Onderzoeksverslag IPASS

Door: Marc Zoomers

Juni 2019

Inhoudsopgave

Inleiding

Voor mijn Individual Propedeuse Assignment voor HBO-ICT Applied Artificial Intelligence zal ik een onderzoek doen naar OCR's in Python. OCR staat voor Optical Character Recognition, dit is de techniek waarbij tekst uit een foto van tekst door middel van patroonherkenning herkend en opgeslagen wordt. De eerste vormen van OCR komen uit de jaren 20 van de 20e eeuw, maar pas in de jaren 70 werd te toepassing van OCR populair.

Het beroepsproduct van mijn IPASS zal een library worden die de sterke kanten van verschillende OCR's combineert. Bij deze library zal dit verslag ook geleverd worden.

Uitleg van het project

Voor het IPASS project zal ik een library ontwikkelen die 3 Python OCR-libraries combineert. Deze library zal fungeren als een wrapper voor deze 3 libraries, die het gebruik van de libraries moet vereenvoudigen. Deze libraries hebben ieder hun eigen vaardigheden waarvoor ze gebruikt zullen worden. In de lijst van de libraries zal er uitgelegd worden wat de specialiteit is per library.

Lijst van libraries

OPEN-ALPR

Beschrijving

OPEN-ALPR is een library ontwikkelt door het bedrijf openALPR[1]. OPEN-ALPR is geschreven in C++, maar kan ook worden gebruikt in de talen C#, Java, Node.js, Go en Python. Open-ALPR is gebouwd voor het herkennen van kentekenplaten uit natuurlijke foto's. OpenALPR's eerste release was in 2014 en was gratis verkrijgbaar tot eind 2015. De OPEN-ALPR library zal gebruikt worden voor kentekenplaat herkenning.

Algoritme

Voor detectie van kentekenplaten in openALPR wordt gebruikt gemaakt van het LBP (local binary pattern) algoritme[2]. Dit algoritme wordt vaak gebruikt voor gezichtsherkenning. Door een kleine aanpassing in de constanten van het algoritme kan het ingezet worden voor het vinden van mogelijke kentekenplaten op de foto's.

Methodes

Voor de binarisatie van foto's wordt gebruikt gemaakt van de Wolf-Jolien methode en Sauvola methode[3]. Dit resulteert in meerdere binaire foto's voor de beste kans van het vinden van kentekenplaten.

PYOCR

Beschrijving

PYOCR[4] is een Python library voor het eerst uitgebracht in mei 2013. Het team van ontwikkelaars bestaat uit 21 mensen en werken er nog regelmatig aan (laatste release was in mei 2019). PYOCR zal gebruikt worden voor het herkennen van tekst in natuurlijke foto's. In deze toepassing zal de PYOCR klasse een tijdelijk jpg bestand maken dat een hoog contrast masker is van de originele natuurlijke foto.

Algoritme

PYOCR maakt gebruikt van een "zelfgemaakt"-algoritme, dat goed van toepassing is bij tekst herkennen vanaf foto's. Hiervoor is een bijlage te vinden in de map bijlagen onder het bestand "Detecting Text in Natural Scenes with Stroke Width Transform". Deze methode van tekst herkenning is uitzonderlijk goed voor de toepassing in natuurlijke foto's en om daar tekst uit te herkennen. Eigenlijk is dit algoritme ook gelijk de methode van het binariseren van de foto, dus zal ik de methode van PYOCR niet uitleggen.

PyTesseract

Beschrijving

PyTesseract is een Python interface voor verschillende Python pakketten: tesseract-ocr, ps2text, sox etc.. PyTesseract is gespecialiseerd in het uitlezen van documenten, maar ondersteund ook verschillende foto formaten en kan zelfs tekst uit geluidsbestanden halen[5]. PyTesseract's eerst release was in juli 2014, maar is sinds juni 2017 niet meer geüpdatet. Voor het onderzoek zal ik alleen gebruik maken van de .jpg functies van PyTesseract, deze functie maakt gebruik van Google's tesseract-ocr. Voor een gedetaileerde uitleg van het fungeren van tesseract-ocr zie "AnoverviewofTesseractOCREngine.pdf" in de map Bijlagen.

Algoritme

Google's tesseract-ocr maakt gebruik van Otsu's algoritme en Otsu's methode. Het algoritme van Otsu staat ook wel bekend als het Firefly-algorithm[6]. Het Firefly algoritme is geïnspireerd door het knipperende gedrag van vuurvliegjes. De lichtintensiteit wordt geassocieerd met de aantrekkingskracht van een vuurvliegje, de mogelijkheid om groepen te verdelen in kleine groepen en elke subgroep zwerm rond lokale eigenschappen te categoriseren. Daarom vuurvlieg algoritme bijzonder geschikt voor multimodale optimalisatieproblemen.

Methodes

Voor de binarisatie van een foto wordt gebruik gemaakt van Otsu's methode. Otsu's formule is als volgt: $\sigma_w^2(t) = \omega_0(t) \cdot \sigma_0^2(t) + \omega_1(t) \cdot \sigma_1^2(t)$.

In deze formule zijn ω_0 en ω_1 de mogelijkheid van de 2 kleuren classes onderscheiden door drempelwaarde *t.* σ_0^2 en σ_1^2 zijn de variaties tussen de 2 kleuren classes.

Problemen en keuzes

Tijdens de ontwikkeling van dit project ben ik tegen een aantal kleine problemen aangelopen. Mijn originele doel van het IPASS was om een onderzoek te doen naar performance van verschillende OCR's. Dit bleek niet te voldoen aan de eisen van het AAI gedeelte van het IPASS, dus ben ik overgegaan op het ontwikkelen van een library/app. Mijn initiële doel was om 5 libraries te combineren en deze in 1 app te zetten, maar al gauw bleek dat à 99% van alle Python OCR-libraries op de methode en het algoritme van Google's tesseract OCR liepen. Hierom ben ik van 5 naar 3 libraries gegaan die ieder hun eigen methodes en eigen algoritme hanteerden. Mijn keuze voor libraries kwam uit op: PYOCR, OPEN-ALPR en Textract. Voor de laatst genoemde library bleek geen ondersteuning te zijn voor Python 3.7, en ben overgestapt op PyTesseract, omdat deze library zo goed als identiek is aan Textract.

Eindproduct

Het eindproduct van het IPASS is een library geworden, die fungeert als een wrapper van 3 Python OCR-libraries; OPEN-ALPR, PYOCR en PyTesseract. Bij deze library heb ik ook een GUI gebouwd voor gebruik en test mogelijkheden met deze library. Deze GUI laat in hoeverre het mogelijk is alle potentie van de library zien.

Bronnenlijst en verwijzingen

1. openalpr/openalpr. (2014, 19 januari). Geraadpleegd 19 juni 2019, van https://github.com/openalpr/openalpr
2. do Prado, K. S. (2017, 10 november). *Face Recognition: Understanding LBPH Algorithm*. Geraadpleegd 19 juni 2019, van https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b
3. Bataineh, B., Abdullah, S. N. H. S., & Omar, K. (2011). An adaptive local binarization method for document images based on a novel thresholding method and dynamic windows. *Pattern Recognition Letters*, *32*(14), 1805–1813. https://doi.org/10.1016/j.patrec.2011.08.001
4. *pyocr on Pypi*. (2019, 22 juni). Geraadpleegd 22 juni 2019, van https://libraries.io/pypi/pyocr
5. *textract — textract 1.6.1 documentation*. (2017, 17 juli). Geraadpleegd 22 juni 2019, van https://textract.readthedocs.io/en/stable/
6. Sri Madhava Raja, N., Rajinikanth, V., & Latha, K. (2014). Otsu Based Optimal Multilevel Image Thresholding Using Firefly Algorithm. *Modelling and Simulation in Engineering*, *2014*, 1–17. https://doi.org/10.1155/2014/794574