gebruiken de RMSE om de kwaliteit van de evaluatiematen van ons prorgramma te testen. Dit meet in hoeverre er verschil is tussen waardes die door ons programma worden voorspeld en de werkelijke waardes. De formule ziet er als volgt uit:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \hat{x_i})^2}{N}} \tag{3.3}$$

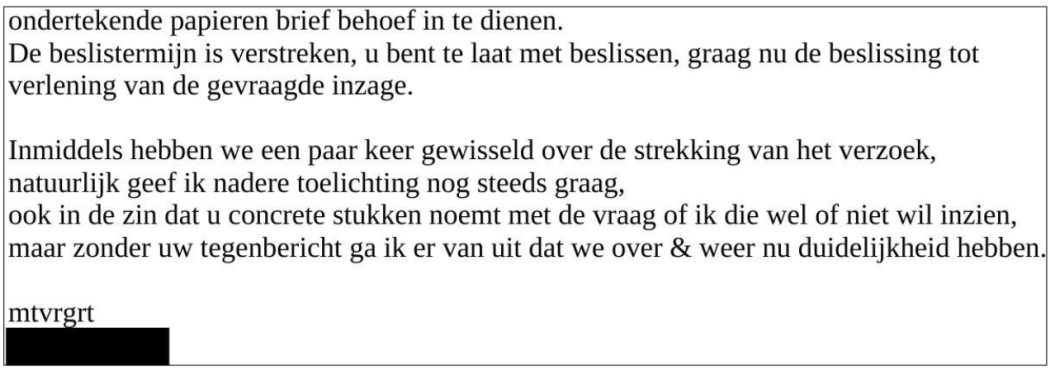
Hierbij is x_i de voorspelling van ons programma en $\hat{x_i}$ de daadwerkelijke waarde die we met de hand opmeten. N is het aantal data punten. We gebruiken de RMSE voor de evaluatiematen van ons programma. Deze evaluatiematen beschrijven we in de subsecties hieronder.

Weggelakte Stukken

De eerste evaluatiemaat over het lakgehalte op een pagina is het optellen van het aantal weggelakte stukken op een pagina. Dit geeft een duidelijke indicatie over hoeveel gevoelige informatie op een pagina te vinden is.

Percentage Weggelakte Tekst in Tekstvlak

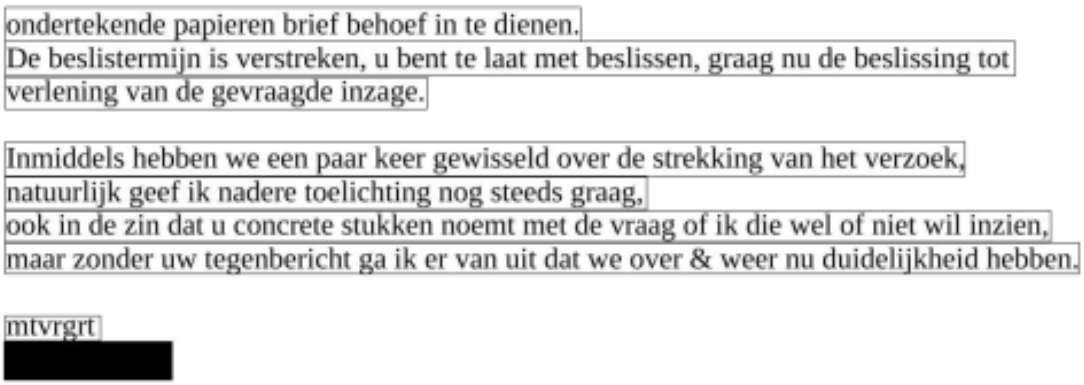
De tweede evaluatiemaat over het lakgehalte is het percentage van weggelakte tekst in het tekstvlak van de pagina. Het tekstvlak wordt gedefinieerd als de rechthoek tussen het karakter dat het meest linksboven staat en het karakter dat het meest rechtsonder staat. Figuur 3.6 laat een voorbeeld van een tekstvlak zien. Dit wordt toegepast door met Tesseract de extreme punten van de tekst te vinden en met OpenCV de extreme punten van de weggelakte stukken te vinden. Vervolgens wordt er een rechthoek gemaakt tussen het punt dat het meest linksboven ligt en het punt dat het meest rechtsonder ligt. Als laatste wordt de totale oppervlakte van de weggelakte stukken gedeeld door de oppervlakte van het tekstvlak om zo een percentage te krijgen van weggelakte tekst.



Figuur 3.6: Voorbeeld van de oppervlakte van een tekstvlak.

Percentage Weggelakte Tekst in Karakteroppervlakte

De derde evaluatiemaat over het lakgehalte is het percentage van weggelakte tekst in de karakteroppervlakte. Dit maken we door met Tesseract de oppervlakte van alle herkende tekst op te tellen. Hierna wordt de oppervlakte van de weggelakte stukken bij elkaar opgeteld. Vervolgens delen we deze oppervlakte door de totale oppervlakte om een percentage te krijgen. Een voorbeeld van de oppervlakte van de herkende tekst is in Figuur 3.7 gegeven.



Figuur 3.7: Voorbeeld van de oppervlakte van de karakters.

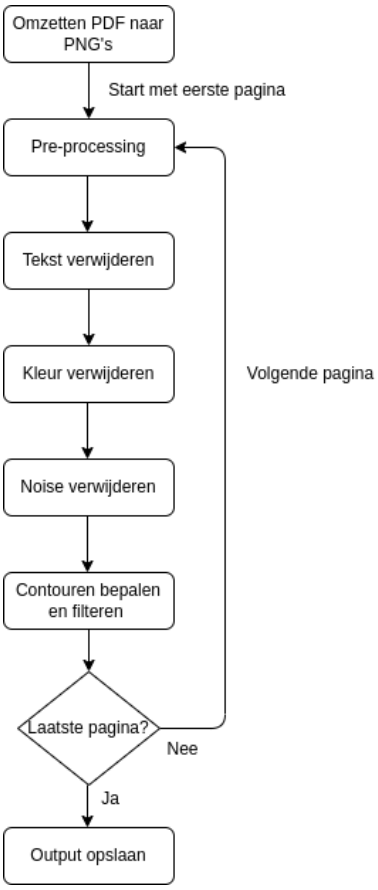
Weggelakte Karakters

De laatste evaluatiemaat over het lakgehalte is een inschatting over het aantal weggelakte karakters op een pagina. Deze inschatting maken we met de volgende formule:

$$\text{Weggelakte Karakters} = \frac{\text{Aantal Gevonden Karakters}}{\text{Totale Oppervlakte Gevonden Karakters}} \cdot \text{Totale Oppervlakte Weggelakte Gebieden} \tag{3.4}$$

3.6 Methode

Documenten in onze dataset bestaan over het algemeen uit voornamelijk tekst. Echter kunnen er bijvoorbeeld ook afbeeldingen, logo's, grafieken en tabellen in voorkomen. In de regel-gebaseerde methode moeten deze elementen worden gesegmenteerd om uiteindelijk de weggelakte stukken over te houden. Figuur 3.8 laat een flowchart zien van de verschillende stappen in deze methode. Deze stappen worden in het volgende hoofdstuk verder toegelicht. Naast het herkennen van weggelakte stukken geven we per pagina een evaluatie over de hoeveelheid weggelakte tekst.



Figuur 3.8: De flowchart van de verschillende stappen in de regel-gebaseerde methode.

HOOFDSTUK 4

Implementatie

Wob-documenten worden opgeslagen in PDF formaat. Voor beide gebruikte methodes is het nodig om de PDF bestanden te converteren naar een foto formaat. Hierbij wordt gekozen voor PNG formaat vanwege de goede kwaliteit van de afbeeldingen. Na het converteren kunnen beide methodes worden toegepast op de pagina's.

4.1 Regel-gebaseerde Methode

Om deze methode zo snel mogelijk te maken zullen we de stappen die hieronder zijn beschreven parallel uitvoeren over meerdere pagina's tegelijk.

4.1.1 Pre-processing

Om weggelakte stukken te herkennen is de eerste stap het gebruik van OCR. Om een zo goed mogelijk resultaat te behalen voor de OCR moeten er eerst een aantal pre-processing technieken worden toegepast op de originele afbeeldingen (Patil e.a. 2022). Eerst wordt de originele afbeelding van kleur naar grijswaarden geconverteerd. Daarna wordt een erosion en dilation achter elkaar toegepast om mogelijke noise te verwijderen. Vervolgens wordt er een blur op de afbeelding toegepast en als laatste wordt de afbeelding getreshold. Er wordt een Gaussian blur gebruikt en voor de treshold wordt gebruik gemaakt van Otsu's Binarization[1] omdat dit na testen het beste resultaat gaf.

4.1.2 Tekst Herkenning

Tesseract OCR

Na pre-processing kan OCR worden toegepast. Na hiermee getest te hebben bleek dat Tesseract een aantal imperfecties kent. Zo worden er soms woorden 'herkend' ter grootte van de hele pagina. Ook worden soms weggelakte stukken als tekst herkend en zijn er woorden die niet worden herkend. Hiervoor worden een aantal instellingen gebruikt om het resultaat te optimaliseren. Allereerst zal de ingestelde taal in Tesseract op Nederlands en Engels worden gezet omdat de dataset alleen Nederlandse documenten bevat waarin geen andere talen dan deze twee worden verwacht. Daarnaast geeft Tesseract een *confidence* per woord welke aangeeft hoe zeker het model is over de herkenning. We gebruiken dit gegeven om slechte voorspellingen van woorden niet te gebruiken. De confidence ligt tussen 0 en 100 en na testen bleek dat een confidence van groter dan 65 het beste resultaat geeft waar weggelakte stukken niet als tekst worden herkend en de meeste echte woorden wel. Daarnaast wordt er een minimale en maximale grootte van bounding boxes om woorden gebruikt om fout herkende stukken te verwijderen.

Na het toepassen van OCR houden we een lijst van bounding boxes over die de plekken van

[1] https://en.wikipedia.org/wiki/Otsu%27s_method

woorden op de afbeelding aangeven. Deze bounding boxes kunnen met witte kleur worden ingevuld om zo alle woorden van de originele afbeelding te verwijderen. Een voorbeeld van dit proces is te zien in Figuur 4.1.

Figuur 4.1: Een document met herkende tekst aangegeven met bounding boxes (links) en het document met de bounding boxes wit gemaakt (rechts).

Morphology

Bockholt, Cavalcanti en Mello (2011) beschrijven een alternatieve methode voor het verwijderen van tekst in een document. Hierbij wordt gebruik gemaakt van een combinatie van erosion en dilation om zo de tekst van een document te verwijderen terwijl grotere elementen zoals afbeeldingen wel op het document blijven. Dit werkt omdat letters in een document vaak klein zijn en daardoor snel worden verwijderd door een erosion operatie. Grotere elementen worden slechts verkleind door de erosion en kunnen door de dilation operatie weer worden vergroot terwijl de letters niet meer terug zullen komen. In ons geval kan met behulp van erosion en dilation tekst worden verwijderd terwijl weggelakte stukken blijven staan.

Deze methode werkt goed op documenten waarin weggelakte stukken volledig zwart of grijs zijn. Er ontstaan echter problemen wanneer er op weggelakte stukken tekst staat geschreven zoals in Figuur 3.1 (a). De erosion operatie neemt dan delen van de weggelakte stukken mee waardoor deze niet meer herkend kunnen worden. Daarnaast is een probleem van deze methode dat grotere tekst zoals headers niet wordt verwijderd waardoor er fouten ontstaan bij het herkennen van contouren.

We kiezen er daarom voor om de Tesseract OCR methode te gebruiken om de tekst te verwijderen omdat dit het beste resultaat geeft voor de verschillende soorten lak.

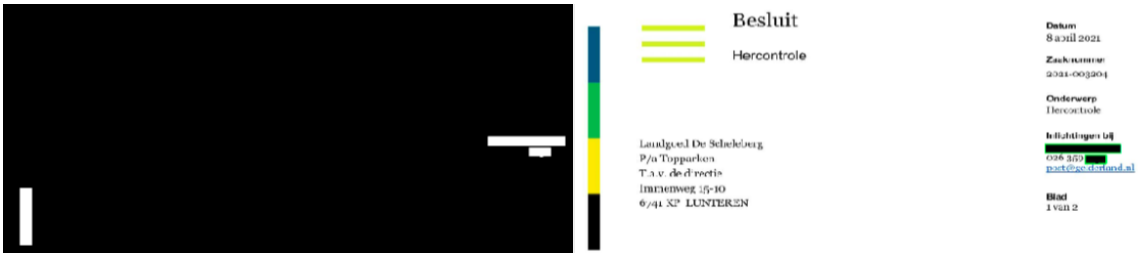
4.1.3 Kleurdetectie

De volgende stap is de detectie van kleuren. Deze stap is belangrijk omdat er na het toepassen van de OCR bijvoorbeeld nog afbeeldingen en logo's overblijven op een document. Deze afbeeldingen en logo's hebben vaak een kleur. Omdat weggelakte stukken tekst over het algemeen zwart of grijs zijn kunnen we hiermee segmenteren. We zoeken eerst zwarte kleuren en maken hiermee een *mask* en vervolgens doen we ditzelfde voor grijstinten door een range te selecteren van alle grijswaarden. Daarna doen we een *bitwise OR* operatie waardoor we een mask overhouden met alle mogelijke weggelakte gebieden in het wit en de rest van de pagina in het zwart. Figuur 4.2 geeft de mask die overblijft na deze stap toe te passen op de rechter afbeelding in Figuur 4.1.

Figuur 4.2: De mask die overblijft na het segmenteren op kleur voor hetzelfde document als in Figuur 4.1.

4.1.4 Vinden van Contouren

De laatste stap is het verwijderen van noise door erosion en dilation toe te passen. Wat over blijft is een zwarte afbeelding met witte contouren. Weggelakte stukken zijn over het algemeen vierkant of groter in breedte dan in hoogte. Daarom zullen we alleen deze contouren selecteren. Tenslotte kunnen dan de bounding boxes van deze contouren worden bepaald met OpenCV om zo de plek van het weggelakte stuk te verkrijgen. In Figuur 4.3 is te zien welke contouren overblijven en welke gebieden uiteindelijk worden geselecteerd.

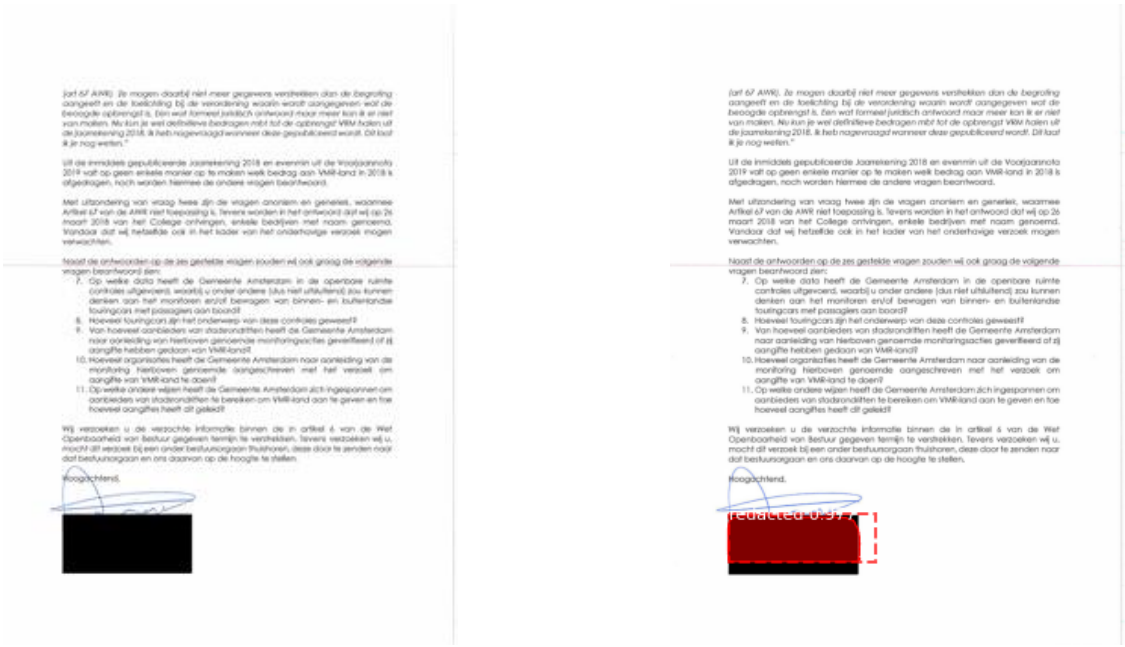


Figuur 4.3: De mask na het verwijderen van noise met daarop de mogelijke weggelakte stukken (links) en de uiteindelijk gekozen stukken omlijnd in groen op de originele afbeelding (rechts).

4.2 Machine Learning Methode

Een andere methode om weggelakte tekst te herkennen in een afbeelding is door gebruik te maken van de Mask R-CNN. Abdulla (2017) beschrijft een implementatie van de Mask R-CNN die we zullen gebruiken voor onze dataset. Hiervoor gebruiken we een training set van 170 afbeeldingen en een validatie set van 25 afbeeldingen. We kunnen gebruik maken van deze kleine dataset doordat we training van het model starten met gewichten die voorgetraind zijn op de MS COCO dataset[2]. We gebruiken 100 *steps per epoch* en laten alle andere instellingen op de standaardwaarden. We trainen het model voor 10 epochs. Een voorbeeld van een voorspelling van deze methode is te zien in Figuur 4.4.

[2]https://cocodataset.org/#home



Figuur 4.4: De originele afbeelding (links) en de voorspelling van het Mask R-CNN model (rechts). De IoU waarde is hier 0.76 en telt dus mee als true positive.

HOOFDSTUK 5

Experimenten

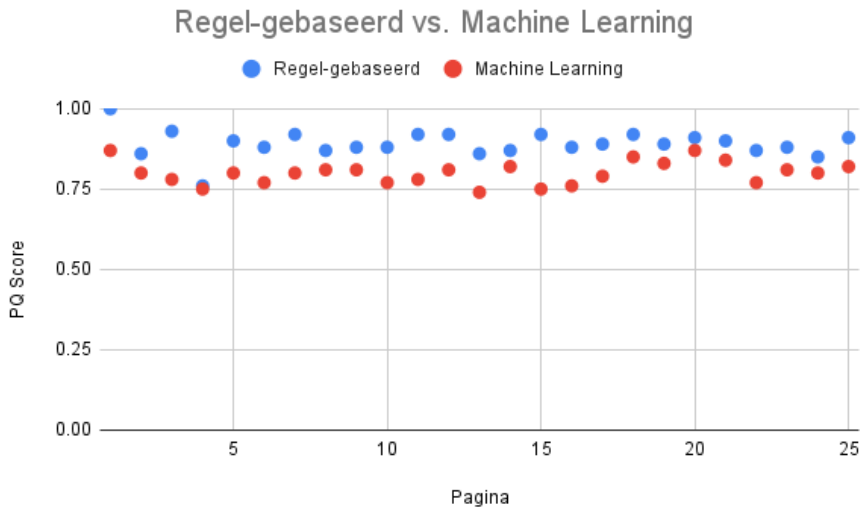
Dit hoofdstuk beschrijft de experimenten met bijbehorende resultaten van de methodes.

5.1 Vergelijking van Beide Methodes

We vergelijken eerst beide methodes door ze te testen op dezelfde pagina's. Voor het trainen van de Mask R-CNN is het nodig om onze dataset op te splitsen in een train- en validatieset. De gebruikte dataset bevat alleen zwart weggelakte stukken. Voor de vergelijking tussen de twee methodes zullen we beide methodes testen op de validatie set zodat op dezelfde pagina's getest wordt. De validatie set bestaat uit 25 afbeeldingen. We testen beide implementaties eerst op dezelfde CPU zonder parallellisaties. We vergelijken de gemiddelde uitvoertijd per pagina en de SQ, RQ en PQ scores tussen de methodes. De resultaten zijn te zien in Tabel 5.1 en Figuur 5.1.

Tabel 5.1: Vergelijking van de gemiddelde accuraatheid en uitvoertijd per pagina van de Regel-gebaseerde en de Machine Learning methodes getest op dezelfde CPU. We testen op 25 pagina's.

Methode	Regel-gebaseerd	Machine Learning
Gemiddelde SQ Score	0.93	0.91
Gemiddelde RQ Score	0.95	0.88
Gemiddelde PQ Score	0.88	0.80
Gemiddelde Uitvoertijd per Pagina	1.92s	6.68s



Figuur 5.1: PQ scores van de Regel-gebaseerde methode tegenover de Machine Learning methode over 25 pagina's.

5.2 Vergelijking van de Uitvoertijd

De Regel-gebaseerde methode is geoptimaliseerd om parallel over meerdere pagina's te draaien. Op de door ons geteste CPU gaf het draaien op twee pagina's tegelijk de grootste tijdwinst ten opzichte van de sequentiële uitvoering van het programma. We testen op dezelfde 25 pagina's als in experiment 5.1. De resultaten zijn te zien in Tabel 5.2.

De Machine Learning methode is geoptimaliseerd om te draaien op een GPU. We testen de implementatie op de Tesla T4 GPU van Google Colab[1] met dezelfde 25 pagina's als in experiment 5.1 en laten de resultaten zien in Tabel 5.3.

Tabel 5.2: Vergelijking van de gemiddelde uitvoertijd per pagina van sequentiële en parallelle uitvoering van de Regel-gebaseerde methode. Bij de parallelle uitvoering worden twee pagina's tegelijk verwerkt.

Methode	Regel-gebaseerd sequentieel	Regel-gebaseerd parallel
Gemiddelde Uitvoertijd per Pagina	1.92s	1.45s

Tabel 5.3: Vergelijking van de gemiddelde uitvoertijd per pagina van de Machine Learning methode op de Intel® Core™ i5-8265U CPU en op de Tesla T4 GPU.

Methode	Machine Learning CPU	Machine Learning GPU
Gemiddelde Uitvoertijd per Pagina	6.68s	0.81s

[1] https://colab.research.google.com/

5.3 Regel-gebaseerde Methode Getest op de Verschillende Datasets

Bij de Regel-gebaseerde methode is het niet nodig om de datasets op te splitsen in een train- en validatieset. We testen deze methode daarom op meer pagina's om een beter beeld te geven van de accuraatheid van het model op verschillende soorten weggelakte stukken.

We testen op de Zylab, Corpus1 en Corpus2 datasets die beschreven zijn in sectie 3.3. De resultaten zijn te zien in Tabel 5.4. De laatste kolom geeft de scores wanneer getest wordt op alle pagina's in de drie datasets.

Tabel 5.4: Gemiddelde accuraatheid van de Regel-gebaseerde methode getest op de Zylab, Corpus1 en Corpus2 datasets.

Dataset	Zylab (N=46)	Corpus1 (N=55)	Corpus2 (N=45)	Totaal (N=146)
Gemiddelde SQ Score	0.92	0.88	0.95	0.92
Gemiddelde RQ Score	0.81	0.89	0.95	0.89
Gemiddelde PQ Score	0.74	0.82	0.90	0.83

Er vallen een aantal dingen op aan deze resultaten. Bij de Zylab dataset is de SQ score hoog terwijl de RQ score lager is. Dit betekent dat als een weggelakt stuk goed wordt gedetecteerd (IoU > 0.5), dat de bounding box erg nauwkeurig is. De lage RQ geeft echter aan dat niet alle stukken goed worden herkend. Er zijn dus meer false positives en/of false negatives dan bij de andere datasets. Dit heeft met name te maken met het feit dat aangrenzende weggelakte stukken niet goed van elkaar worden onderscheiden en dus worden herkend als één stuk.

De Corpus1 dataset heeft juist een iets hogere RQ dan SQ score. Dit betekent dat hier dus wel het aantal weggelakte stukken beter wordt herkend maar de bounding box om de stukken iets minder accuraat is dan bij de andere datasets.

De Corpus2 dataset heeft zowel een hoge SQ als RQ score wat laat zien dat bij deze dataset bijna alle weggelakte stukken worden herkend en daarbij ook nog de bounding boxes goed om de stukken wordt geplaatst.

5.4 Testen van de Evaluatiematen

Het programma geeft een aantal evaluatiematen als output welke zijn beschreven in Sectie 3.5.2. Het geeft een voorspelling van:

1. Het aantal weggelakte stukken
2. Het aantal weggelakte karakters
3. Het percentage van de hoeveelheid weggelakte tekst ten opzichte van het tekstvlak
4. Het percentage van de hoeveelheid weggelakte tekst ten opzichte van de oppervlakte van de bounding boxes om de karakters

Om te testen of de voorspellingen accuraat zijn maken we gebruik van de RMSE. Dit laat zien in welke mate er verschil is tussen de daadwerkelijke waardes en de voorspelde waardes. We testen op 5 willekeurig gekozen pagina's in de datasets. De resultaten zijn gegeven in Tabel 5.5. De lijst hierboven geeft aan om welke evaluatiemaat het gaat.

Tabel 5.5: RMSE van de verschillende evaluatiematen.

Evaluatiemaat	RMSE
1	2.24
2	22.94
3	0.71
4	2.20

HOOFDSTUK 6

Discussie & Conclusie

In dit hoofdstuk zullen de resultaten van de experimenten uit hoofdstuk 5 besproken worden. Daarnaast worden er een aantal problemen besproken en zal er een suggestie voor mogelijk toekomstig onderzoek worden gegeven. We eindigen met een samenvatting en conclusie van het onderzoek.

6.1 Discussie

Uit experiment 5.1 blijkt dat de Regel-gebaseerde methode een hogere PQ score haalt dan de Machine Learning methode. Daarnaast wordt aangetoond dat de uitvoertijd van de Regel-gebaseerde methode korter is dan die van de Machine Learning methode op dezefde CPU. Dit toont aan dat de Regel-gebaseerde methode een meer geschikte methode is voor het efficiënt vinden van weggelakte stukken in Wob-documenten.

De experimenten in sectie 5.2 laten zien dat beide methode getoptimaliseerd kunnen worden om zo de uitvoertijd te verbeteren. Hieruit kan worden opgemaakt dat de Machine Learning methode een goede methode kan zijn als gebruik wordt gemaakt van een GPU. De uitvoertijd van de Regel-gebaseerde methode kan verbeterd worden door deze parallel te draaien over meerdere pagina's tegelijk. Deze methode is echter niet geoptimaliseerd voor een GPU. Daarom kan er geen vergelijking worden gemaakt tussen beide methodes op dezelfde GPU. Mogelijk zou de Regel-gebaseerde methode op een CPU met veel cores kunnen worden gedraaid waarmee de snellere tijd van de Machine Learning methode wellicht kan worden benaderd.

Sectie 5.3 laat de resultaten van de Regel-gebaseerde methode zien op de verschillende datasets. Hier is een duidelijk verschil te zien in de PQ score tussen de datasets. De Corpus2 dataset heeft het beste resultaat met een gemiddelde PQ score van 0.90. Dit komt doordat deze dataset de meest simpele pagina's bevat, met alleen zwart weggelakte stukken. De Zylab dataset behaalt de laagste gemiddelde PQ score. Dit heeft met name te maken met de lagere gemiddelde RQ score voor deze dataset. Dit komt voornamelijk door het feit dat deze dataset veel pagina's bevat waar meerdere weggelakte stukken aan elkaar grenzen. Deze stukken worden dan niet van elkaar onderscheiden waardoor de RQ lager is. De Corpus1 dataset bevat een combinatie van verschillende soorten weggelakte stukken en het feit dat deze PQ score tussen die van de Zylab en Corpus2 dataset ligt was daarom ook verwacht.

6.2 Problemen

Beide methodes hebben als probleem dat zwarte balken met tekst erin bij bijvoorbeeld een tabel worden herkend als een weggelakt stuk. Dit is een heel lastig onderscheid om visueel gezien te maken en zou mogelijk kunnen worden opgelost door een analyse te maken van de tekst die op de balk staat geschreven. Vaak bevatten weggelakte stukken een code of zit het woord 'Wob' in de

zin terwijl dit bij een tabel niet het geval is. Op die manier zou het mogelijk zijn een onderscheid te maken. Daarnaast hadden beide methodes moeite met het herkennen van stukken die met de hand zijn weggestreept omdat deze stukken niet een solide blok zijn maar juist meerdere strepen. Deze strepen worden dan als aparte stukken herkend en niet als een geheel.

Een probleem van de Regel-gebaseerde methode is dat er geen onderscheid wordt gemaakt tussen weggelakte stukken die aan elkaar grenzen zoals veel het geval is in de Zylab dataset. Hierdoor wordt het totale gebied wel juist herkend maar de verschillende instanties niet. Dit is wel van invloed op de PQ score maar maakt eigenlijkt niet uit voor de inschatting van het aantal weggelakte karakters omdat de oppervlakte van het weggelakte stuk wel goed wordt herkend. Een oplossing kan zijn om de ground truth minder streng te maken door aangrenzende stukken toch als één stuk te tellen. De Machine Learning implementatie kan wel worden getraind om de verschillende instanties van de Zylab dataset te onderscheiden.

De Machine Learning implementatie maakt soms voor één weggelakt stuk twee voorspellingen en maakt soms helemaal geen voorspelling voor een weggelakt stuk. Dit kan te maken hebben met de kleine dataset die gebruikt is voor het trainen van dit model. Daarnaast is het trainen van dit model en het maken van voorspellingen langzaam op een CPU.

6.3 Toekomstig Werk

Een interessant vervolgonderzoek op deze scriptie zou zijn om de Machine Learning implementatie te trainen met een grotere dataset. De verwachting is dat met meer training dit model betere resultaten zal behalen. Door de opkomst van automatische laksoftware zal er in de toekomst minder met de hand moeten worden geannoteerd waardoor sneller aan trainingsvoorbeelden kan worden gekomen. Daarnaast is de verwachting dat automatische laksoftware ervoor zorgt dat lak in documenten minder van elkaar zullen verschillen waardoor trainen ook effectiever wordt. Ook zou in de Machine Learning methode kunnen worden getest of het gebruik van minder lagen in de ResNet architectuur de training- en uitvoertijd van de Mask R-CNN verbetert zonder dat de accuraatheid (sterk) vermindert.

Daarnaast zou een machine learning model zoals YOLO[1] kunnen worden getest. Dit model is een *single-stage detector* en doet de classificatie en bounding box regressie op hetzelfde moment (Redmon en Farhadi 2018). Bij Mask R-CNN gebeurt dit in twee stappen. Dit model zou dus ook een lagere uitvoertijd kunnen hebben.

Ook zou het interessant zijn om de Regel-gebaseerde methode te optimaliseren voor een GPU en hierop te testen. Zo kan een goede vergelijking worden gemaakt van de uitvoertijden van de twee methodes op eenzelfde GPU. Daarnaast kan de Regel-gebaseerde methode worden getest op een CPU met meer cores dan nu is getest.

6.4 Conclusie

In dit onderzoek is gezocht naar geschikte methodes om weggelakte stukken in Wob-documenten te herkennen. Daartoe is een methode gemaakt aan de hand van Pytesseract en een aantal OpenCV operaties. Deze methode is vergeleken met de Mask R-CNN, een state-of-the-art methode in object detectie en afbeelding segmentatie.

Uit dit onderzoek is gebleken dat beide methodes nauwkeurig weggelakte stukken in Wob-documenten kunnen herkennen. De Machine Learning methode is met name effectief op een GPU terwijl de Regel-gebaseerde methode ook goed op een CPU gedraaid kan worden. Wanneer beide methodes op dezelfde pagina's worden getest, behaalt de Regel-gebaseerde methode een gemiddelde PQ score van 0.88 tegenover een gemiddelde PQ score van 0.80 voor de Machine

[1] https://pjreddie.com/darknet/yolo/

Learning methode. Daarnaast is de uitvoertijd van de Regel-gebaseerde methode ook korter dan de Machine Learning methode wanneer beide op dezelfde CPU worden getest. Na getest te hebben op meerdere datasets bleek de Regel-gebaseerde methode een gemiddelde PQ score van 0.83 te halen. De Regel-gebaseerde methode is daarom de meer geschikte methode voor het vinden van weggelakte stukken in Wob-documenten.

Bibliografie

Abdulla, Waleed (2017). *Mask R-CNN for object detection and instance segmentation on Keras and TensorFlow*. https://github.com/matterport/Mask_RCNN.

Bland, Maxwell, Anushya Iyer en Kirill Levchenko (2022). *Story Beyond the Eye: Glyph Positions Break PDF Text Redaction*. DOI: 10.48550/ARXIV.2206.02285. URL: https://arxiv.org/abs/2206.02285.

Bockholt, Tiago C., George D. C. Cavalcanti en Carlos A. B. Mello (2011). "Document image retrieval with morphology-based segmentation and features combination". In: *Document Recognition and Retrieval XVIII*. Red. door Gady Agam en Christian Viard-Gaudin. Deel 7874. International Society for Optics en Photonics. SPIE, p. 787415. DOI: 10.1117/12.876727. URL: https://doi.org/10.1117/12.876727.

Hafiz, Abdul Mueed en Ghulam Mohiuddin Bhat (2020). "A survey on instance segmentation: State of the art". In: *International Journal of Multimedia Information Retrieval* 9.3, p. 171–189. DOI: 10.1007/s13735-020-00195-x.

He, Kaiming e.a. (2017). *Mask R-CNN*. DOI: 10.48550/ARXIV.1703.06870. URL: https://arxiv.org/abs/1703.06870.

Kirillov, Alexander e.a. (2018). "Panoptic Segmentation". In: *CoRR* abs/1801.00868. arXiv: 1801.00868. URL: http://arxiv.org/abs/1801.00868.

Mo, Yujian e.a. (2022). "Review the state-of-the-art technologies of semantic segmentation based on deep learning". In: *Neurocomputing* 493, p. 626–646. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2022.01.005. URL: https://www.sciencedirect.com/science/article/pii/S0925231222000054.

Mohd Ali, Nursabillilah (sep 2013). "Performance Comparison between RGB and HSV Color Segmentations for Road Signs Detection". In: *Applied Mechanics and Materials* 393. DOI: 10.4028/www.scientific.net/AMM.393.550.

Patil, Shruti e.a. (okt 2022). "Enhancing Optical Character Recognition on Images with Mixed Text Using Semantic Segmentation". In: *Journal of Sensor and Actuator Networks* 11, p. 63. DOI: 10.3390/jsan11040063.

Redmon, Joseph en Ali Farhadi (2018). *YOLOv3: An Incremental Improvement*. DOI: 10.48550/ARXIV.1804.02767. URL: https://arxiv.org/abs/1804.02767.

Regt, Koen de en Roland Strijker (jan 2021). "Ministeries overtreden op grote schaal eigen regels bij vrijgeven documenten". In: *RTL Nieuws*. URL: https://www.rtlnieuws.nl/nieuws/artikel/5208668/wob-transparantie-kabinet-ministerie-toeslagenaffaire-documenten-informatie.

Soille, Pierre (mrt 2013). *Morphological image analysis*. en. 2de ed. Berlin, Germany: Springer.

Thakare, Sahil e.a. (2018). "Document Segmentation and Language Translation Using Tesseract-OCR". In: *2018 IEEE 13th International Conference on Industrial and Information Systems (ICIIS)*, p. 148–151. DOI: 10.1109/ICIINFS.2018.8721372.

Yang, Xiao e.a. (2017). *Learning to Extract Semantic Structure from Documents Using Multimodal Fully Convolutional Neural Network*. DOI: 10.48550/ARXIV.1706.02337. URL: https://arxiv.org/abs/1706.02337.

Zanaty, E. (jan 2016). "Medical Image Segmentation Techniques: An Overview". In: *International Journal of informatics and medical data processing (JIMDP)* 1, p. 1.