ABSTRAK

Kelvin Wyeth (01082190015)

IMAGE-TO-TEXT MENGGUNAKAN TESSERACT OCR

(xiii + 107 halaman: 69 gambar; 3 tabel)

Di era modern saat ini, informasi dapat diakses oleh hampir semua orang, baik itu *hardcopy* maupun *softcopy*. Namun saat ini, ada kebutuhan digitalisasi teks fisik, seperti buku atau dokumen, yang berarti kebutuhan untuk mengubah teks dari gambar menjadi teks, dari *hardcopy* menjadi *softcopy*. File gambar dan PDF, yang mudah diperoleh dari kamera dan *scanner*, sebagian besar berukuran besar, tetapi *file* teks jauh lebih ringan. Mengubah gambar menjadi teks membutuhkan *Optical Character Recognition*, atau disingkat OCR. Solusi yang diusulkan adalah program yang ditulis dalam bahasa pemrograman *Python* dalam *file Jupyter Notebook.*

Program Optical Character Recognition berbasis *Deep Learning* dengan sistem yang mengikuti langkah-langkah *Optical Character Recognition*: *pre-processing*, *segmentation*, *feature extraction*, *recognition*, dan *post-processing*. Tahap *pre-processing* menggunakan delapan metode *pre-processing OpenCV*, yaitu *Grayscale, Noise Removal with Median Filter, Gaussian Filter, Bilateral Filter, Global Thresholding, Otsu Thresholding, Otsu Thresholding with Gaussian Filtering,* dan *Canny Edge Detection*. Tahap *segmentation* dilakukan melalui konfigurasi *Tesseract*, yang menggunakan OSD (Orientation and Script Detection). Tahap *extraction* dan *recognition* menggunakan konfigurasi LSTM *Tesseract*, yang menggunakan JST LSTM, mengambil karakter dari segmentasi dan memrosesnya dengan model bahasa dan data *training* yang terdapat dalam *folder tessdata-best*, yang berisi model bahasa dan data *training* yang tersedia untuk *Tesseract* pada waktu penulisan. Itulah tahap *post-processing*.

Program diuji terhadap empat gambar halaman buku dengan teks bahasa Indonesia, dipindai dengan aplikasi *Adobe Scan* pada *smartphone*, yang berfungsi sebagai *input* untuk OCR. Keempat gambar memiliki detail yang bervariasi, dengan gambar pertama memiliki teks dengan *bullet-point*, yang kedua memiliki *font* dengan ukuran yang bervariasi, yang ketiga memiliki teks yang di-*wrap* sekitar gambar, dan yang keempat memiliki sebuah tabel.

Dari hasil teks dan perhitungan accuracy dan error rate dari setiap hasil OCR, hasil untuk mendeteksi gambar pertama sudah baik, dengan tujuh dari sembilan hasil dengan tingkat kesalahan tidak lebih dari 1%, selain *Median Filter* dan gambar *Canny Edge Detection*. Namun, gambar kedua memiliki hasil yang lebih bervariasi, dengan gambar *Global Thresholding* memiliki tingkat kesalahan 24,51%, tertinggi dari ke-36 hasil, dan lainnya memiliki *error rate* antara 1 – 7%. Hasil untuk gambar ketiga memiliki *error rate* sekitar 2 – 10%, dan yang keempat memiliki *error rate* sekitar 1 – 8%. Analisis lebih lanjut memberi kesimpulan bahwa isi dari halaman, metode *pre-processing*, dan kualitas gambar memiliki dampak terbesar pada tingkat kesalahan dan hasil OCR. Secara keseluruhan, LSTM yang bekerja bersama data *tessdata-best* berfungsi sebagai metode OCR yang akurat, meskipun bergantung kepada kualitas gambar, metode *pre-processing*, dan isi halaman.

Referensi: 19 (2015-2022)

ABSTRACT

Kelvin Wyeth (01082190015)

IMAGE-TO-TEXT USING TESSERACT OCR

(xiii + 107 pages: 69 figures; 3 tables)

In today's modern age, information is accessible to almost everyone, whether it's hardcopy or softcopy. But today, there's a need for the digitalization of physical texts, such as books or documents, which means a need to convert texts from images to text, from hardcopy to softcopy. Images and PDF files, which were easily obtained from cameras and scanners, are mostly huge, but text files are much lighter. Converting images to texts calls for the need for Optical Character Recognition, or OCR for short. The proposed solution is a program written in Python programming language in a Jupyter Notebook file.

A Deep Learning-based Optical Character Recognition program, with a system adhering to the phases of Optical Character Recognition: pre-processing, segmentation, feature extraction, recognition, and post-processing. The pre-processing phase uses eight OpenCV pre-processing methods, which are Grayscale, Noise Removal with Median Filter, Gaussian Filter, Bilateral Filter, Global Thresholding, Otsu Thresholding, Otsu Thresholding with Gaussian Filtering, and Canny Edge Detection. The segmentation phase was next, done through the Tesseract configuration, which utilized orientation and script detection. Feature extraction and recognition utilized Tesseract's LSTM configuration, which uses the LSTM Neural Network, capturing segmented characters. The results were then processed with trained data and language models contained in the tessdata-best folder, which contained the best-trained data and language models Tesseract has to offer by the time of writing. It served as the post-processing phase.

The program was tested on four images of pages of a book with Indonesian text, scanned with the Adobe Scan application on a smartphone, which serves as inputs for the OCR. The images were to be of varying details, with the first image having texts with bullet points, the second having fonts of varying sizes, the third having texts wrapped around an image, and the fourth having a table.

From the visualization of the text and supported by the calculations of the accuracies and the error rates of each OCR result, the results to detect the first image were good, with seven out of nine results with error rates not exceeding 1%, aside from the Median Filter image and Canny Edge Detection image. The second image, however, had more varied results, with the Global Thresholding image having an error rate of 24.51%, the highest of all 36 results, and the others ranged between 1 – 7%. The results for the third image had error rates ranging between 2 – 10%, while the fourth ones ranged between 1 – 8%. The further analysis gave us the conclusion that the page's contents, the pre-processing methods, and image quality had the largest impact on the error rate and the OCR results. Overall, the LSTM working alongside the tessdata-best data served as an accurate method of OCR and would yield better results, which depends on the pre-processing methods, image quality, and the page's contents.

References: 19 (2015-2022)