Perbandingan *Character Recognition* dan *Text Recognition* Menggunakan *Extended* MNIST dan IAM *Database* dan *Tesseract* pada Tulisan Tangan Ijazah

Made Yoga Mahardika, Kartika Gunadi, Alexander Setiawan
Program Studi Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) – 8417658
yeogaa02@gmail.com, kgunadi@petra.ac.id, alex@petra.ac.id

ABSTRAK

Permasalahan pada tulisan tangan adalah bagaimana sebuah teknik dapat mengenali berbagai jenis tulisan dengan bentuk beragam. Berbeda dengan huruf komputer yang konsisten, tulisan tangan setiap manusia memiliki keunikan dalam bentuk dan konsistensi penulisan. Permasalahan tersebut dapat ditemukan pada dokumen ijazah. Pengisian data ijazah masih menggunakan tulisan tangan.

Segmentasi lokasi data menggunakan *run length smoothing algorithm* dengan titik sebagai fitur segmentasi. Teknik *handwritten text recognition* (HTR) membutuhkan data disegmentasi menjadi gambar kata. Teknik *handwritten character recognition* (HCR) membutuhkan data disegmentasi menjadi gambar karakter. HCR menggunakan model LeNet5 dengan *EMNIST dataset*. HTR menggunakan *tesseract* dan *convolutional recurrent neural network* dengan *IAM database.*

Pengujian pada 10 sample gambar *scan* ijazah, segmentasi memperoleh akurasi rata-rata 95.1%. Teknik HCR mengalami kegagalan di proses segmentasi huruf pada *cursive handwriting*. Teknik terbaik adalah HTR dengan *tesseract tool* berhasil mendapatkan *word accuracy* diatas 69% di uji pada 5 sample gambar scan ijazah, 15 *data field* keseluruhan.

Kata Kunci: *run length smoothing algorithm, Extended MNIST, IAM database, handwritten character recognition, handwritten text recognition, convolutional recurrent neural network, tesseract,* s*egmentasi*, ijazah.

ABSTRACT

The problem with handwriting is how a technique can recognize various types of writing in various forms. Different from computer letters that consistent, each human's handwriting is unique in the form and consistency. These problems can be found in ijazah *documents where the data is handwriting.*

Data location segmentation uses run length smoothing algorithm with dots as segmentation features. Handwritten text recognition (HTR) technique requires data segmented into words. Handwritten character recognition (HCR) technique requires data segmented into characters. HCR uses the LeNet5 model with the EMNIST dataset. HTR uses tesseract tool and convolutional recurrent neural networks with the IAM database.

Experiment on 10 samples of scan images, segmentation obtained an average accuracy of 95.6%. The HCR technique failed in the letter segmentation process in cursive handwriting. The best technique is the HTR with tesseract tool managed to get word accuracy above 69% tested on 5 scan samples, 15 data fields.

Keywords: *run length smoothing algorithm, Extended MNIST, IAM database, handwritten character recognition, handwritten text recognition, convolutional recurrent neural network, tesseract,* s*egmentasi*, ijazah..

1. PENDAHULUAN

Penelitian mengenai pengenalan tulisan tangan khususnya untuk angka dan huruf latin, merupakan salah satu bahasan dalam pengembangan teknik pengenalan pola yang masih berkembang saat ini [6]. Teknik pengenalan tulisan dapat dibagi menjadi 2, yaitu *character recognition* dan *text recognition*. Proses pengenalan tulisan terdiri dari beberapa mekanisme, yaitu mekanisme ekstraksi fitur atau ciri, dan mekanisme klasifikasi. Kedua teknik tersebut memiliki pendekatan mekanisme yang berbeda dalam melakukan pengenalan tulisan, terutama pada mekanisme klasifikasi.

Permasalahan yang muncul dalam melakukan proses pengenalan huruf komputer adalah bagaimana sebuah teknik pengenalan dapat mengenali berbagai jenis tulisan dengan ukuran, ketebalan, dan bentuk yang berbeda [8]. Permasalahan ini dapat ditemukan terutama pada kasus tulisan tangan. Berbeda dengan huruf komputer yang konsisten dari cara penulisan, cara penulisan tulisan tangan setiap manusia memiliki keunikan masing-masing. Permasalahan tersebut dapat ditemukan pada contoh kasus dokumen ijazah yang masih menggunakan tulisan tangan pada pengisian data pribadi pemiliknya.

Permasalahan tulisan tangan tersebut dapat diselesaikan dengan menerapkan teknik-teknik pengenalan tulisan. Teknik tersebut adalah *character recognition* dan *text recognition*. Teknik *character recognition* dengan menggunakan dataset *Extended* MNIST. Dataset ini adalah varian dataset NIST lengkap, yang disebut *Extended* MNIST (EMNIST), yang mengikuti paradigma konversi yang sama yang digunakan untuk membuat dataset MNIST [3]. Teknik *text recognition* menggunakan dataset IAM. IAM database adalah database yang terdiri dari kalimat bahasa Inggris tulisan tangan, yang mencakup 1.066 formulir yang diproduksi oleh sekitar 400 penulis berbeda [1].

Karya ilmiah ini bertujuan untuk memberi kesimpulan dalam pengenalan tulisan tangan, dan juga gambaran dalam konteks kemudahan penerapan dan performa akurasi yang dapat dihasilkan.

2. LANDASAN TEORI

2.1 Ijazah

Ijazah adalah sebuah sertifikat dokumen yang diberikan kepada pelajar dan diberikan setelah pelajar tersebut selesai atau tamat belajar. Ijazah diberikan oleh instansi pendidikan baik dalam negeri maupun luar negeri. Ijazah adalah suatu dokumen pengakuan prestasi belajar dan/atau penyelesaian suatu jenjang pendidikan tinggi sesudah lulus ujian yang diselenggarakan oleh perguruan tinggi tercantum dalam Peraturan Mendikbud RI No 81 Tahun 2014. Informasi pribadi utama yang terdapat pada ijazah adalah nama, tempat dan tanggal lahir, nomor induk, nama orang tua, sekolah asal.

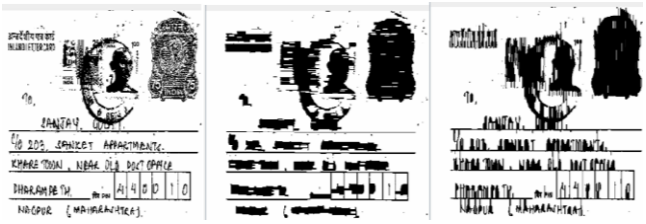
Ijazah memiliki format yang berbeda-beda. Pemerintah Indonesia biasa mengeluarkan format template ijazah yang baru setiap memasuki kurikulum ajaran baru. Contoh lokasi data ijazah tahun 2016/2017 pada Gambar 1.

Gambar 1. Lokasi Data Pribadi pada Ijazah tahun 2016/2017

2.2 *Run Length Smoothing Algorithm*

*Run Length Smoothing Algorithm (*RLSA*)* adalah metode yang digunakan untuk *block segmentation* dan *text discrimination*. RLSA digunakan untuk proses blob pada gambar binary agar dapat dilakukan segmentasi kata atau block pada gambar. RLSA mgnubah pixel hitam (0) dan putih (1) dengan aturan semua pixel pada gambar asli diubah dengan 0 jika pixel 1 setelahnya kurang dari atau sama dengan sebuah value.

Perhitungan algoritma ini dilakukan 2 kali pada umumnya yaitu vertical setelah itu horizontal. Hal ini bisa diubah pada konfigurasi parameter jika hanya ingin melakukan salah dari proses tersebut. Fungsi iterasi aturan yang telah dijelaskan sebelumnya merupakan perhitungan untuk horizontal, dan dapat digunakan juga pada kalkulasi vertical. Transpos dari gambar dan menggunakan iterasi horizontal akan mendapatkan hasil kalkulasi vertical. Contoh input dan output RLSA pada Gambar 2.

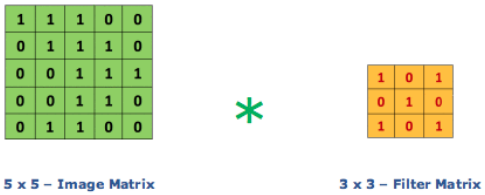


Gambar 2. Input Output RLSA horizontal & vertical [2]

2.3 *Convolutional Neural Network*

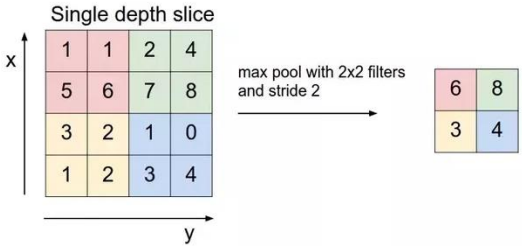
Dalam *neural network*, *Convolutional Neural Network* (CNN) adalah satu model yang dibuat khusus untuk kategori *image recognition*, *image classification*, *object detection*, *face recognition*, dan lainnya. CNN adalah tipe *artificial neural network* yang didesain khusus untuk mengolah pixel data. CNN menerima input berupa gambar, di proses, dan diklasifikasi dalam beberapa kategori (contoh: kucing, macan, singa). CNN memiliki performa yang baik terutama dalam mencari sebuah pola.

CNN memiliki 3 layer utama yaitu *convolutional layer*, *pooling layer*, dan *full connected layer*. Proses konvolusi pada *convolution layer* bertujuan untuk mengektrak fitur dari gambar input [2]. Proses ini mengkalikan matriks gambar dengan filter matriks ukuran tertentu dan hasil kali akan disum untuk mendapatkan output feature. Proses konvolusi divisualisasikan seperti pada Gambar 3.



Gambar 3. Proses konvolusi [7]

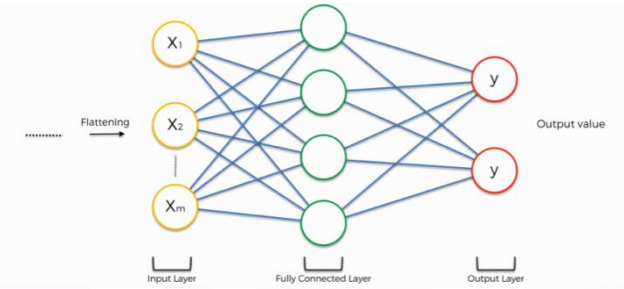
Pooling layer adalah layer yang mengurangi dimensi dari feature map. Proses pada layer ini lebih dikenal dengan langkah untuk downsampling, Hal ini berguna untuk mempercepat komputasi karena parameter yang harus diupdate semakin sedikit dan mengatasi overfitting [7]. Pooling layer yang biasa digunakan terdapat 2 jenis yaitu *max pooling* dan *average pooling. Max pooling* menggunakan nilai maksimum tiap pergeseran filter, sementara average pooling menggunakan nilai rata-rata. Contoh hasil output pooling layer dapat dilihat pada Gambar 4.



Gambar 4. Max Pooling [7]

Fully-connected layer adalah layer yang terhubung penuh layaknya neural network biasa. Layer ini akan menghitung skor kelas. Seperti neural network biasa setiap neuron dalam layer ini akan terhubung ke semua neuron berikutnya dalam volume.

Layer flatten mengubah matrix menjadi vector untuk dapat difeed ke fully connected layer. Contoh hasil flatten dan fully connected layer pada Gambar 5.



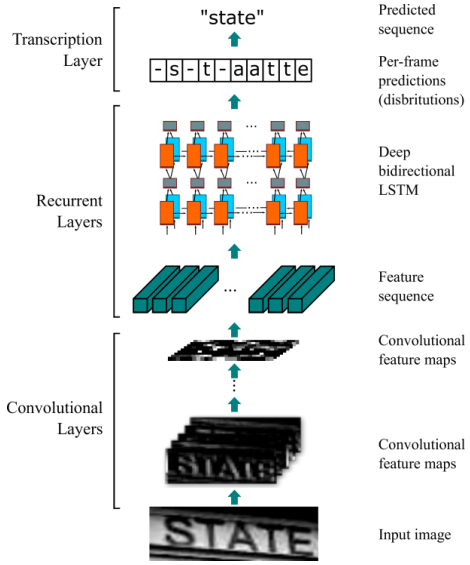
Gambar 5. Flattening & Fully-Connected Layer [4]

2.4 Convolutional Recurrent Neural Network

Convolutional recurrent neural network (CRNN) adalah model neural network yang memiliki layer *convolution*, layer *recurrent*, dan terakhir layer *connectionist temporal classification* (CTC). Berbeda dengan model CNN biasanya, CRNN hasil ekstraksi fitur dari *convolution layer* di *feed* ke *recurrent layer* (bukan *fully connected layer*) dan output layer menggunakan CTC.

CTC adalah sebuah layer fungsi yang digunakan untuk mengklasifikasi hasil akhir menjadi *string* dari sebuah matriks. CTC digunakan sebagai tahap terakhir dalam rangkaian *neural network* yang bertujuan untuk mengenali karakter. Hasil dari *neural network* sebelumnya pada umumnya berbentuk matriks, matriks ini diolah agar didapatkan teks *string* sebagai hasil akhir dari rangkaian proses yang ada.

CTC berguna sebagai *transcription layer* untuk menerjemahkan matriks *output* layer *recurrent* dan teks kebenaran dasar dan menghitung nilai loss. Dapat disimpulkan, CTC di *feed* matriks awal raw ouput dari RNN dan menerjemahkannya ke dalam teks akhir. Panjang teks kebenaran dasar dan teks yang dikenali paling panjang hingga n panjang karakter maksimal dari dataset *training*. Visualisasi CRNN dapat dilihat pada Gambar 6.



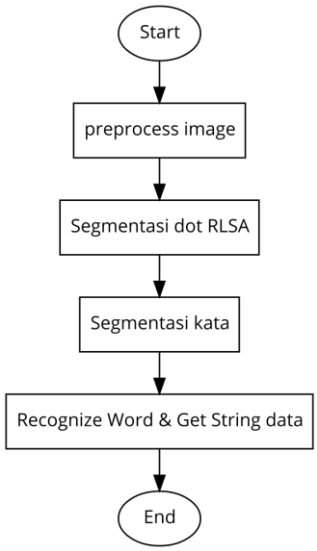
Gambar 6. *Convolutional Recurrent Neural Network* [5]

3. DESAIN SISTEM

Sistem menerima input berupa gambar hasil scan ijazah SD hingga SMA berwarna atau fotocopy (*grayscale*). Sistem tidak menerima gambar dari hasil foto kamera. Gambar dari kamera masih memerlukan sangat banyak praproses, untuk menghilangkan noise, perspective, hingga pengaturan terang gelap dari dokumen. Resolusi input gambar yang disarankan disekitar atau lebih dari atau sama dengan 850 lebar x 1100 tinggi, karena gambar akan diproses awal pada dimensi tersebut. Output dari sistem adalah hasil pembacaan data berupa *string*.

Pada ijazah, teknik segmentasi dengan fitur tulisan tidak dapat digunakan. Hal ini dikarenakan pada ijazah tidak memiliki konsistensi pada penulisan data. Dari posisi, ukuran, *weight* dan *style* dari tulisan sangat beragam bahkan penempatan stampel legalisir juga sangat mempengaruhi performa segmentasi berbasis tulisan ini. Hasil dari segmentasi masih banyak ditemukan noise (menggunakan metode RLSA). Oleh karena itu, metode segmentasi yang digunakan menggunakan fitur titik pada form.

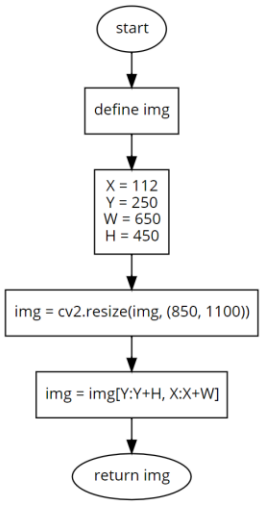
Penelitian ini akan menggunakan metode segmentasi dengan fitur titik untuk segmentasi ijazah. Metode ini digunakan untuk mengatasi permasalahan pada ijazah yang memiliki format berbeda-beda. Daerah pada ijazah yang ditandai dengan titik-titik, merupakan tempat untuk di isi data. Gambaran garis besar sistem pada Gambar 7.



Gambar 7. Gambaran Garis Besar Sistem.

3.1 Preprocess Gambar

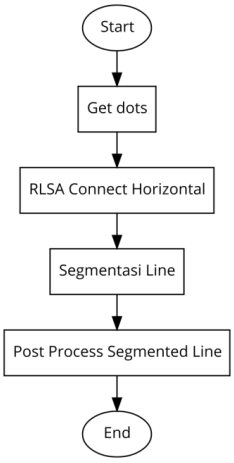
Gambar input akan dilakukan proses yang dinamakan *crop and reshape*. Proses ini berguna untuk menyeragamkan resolusi semua inputan gambar ijazah untuk proses segmentasi berikutnya. Proses ini mempengaruhi ukuran dot dan area fokus pada gambar inputan. Gambar ijazah akan dicrop pada bagian area fokus dan diresize dengan ukuran 650 *width* dan 850 *height*. Proses ini bersifat subjektif khusus hanya untuk gambar ijazah. Flowchart proses pada Gambar 8.



Gambar 8. Flowchart preprocess image.

3.2 Segmentasi Lokasi Data

Proses ini bertujuan untuk mendapatkan segmentasi dari lokasi data pada ijazah. Segmentasi dot atau segmentasi titik menggunakan metode *run length smoothing algorithm* dan digunakan untuk proses menghubungkan daerah titik-titik yang berdekatan secara horizontal dari gambar. Proses segmentasi dot sebenarnya hanya terdiri dari 3 tahapan. Memisahkan lokasi dot dengan bagian bagian lainnya dan menghubungkannya dengan metode.RLSA berikutnya dilakukan segmentasi pada garis. Flowchart proses pada Gambar 9.

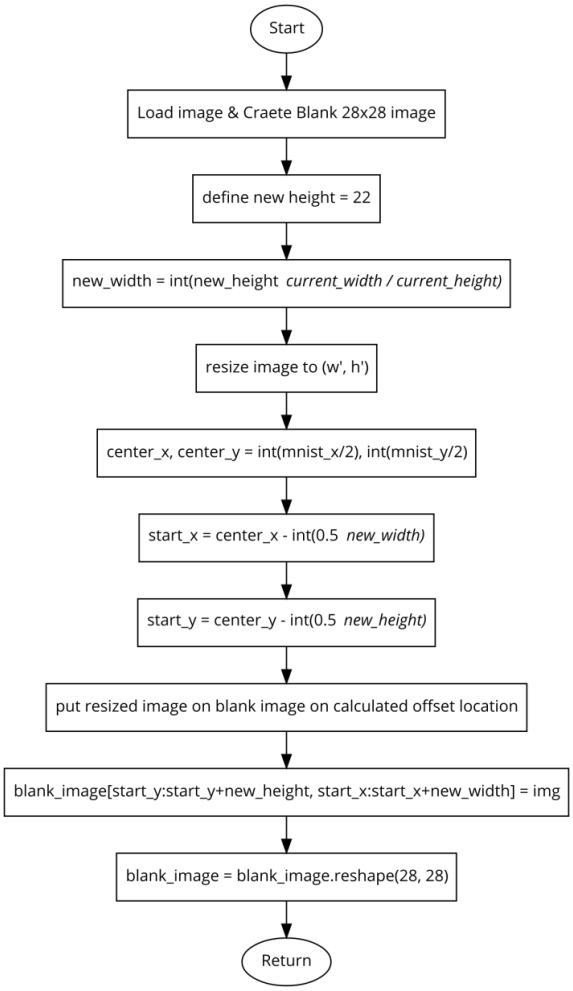


Gambar 9. Flowchart segmentasi lokasi data pada ijazah

3.3 Preprocess Data

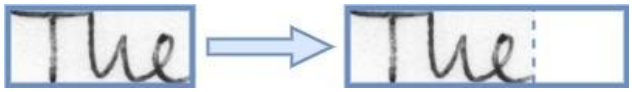
Sebelum dapat di feed ke model untuk proses rekognisi. Gambar data harus terlebih dahulu diproses. Setiap teknik yang digunakan membutuhkan tahapan proses yang berbeda. Teknik *text recognition* membutuhkan data untuk disegment menjadi gambar per kata. Sedangkan teknik *character recognition* membutuhkan data untuk disegment menjadi perhuruf/angka.

Pada segmentasi huruf memiliki kegagalan pada tulisan *cursive*, dimana pada penelitian ini dicoba untuk membuat algoritma penyesuaian. Proses segmentasi huruf hanya menggunakan pencarian contour biasa pada prosesnya. Gambar huruf diproses untuk menyerupai gambar pada dataset MNIST berukuran 28x28 dan *centerized*. Flowchart proses pada Gambar 10.



Gambar 10. Flowchart convert image to MNIST.

Hasil segmentasi kata akan diproses untuk dapat diprediksi pada proses berikutnya. Hasil segmentasi tersebut akan diolah lagi untuk dapat di *feed* ke model untuk di prediksi. Menggunakan fungsi yang sama dengan fungsi pengolah gambar untuk *training model text recognition* ini. Gambar akan diubah menjadi *grayscale* dan perubahan ukuran secara *aspect ratio* sesuai dengan input shape model. Input gambar harus *grayscale* (1 *channel*) dengan ukuran 128x32 pixels, berwarna putih pada tulisan dan berwarna hitam pada background (*invert*). Contoh pengubahan ukuran gambar menjadi 128x32 dapat dilihat pada gambar 11.



Gambar 11. Contoh desain gambar setelah di proses.

4. IMPLEMENTASI SISTEM

Imeplementasi system dilakukan pada computer dengan spesifikasi:

- RAM: 16GB, DDR4
- Memory: 1TB HDD
- CPU: Intel Core i7 8700K
- GPU: NVIDIA GeForce GTX 1060
- OS Windows 10 Pro

Implementasi kode system menggunakan bahasa pemrograman Python dengan versi 3.6. Adapun beberapa library yang mendukung system ini adalah:

- Numpy
- OpenCV (cv2)
- Tensorflow-gpu
- Matplotlib

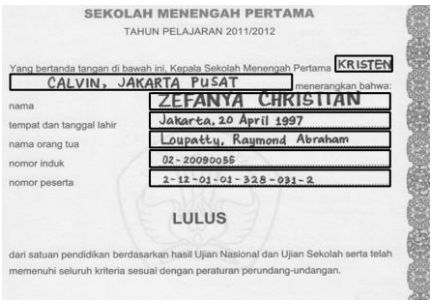
5. ANALISIS DAN PENGUJIAN

5.1 Segmentasi Lokasi Data

Pengujian akurasi hasil segmentasi RLSA dengan fitur titik dibandingkan dengan fitur teks. Segmentasi bertujuan memperoleh semua bagian data secara terpisah. Perhitungan akurasi menggunakan 10 sample ijazah yang telah dipilih dengan alasan memiliki standar kualitas gambar cukup baik dan juga ditemukannya kasus-kasus yang dapat menurunkan akurasi segmentasi. Dari setiap sample dilakukan segmentasi, dan dari setiap segment akan dihitung berapa kebenaran *character* yang tersegmentasi dibagi dengan jumlah slot pada form sample ijazah tersebut. Hasil akhir tiap sample dijumlah dan dibagi jumlah sample yang ada. Tabel pengujian dapat dilihat pada Tabel 1. Visualisasi hasil segmentasi akhir dapat dilihat pada Gambar 12.

Tabel 1. Pengujian akurasi segmentasi lokasi data ijazah.

Sample	RLSA-dot	RLSA-text-5
ijazah1.jpg	0.98	0.86
ijazah2.jpg	0.85	0.33
ijazah3.jpg	0.95	0.55
random1.jpg	0.97	0.50
random2.jpg	1.00	0.62
random3.jpg	1.00	0.90
random4.jpg	1.00	0.87
random5.jpg	0.77	0.22
random6.jpg	1.00	0.45
random7.jpg	0.96	0.57
Acc	0.951	0.587



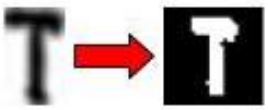
Gambar 12 Hasil Segmentasi lokasi data metode RLSA *dot feature*.

5.2 Preprocess Data

Pengujian terhadap metode untuk konversi gambar ke format untuk model text recognition dilakukan dengan visualisasi hasil implementasi. Pengujian dilakukan dengan sample ijazah setelah proses segmentasi kata pada pengujian sebelumnya. Hasil dari proses ini berupa gambar dengan ukuran lebar 128 dan tinggi 32 (Gambar 13).

Gambar 13. Visualisasi hasil preprocess

Pengujian terhadap metode untuk konversi gambar ke format MNIST dilakukan dengan visualisasi hasil implementasi. Proses ini digunakan untuk proses prediksi karakter menggunakan model MNIST atau Extended MNIST. Hasil pada pengujian sebelumnya dilakukan konversi agar gambar dapat di *feed* ke model neural network. Visualisasi pengujian implementasi pada Gambar 14. Dari gambar yang berukuran random, di ubah menjadi grayscale/binary dengan ukuran 28x28.



Gambar 14 Visualisasi proses konversi gambar ke MNIST

5.3 Pengujian Akurasi Pengenalan

Pengujian dilakukan pada 5 sample ijazah di 3 bagian data. ijazah telah dipilih kusus karena memiliki kualitas gambar yang cukup baik dan memiliki kasus-kasus yang mempengaruhi performa segmentasi maupun rekognisi. Pengujian dilakukan pada data ijazah bagian nama, nomor induk, dan juga nama orang tua/wali. Label tersebut terdapat pada setiap ijazah dari berbagai tahun. Nama pada ijazah sebagian besar menggunakan huruf capital, nomor induk untuk menguji pengenalan angka dan orang tua/wali untuk pengujian pengenalan *cursive handwriting.*

Pengujian dilakukan dengan model *character recognition* LeNet5 yang ditraining dengan data *Extended MNIST.* Model *text recognition* menggunakan 5 layer convolution dan 2 layer *bidirectional LSTM* 256 unit, yang ditraining dengan IAM database. *Tesseract tool* juga digunakan sebagai model untuk *text recognition. Preprocessing* data dilakukan seperti pada Bab 3.

Tabel 2. Pengujian Akurasi (*Jaro Distance*) bagian nama

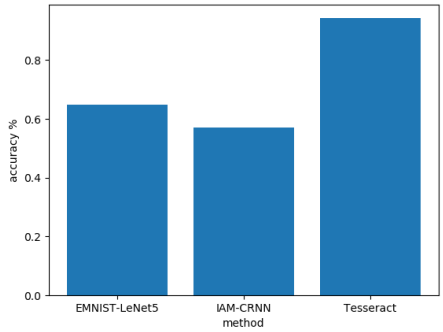
	Model Name		
Sample	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	0.6714200831847891	0.658312447786132	0.9004127966976264
ijazah3	0.7495122630992196	0.41629629629629633	1.0
random3	0.5166666666666667	0.5526960784313726	0.9226579520697169
random11	0.5943627450980392	0.5468409586056645	0.9215686274509803
random12	0.7037037037037037	0.6851851851851851	0.9666666666666667
Average	0.6471330923504837	0.5718661932609301	0.942261208576998

Tabel 3. Pengujian Akurasi (*Ratio*) bagian nama.

	Model Name		
Sample	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	0.5	0.65	0.8947368421052632
ijazah3	0.6122448979591837	0.34615384615384615	1.0
random3	0.4	0.48484848484848486	0.9714285714285714
random11	0.48484848484848486	0.5142857142857142	0.8823529411764706
random12	0.5555555555555556	0.5263157894736842	0.9473684210526315
Average	0.5105297876726448	0.5043207669523458	0.9391773551525875

Tabel 2 dan Tabel 3 merupakan pengujian akurasi model membaca bagian nama pada sample. Nama pada ijazah sebagian besar menggunakan huruf capital. Pengujian ini bertujuan untuk mengetahui seberapa baik model membaca tulisan tangan dengan huruf capital. Hal ini membuat model rekognisi karakter (Extended MNIST) memiliki akurasi cukup tinggi karena huruf dapat tersegment dengan cukup baik. Model rekognisi karakter juga memiliki akurasi yang lebih tinggi dibandingkan dengan model rekognisi teks (IAM). Berdasarkan pengamatan, model IAM di *training* dengan tidak banyak data yang menggunakan huruf capital (sebagian besar *cursive handwriting*).

Berdasarkan pengamatan akurasi, tool tesseract memperoleh akurasi yang sangat baik dalam membaca berbagai jenis tulisan tangan bagian nama pada ijazah. Akurasi yang diperoleh tool tesseract mencapai rata-rata hingga 94%. Tesseract terbukti memiliki akurasi jauh lebih unggul dalam membaca bagian nama pada ijazah dibandingkan dengan opsi model lainnya. Hal ini juga mengatasi permasalahan rekognisi huruf sekaligus menjadi solusi rekognisi teks. Grafik akurasi dapat dilihat pada Gambar 15.



Gambar 15. Grafik *Jaro Distance* Pengenalan Bagian Nama

Tabel 4. Pengujian akurasi (*Jaro Distance*) model pada Dataset Sample bagian nomor induk.

	Model Name			
Sample	MNIST-LeNet5	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	1.0	0.8666666666666667	0.0	0.8666666666666667
ijazah3	0.9259259259259259	0.8518518518518517	0.0	0.8518518518518517
random3	1.0	0.6666666666666666	0.0	1.0
random11	0.4952380952380952	0.4142857142857143	0.0	0.9333333333333332
random12	0.5277777777777778	0.0	0.5555555555555555	0.0
Average	0.7897883597883598	0.5598941798941798	0.1111111111111109	0.7303703703703703

Tabel 5. Pengujian akurasi (*Ratio*) model pada Dataset Sample bagian nomor induk.

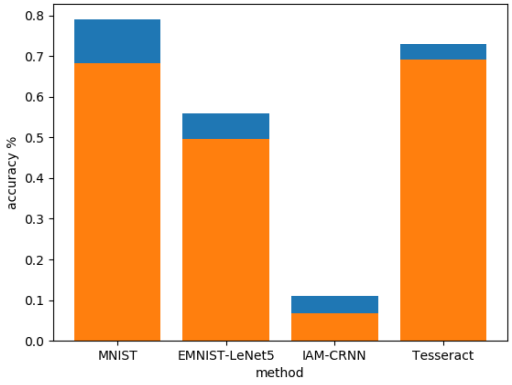
	Model Name			
Sample	MNIST-LeNet5	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	1.0	0.8	0.0	0.8
ijazah3	0.8888888888888888	0.7777777777777778	0.0	0.7619047619047619
random3	1.0	0.5	0.0	1.0
random11	0.23529411764705882	0.11764705882352941	0.0	0.9
random12	0.2857142857142857	0.2857142857142857	0.3333333333333333	0.0
Average	0.6819794584500467	0.4962278244631186	0.06666666666666667	0.6923809523809523

Tabel 4 dan Tabel 5 merupakan pengujian akurasi model membaca bagian nomor induk pada ijazah. Pengujian bertujuan untuk mengetahui seberapa baik model dalam membaca data tulisan tangan angka. Angka tidak memiliki kasus *cursive*, hanya ditemukan angka yang berhimpit. Hal ini membuat akurasi rekognisi karakter rendah tergantung dengan hasil segmentasi.

Model IAM mendapatkan akurasi yang sangat rendah. Akurasi yang diperoleh hanya mencapai 11% jaro dan 6% ratio kemiripan kata. Sample dataset IAM yang digunakan untuk training tidak banyak mengandung huruf kapital maupun angka didalamnya. Hal ini membuat model IAM yang digunakan hanya dapat membaca tulisan tangan *cursive.*

Model Extended MNIST mendapatkan akurasi yang lebih rendah dibandingkan MNIST. Model EMNIST memiliki performa yang kurang baik saat membaca karakter seperti angka ‘6’ dan huruf ‘b’, angka ‘1’ dan huruf ‘i’, angka ‘0’ dan huruf ‘O’. Oleh karena itu, model MNIST yang dibuat khusus untuk digit recognition memiliki akurasi yang lebih tinggi. Meskipun begitu, akurasi yang diperoleh model EMNIST masih lebih baik dibandingkan model IAM dalam membaca data angka.

Model MNIST mendapatkan akurasi lebih baik dibandingkan model tesseract pada metrics jaro, sebaliknya pada metrics ratio. Pada 2 sample terakhir model MNIST mendapatkan akurasi kemiripan yang rendah, hal ini dikarenakan hasil segmentasi karakter yang tidak baik. Grafik akurasi dapat dilihat pada Gambar 16.



Gambar 16. Grafik *Jaro Distance* (biru) dan *Levenshtein Ratio* (oren) pengenalan bagian ID

Tabel 6. Pengujian akurasi (jaro) model pada Sample ijazah bagian nama orang tua/wali.

	Model Name		
Sample	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	0.5222222222222223	0.8141025641025642	0.8675213675213675
ijazah3	0.3923076923076923	0.7073593073593073	1.0
random3	0.5416666666666666	0.7235449735449735	1.0
random11	0.234006734006734	0.6631313131313131	1.0
random12	0.4256410256410256	0.5904040404040404	0.6972222222222221
Average	0.4231688681688682	0.6997084397084398	0.912948717948718

Tabel 7. Pengujian akurasi (ratio) model pada Sample ijazah bagian nama orang tua/wali.

	Model Name		
Sample	EMNIST-LeNet5	IAM-CRNN	Tesseract
ijazah1	0.23529411764705882	0.72	0.8
ijazah3	0.08695652173913043	0.6666666666666666	1.0
random3	0.3	0.6153846153846154	1.0
random11	0.1	0.5964912280701754	1.0
random12	0.2222222222222222	0.3783783783783784	0.5161290322580645
Average	0.1888945723216823	0.5953841776999672	0.863225806451613

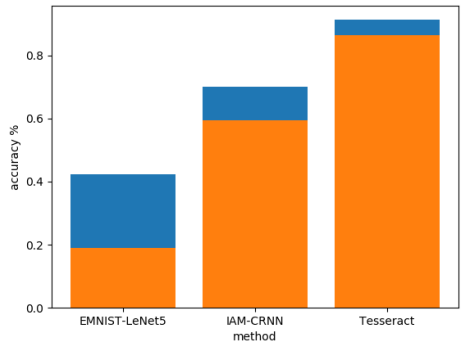
Tabel 6 dan Tabel 7 merupakan pengujian akurasi model pada sample ijazah pada bagian nama orang tua/wali. Pengujian bertujuan untuk mengukur akurasi model dalam membaca tulisan tangan dengan huruf yang terhubung atau *cursive*. Berbeda dengan tulisan tangan pada nama yang sebagian besar huruf kapital, pada bagian ini penulisan dilakukan sesuai standard, huruf awal kapital dan bahkan ada yang *cursive*. Keunikan tulisan tangan juga terlihat pada bagian ini.

Model Extended MNIST memiliki akurasi yang sangat rendah. Akurasi yang diperoleh hanya 18% pada metrics ratio. Hal ini

dikarenakan kegagalan segmentasi karakter. Model EMNIST tidak dapat digunakan untuk rekognisi tulisan dengan *cursive* atau tulisan yang ditemukan karakter terhubung.

Model IAM mendapatkan akurasi yang cukup baik dibandingkan dengan 2 percobaan sebelumnya. Akurasi diperoleh hingga 70% dengan metrics jaro dan 60% pada metrics ratio. Model IAM memiliki performa yang lebih baik dalam pembacaan data tulisan *cursive*. Bahkan pada sample terakhir yang memiliki jenis tulisan tangan sangat unik, masih bisa mendapatkan akurasi 59% jaro dan 37% ratio.

Model tesseract memiliki performa terbaik dibagian *cursive*. Tesseract dapat memperoleh akurasi hingga 100% dengan rata-rata keseluruhan 91% dengan metrics jaro dan 86% dengan metrics ratio. Tesseract terbukti dapat membaca tulisan tangan *cursive* maupun penulisan standard. Bahkan pada sample 'random12' yang menggunakan jenis tulisan cursive rait Indonesia, tesseract dapat memperoleh akurasi hingga 70% dengan jaro dan 51% dengan ratio Grafik akurasi dapat dilihat pada Gambar 17.



Gambar 17. Grafik *Jaro Distance (biru)* dan *Levenshtein Ratio* (oren) pengenalan bagian nama orang tua/wali

6. KESIMPULAN

Berdasarkan hasil pengujian dapat disimpulkan beberapa hal sebagai berikut:

- *Run length smoothing algorithm (text)* dan *EAST Text Detection* dapat digunakan untuk mencari kandidat data pada ijazah. Tetapi memiliki kelemahan akurasi saat proses *filter* kandidat tersebut.
- Akurasi metode segmentasi RLSA dengan fitur titik (konfigurasi default) memiliki akurasi yang jauh lebih tinggi dibandingkan *RLSA-TextFeature*. Akurasi yang diperoleh rata-rata 95.1% sedangkan *RLSA-TextFeature* hanya mendapatkan rata-rata 58.7%.
- Model dengan teknik rekognisi karakter memiliki akurasi yang sangat rendah dalam pembacaan data dengan tulisan tangan *cursive*. Akurasi yang didapat hanya 42% metrics jaro dan 18% metrics ratio kemiripan. Tetapi data ijazah bagian nama sebagian besar menggunakan huruf capital dimana permasalahan cursive tidak terjadi, hal ini membuat akurasi model meningkat hingga 64% jaro dan 51% ratio.
- Model IAM mendapat akurasi yang rendah pada saat pembacaan data angka dibawah 30% dan mendapatkan akurasi hingga 70% pada tulisan tangan *cursive*. Data angka dapat dibaca dengan baik oleh model MNIST dan tesseract.
- Tool tesseract memiliki performa yang sangat baik dalam merekognisi segala jenis tulisan tangan pada ijazah. Akurasi yang didapat sangat baik, rata-rata akurasi pembacaan 3 bagian data memperoleh akurasi diatas 69% hingga 100%.

7. DAFTAR PUSTAKA

[1] Bunke, H., Marti, U. 2002. The IAM-database: an English sentence database for offline handwriting recognition. *IJDAR* **5,** 39–46. DOI: https://doi.org/10.1007/s100320200071

[2] Borlepwar, A. P., Borakhade, S. R., & Pradhan, B. 2017. *Run Length Smoothing Algorithm for Segmentation*

[3] G. Cohen, S. Afshar, J. Tapson and A. van Schaik. 2017. EMNIST: Extending MNIST to handwritten letters. International Joint Conference on Neural Networks (IJCNN), Anchorage, AK, 2017, pp. 2921-2926, DOI: 10.1109/IJCNN.2017.7966217.

[4] SuperDataScienceTeam. 2018. Convolutional Neural Networks (CNN): Step 4 – Full Connection. (Aug 2018). Retrieved from superdatascience: https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection

[5] Shi, B., Bai, X., & Yao, C. 2016. *An End-to-End Trainable Neural Network for Image-based Sequence Recognation and Its Application to Scene Text Recognation*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2298-2304.

[6] Supriana, I., Ramadhan, E. 2015. *Pengenalan Tulisan Tangan untuk Angka tanpa Pembelajaran.* Bandung, Indonesia: Konferensi Nasional Informatika 2015.

[7] Ujjwalkarn. 2017, May 29. *An Intuitive Explanation of Convolutional Neural Networks*. Retrieved from ujjwalkarn: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

[8] Wirayuda, T.A.B., Syilvia, V. & Retno, N.D. (2009). *Pengenalan Huruf Komputer Menggunakan Algoritma Berbasis Chain Code dan Algoritma Sequence Alignment*, 19-24. Bali, Indonesia: Konferensi Nasional Sistem dan Informatika