Implementasi Metode *Backpropagation* dalam Pengolahan Citra Teks Tulisan Tangan Menjadi Teks Digital dan *Text-to-Speech* pada Sistem Operasi Android Sebagai Alat Bantu Komunikasi Tuna Wicara

Salsabila Maharani Alvananda Herlambang[1], Anita Qoiriah[2]
[1,2] Jurusan Teknik Informatika, Fakultas Teknik, Universitas Negeri Surabaya
[1]salsabila.17051204009@mhs.unesa.ac.id
[2]anitaqoiriah@unesa.ac.id

Abstrak— **Penyandang tuna wicara umumnya berkomunikasi menggunakan bahasa isyarat, namun juga ada beberapa yang menggunakan media alat tulis. Kasus ini berlaku untuk penyandang tuna wicara yang mengerti dan memahami bahasa tulisan atau abjad dengan baik. Aplikasi ini dibangun untuk membantu komunikasi penyandang tuna wicara dengan memanfaatkan metode *backpropagation* dalam mengubah tulisan tangan menjadi teks digital. Teks digital tersebut nantinya akan diubah menjadi ucapan (*text-to-speech*) agar langsung bisa dimengerti oleh lawan bicara. Pengembangan aplikasi ini membutuhkan dua bahasa pemrogaman dalam pengembangannya, yakni *python* dan Android Studio. Pemrosesan metode *backpropagation* dan pengembangan model dilakukan pada *python* dilanjutkan dengan mengekeskpor hasil *graph* model yang telah dibekukan (*freeze*) menggunakan bantuan library *TensorFlow* pada Android Studio untuk melanjutkan pembuatan *interface* aplikasi sehingga menjadi aplikasi berbasis Android yang dapat digunakan pada smartphone berbasis sistem operasi android. Hasil akurasi model yang didapat dalam proses pelatihan sebesar 83,22 %. Pengujian aplikasi dengan menggunakan pengujian blackbox testing mendapat hasil sempurna, sedangkan pengujian berdasarkan sampel karakter masukan dari pengguna sebanyak 67 karakter yang mewakili huruf abjad besar, huruf abjad kecil, dan angka mendapat hasil 57 karakter terdeksi dengan benar dan 10 karakter terdeteksi salah.**

Kata Kunci— *Backpropagation, text-to-speech, handwriting*, **android.**

I. PENDAHULUAN

Penyandang tuna wicara berkomunikasi dengan menggunakan bahasa khusus atau bahasa non-verbal yang biasa disebut bahasa isyarat. Selain itu, penyandang tuna wicara juga berkomunikasi dengan melihat gerak bibir, gesture, gerakan kepala maupun gerakan tubuh dari lawan bicara [1]. Banyak juga yang menggunakan alat tulis sebagai media komunikasi karena dirasa lebih efektif, dan juga bisa langsung dimengerti oleh penyandang disabilitas. Namun, hal ini bekerja pada tuna wicara yang mengerti bahasa tulisan atau abjad dengan baik, disisi lain masih banyak penyandang tuna wicara yang belum mengerti betul bagaimana bahasa tulisan tersebut. Ada beberapa penyandang tuna wicara yang alat indra pendengarannya masih berfungsi dengan baik dalam menangkap suatu bunyi sehingga mereka masih dapat mendengar dan mengerti apa yang dikatakan lawan bicara menggunakan bahasa sehari-hari [2].

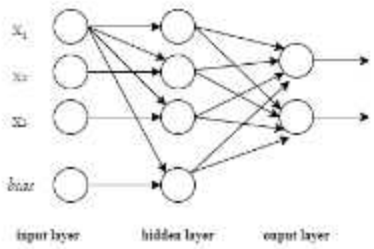
Melihat dari penelitian yang dilakukan oleh Intan Sari Areni, Sri Wahyuni, Indrabayu, dan Anugrahyani dengan judul "Solution to Abbreviated Words in Text Messaging for Personal Assistant Application" menghasilkan sistem Text-to-Speech untuk membaca teks pesan singkatan pada SMS berbasis Android [3], dan juga dari jurnal penelitian oleh Fifin Hietania, Wayan Santiyasa, dan Ida Bagus Gede Dwidasmara yang berjudul "Implementasi Backpropgation Dalam Pengolahan Citra Teks Tulisan Tangan Menjadi Teks Digital" menghasilkan sistem yang dapat mengenali pola tulisan tangan dan mengubahnya menjadi teks digital menggunakan metode *backpropagation* dalam proses pengenalannya [4]. Penulis kemudian menggabungkan topik dari kedua penelitian tersebut untuk membuat suatu sistem yang dapat mengenali tulisan tangan untuk dijadikan teks digital kemudian dibuat menjadi Text-to-Speech dalam sistem operasi Android. Selain itu, penulis juga mengambil referensi dari penelitian oleh Barry Ceasar Octariadi dan Yulrio Brianorman dengan judul "Pengenalan Pola Tanda Tangan Menggunakan Metode Jaringan Syaraf Tiruan *Backpropagation*" menghasilkan keakurasian sebesar 82% dalam mengidentifikasi sampel tanda tangan yang tidak diketahui atau tidak dilatihkan menggunakan metode jaringan saraf tiruan *backpropagation* [5]. Oleh karena itu, penulis menggunakan metode jaringan saraf tiruan *backpropagation* dalam mengolah masukan teks tulisan tangan dari pengguna menjadi teks digital.

Dari referensi - referensi dan masalah tersebut maka perlu dibuatkan suatu aplikasi sebagai alat bantu untuk berkomunikasi dengan penyandang tuna wicara yang sederhana dan bisa digunakan oleh banyak orang maupun penyandang tuna wicara itu sendiri. Pengguna hanya perlu menuliskan kalimat yang ingin diucapakan pada aplikasi, dan sistem akan memproses tulisan tangan tersebut menjadi teks digital yang kemudian akan diucapkan.

A. *Backpropagation*

Backpropagation adalah salah satu jenis metode Jaringan Syaraf Tiruan dengan supervised learning. Metode ini dapat mengenali pola masukan yang tidak lengkap dan memiliki kestabilan pada setiap kali *recall* untuk mengenali pola citra.

Jaringan *backpropagation* terdiri dari *input layer, hidden layer,* dan *output layer*. Pada prosesnya, jaringan tidak memiliki koneksi khusus untuk melakukan perhitungan mundur dari tiap *layer* tetapi, menggunakan error pada *output layer* yang dipropagasikan ke belakang menuju *input layer*. Berikut adalah gambaran dari jaringan *backpropagation* seperti yang ditunjukkan Gambar 1.



Gambar. 1 Arsitektur jaringan backpropagation

Proses yang terjadi dalam *backpropagation* merupakan turunan dari jaringan saraf tiruan, yaitu *learning, networks storing,* dan *recalling*. Proses *learning* yang terjadi adalah input layer menerima pola masukan dan memprosesnya berdasarkan bobot awal yang diperoleh secara acak. Jaringan akan terus menyesuaikan terhadap bobot yang ada sampai menghasilkan keluaran dari jaringan target yang diharapkan menjadi sama. Proses *learning* menggunakan parameter *learning rate* pada setiap siklusnya. Karena membutuhkan waktu yang lama, proses ini dibatasi dan akan berhenti saat selisih antara keluaran dan target mencapai nilai terkecil dari *error rate*.

Proses *learning* selesai, dilanjutkan proses *recalling*. Merupakan proses komputasi hasil proses *learning* dari *input layer* dan bias pada masing-masing neuron layer. *Recalling* menghasilkan keluaran pada output layer.

Backpropagation mempunyai dua fase dalam pelatihannya, yakni proses propagasi nilai aktivasi atau masukan dan proses penyesuaian dengan keluaran yang diharapkan. Proses propagasi nilai aktivasi merupakan proses perubahan nilai *weight* yang menghubungkan tiap *layer*.

Hasil dari propagasi berupa fungsi aktivasi yang kemudian diteruskan dalam proses penyesuaian. Fungsi aktivasi digunakan untuk menurunkan nilai aktivasi dan mengubahnya menjadi nilai keluaran, atau untuk menambahkan nilai bias. Fungsi aktivasi yang digunakan dalam penelitian ini adalah fungsi sigmoid.

B. *Citra Digital*

Citra digital merupakan gambar dua dimensi dan terdiri dari himpunan nilai digital atau pixel yang dapat ditampilkan pada layar. Pengolahan citra digital (*image processing*) adalah proses yang berhubungan dengan peningkatan kualitas suatu citra agar sesuai dengan tujuan yang diinginkan.

Pada penelitian ini proses pengolahan citra dilakukan dengan machine vision, citra digital direpresentasikan kedalam bentuk array dua dimensi atau matrik. Besar ordo dari matrik menjadi parameter tingkat kedetailan suatu citra. Semakin banyak ordo matrik, maka semakin detail representasi citra yang dihasilkan dan sebaliknya. Setiap pixel dalam matrik mempresentasikan tiap pixel goresan tulisan tangan pada kanvas. Hasil inilah yang akan di masukkan pada proses selanjutnya.

C. *TensorFlow*

TensorFlow merupakan salah satu *library* dalam *python* yang mendukung pemrogaman *deep neural network* dan *machine learning* yang dikembangkan oleh Tim Google Brain. Salah satu fungsi utamanya yaitu, mendefinisikan, mengoptimalkan, dan menghitung secara efisien perhitungan yang melibatkan *array multidimension*. *TensorFlow* bisa menulis kode yang sama dan menjalankan di CPU maupun GPU, bahkan *TensorFlow* akan mengetahui bagian perhitungan yang harus dipindah ke GPU.

D. *Keras*

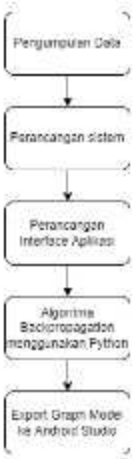
Keras merupakan salah satu library jaringan saraf tiruan yang bersifat open source dalam *python*. *Keras* berisi banyak fungsi implementasi pengembangan jaringan saraf yang umum digunakan seperti *layers, objectives, activation functions, optimizers,* dan sejumlah alat untuk mengolah data gambar. *Library keras* dapat beroperasi secara terpisah dan mandiri, namun juga dapat berintegrasi dengan *library* lain yang disebut *Backend* termasuk *TensorFlow* dengan mengaksesnya melalui modul tf.keras.

E. *Android TextToSpeech*

Android TextToSpeech adalah salah satu *library* dalam Android Studio yang berfungsi untuk mensitesis ucapan dari teks untuk pemutaran langsung atau disimpan menjadi file suara. *Library* Android TTS dilengkapi dengan pengaturan bahasa pilihan dan pengaturan ucapan, seperti kontrol pitch dan kecepatan. Inisialisasi awal Android TTS degan cara mengimplementasikan TextToSpeech OnInitListener agar dapat digunakan.

II. METODE PENELITIAN

Kerangka penelitian yang digunakan sebagai patokan dalam penelitian ini dapat dilihat pada Gambar 2.



Gambar. 2 Tahapan penelitian

Pada Gambar 2 dapat dilihat bahwa terdapat tahapan-tahapan yang akan dilalui selama penelitian ini, diantaranya adalah :

A. *Pengumpulan data*

Tahap pengumpulan data adalah tahap penulis mengumpulkan sampel untuk dataset. Pengumpulan data sampel pada penelitian ini digunakan untuk proses pelatihan dan pengujian. Data sampel dari penulis berisi 15 varian untuk setiap karakter. Sedangkan, pada tahap pengujian menggunakan sampel dari pengguna sebanyak 67 sampel. Selain itu, untuk lebih mengoptimalkan program aplikasi, ditambahkan data sampel luar yang diambil dari *MNIST Handwritten Digit Classification Dataset*.

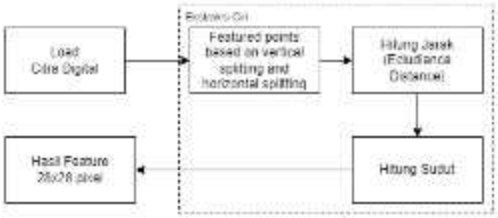
B. *Perancangan sistem.*

Tahap perancangan sistem adalah tahap penulis membuat rancangan sistem dari aplikasi agar dapat menjadi acuan dalam proses pembuatan atau implementasi aplikasi sesungguhnya.

Sistem aplikasi *handwriting recognition* adalah sistem yang dalam penggunaannya mendeteksi karakter tulisan tangan dan mengubahnya menjadi teks digital dan ucapan. Bahasa pemrograman yang digunakan adalah *python* digunakan dalam pemrosesan pendeteksian karakter hingga penyimpanan model bentuk graph dan *Java* pada Android Studio digunakan untuk membangun antarmuka aplikasi. Berikut tahapan dalam pendeteksian karakter.

1. *Pre-processing*

Proses *pre-processing* adalah pemrosesan awal yang bertujuan untuk mengolah citra agar dapat diambil karakteristiknya disebut juga dengan menyederhanakan gambar. Tahapan pre-processing dapat dilihat dalam Gambar 3.



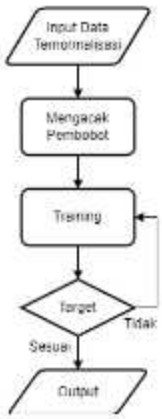
Gambar. 3 Tahapan proses pre-processing

Teknik pengolahan citra biasanya diawali dengan melakukan pendeteksian tepian obyek yang berada dalam gambar (*edge detection*). Karakter dianalisa satu per satu. Pendeteksian dilakukan dengan menggunakan teknik vertical run and horizontal run encoding, yakni mendeteksi batas paling kiri, paling kanan, paling bawah, dan paling atas dari karakter sampai diperoleh batas pinggir dari objek.

Setiap karakter yang sudah dideteksi tepi, besar dan ukurannya dikonversi kedalam bentuk matriks agar mudah disederhanakan atau disesuaikan ukurannya atau disebut juga tahap reshape. Karakter direshape menjadi 28 x 28 pixel. Tiap karakter yang sudah dimatrikkan mempunyai nilai pixel. Modifikasi nilai setiap pixel pada karakter menjadi kisaran dari 0 sampai 1 akan meningkatkan rate pada proses pembelajaran.

2. *Proses Training*

Struktur *backpropagation* yang digunakan dalam proses pembelajaran ini menggunakan 3 lapisan, yaitu lapisan input sebanyak 784 neuron, lapisan hidden sebanyak 128 neuron, dan lapisan output sebanyak 10 neuron. Nilai bobot (*weight*) ditentukan secara acak. Nilai bobot ditentukan sebelum proses pembelajaran dilakukan. Pada saat proses pembelajaran, akan ada error, yakni perbedaan nilai antara target dengan output. Error tersebut dapat diminimalisir dengan *sum square error*. Alur proses training digambarkan seperti pada Gambar 4.



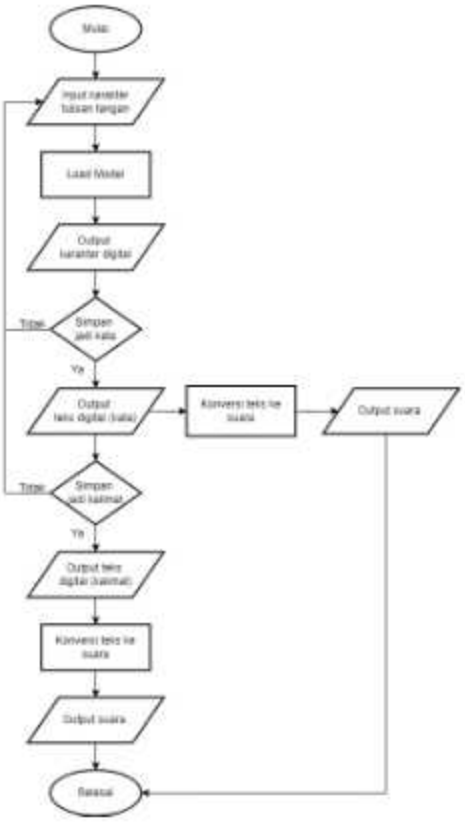
Gambar. 4 Diagram alur proes training pendeteksian karakter

3. *Graph Modeling*

Proses *graph modeling* adalah pemrosesan data dari hasil training menjadi graph. Pada graph menunjukkan hasil dan model accuracy terhadap epoch. Juga hasil dan model loss terhadap epoch. Graph selanjutnya akan diekspor dan diproses pada Android Studio.

Selanjutnya, pembuatan rancangan sistem aplikasi pada Android Studio. Rancangan sistem berupa diagram alur (flowchart) seperti pada Gambar 5.

ISSN : 2686-2220



Gambar. 5 Diagram alur aplikasi

Seperti yang ditunjukkan pada Gambar 5 sistem diawali dengan pengguna menulis di kanvas berupa satu karakter tulisan tangan dan menjadi masukan untuk sistem. Sistem selanjutnya akan memproses karakter tulisan tangan tersebut dalam model yang sudah tersimpan. Model ini adalah graph model hasil proses pembelajaran metode backpropagation yang sebelumnya telah dibuat dan dibekukan yang kemudian diekspor ke dalam Android Studio.

Hasil dari pemrosesan tersebut adalah deteksi karakter berupa teks digital. Pengguna harus menuliskan dan menyimpan hasil deteksi tiap-tiap karakter untuk dijadikan sebuah kata, berikut juga sama dengan membuatnya menjadi sebuah kalimat. Setelah itu, yang terakhir sistem akan melakukan proses konversi teks ke suara dan mengeluarkan *output* berupa suara.

C. *Perancangan Interface Aplikasi*

Tahap perancangan *interface* aplikasi dilakukan sebelum membuat aplikasi dengan tujuan untuk mempermudah pembuatan *interface* aplikasi. Berikut adalah prototype desain awal dari aplikasi yang akan dibuat.

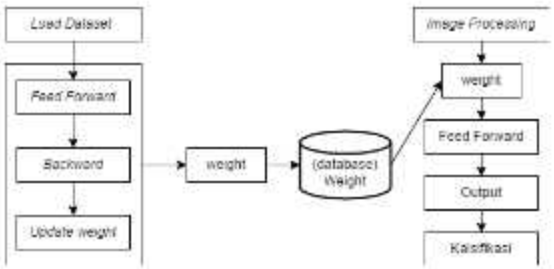
Gambar. 6 Tampilan desain awal aplikasi

Seperti pada Gambar 6 pada sisi atas terdapat tampilan bar yang berisi nama aplikasi. Dibawahnya adalah kanvas untuk tempat pengguna menulis karakter tulisan tangan. Kemudian, dibawah kanvas terdapat *view* untuk menampilkan hasil dari deteksi karakter yang dimasukkan, diikuti dengan *view* hasil deteksi kata dan *view* hasil deteksi kalimat. Setiap *view* ditambahkan dua tombol aksi. Tombol aksi pertama, yaitu tombol untuk mendeteksi dan menyimpan karakter, kata, maupun kalimat. Tombol aksi yang kedua adalah tombol untuk menghapus karakter, kata, ataupun kalimat yang tersimpan. Selain itu ditambahkan juga icon suara yang akan ditampilkan saat sistem mengeluarkan suara dari teks yang dibaca dan akan hilang saat pengguna menekan tombol hapus.

D. *Metode Backpropagation Menggunakan Python*

Tahap selanjutnya adalah mengimplementasikan metode *backpropagation* dalam sistem. Metode *backpropagation* yang digunakan penulis disini dibuat menggunakan bahasa pemrogaman *python*.

Berikut adalah gambar diagram alur metode *backpropagation* yang digunakan penulis.



Gambar. 7 Diagram alur metode backpropagation

Seperti pada Gambar 7 *backpropagation* memiliki 3 tahapan, yakni tahap propagasi maju, propagasi mundur, dan penyesuaian nilai bobot [6].

1. Tahap Propagasi Maju (*feed forward*)

   Adalah tahap pelatihan masukan. Setiap sinyal masukan perhitungan maju dihitung maju menggunakan fungsi aktivasi yang ditentukan

dari layar *input* ke layar tersembunyi hingga layar keluaran.

2. Tahap Propagasi Mundur (*backward*)

   Adalah tahap perhitungan terhadap error. Selama proses pelatihan terjadi error atau selisih antara keluaran jaringan dengan target yang diinginkan. Setiap unit keluaran membandingkan aktivasi dengan target keluaran yang telah ditetapkan sebelumnya untuk menentukan galat antara pola masukan dengan unit keluaran selama pelatihan.

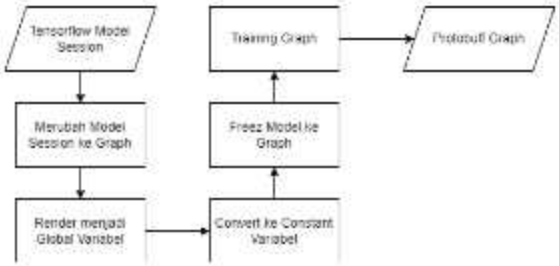
3. Tahap Penyesuaian Nilai Bobot (*update weight*)

   Guna menurunkan error yang terjadi dilakukan modifikasi bobot. Galat yang diperoleh pada langkah sebelumnya digunakan untuk mengubah bobot antara keluaran dengan lapisan tersembunyi.

Ketiga tahapan itu menghasilkan weight. Weight merupakan tujuan akhir dari metode *backpropagation*. Weight yang didapat dari proses sebelumnya akan disimpan dan dijadikan acuan dalam proses pengujian selanjutnya. Proses pengujian yang dimaksud adalah sistem memproses masukan citra dari pengguna yang nantinya akan menghasilkan *output* klasifikasi.

E. *Export Graph Model ke Android Studio*

Tahapan berikutnya adalah export *Graph* Model ke Android Studio atau juga disebut *Export Inference Graph*. Pada tahap ini, hasil dari model pelatihan *backpropagation* di bekukan (*freeze*) untuk diubah menjadi *graph* model, sebelum di export ke dalam Android Studio. Berikut diagram alur pembekuan model training terlihat seperti Gambar 8.



Gambar. 8 Diagram alur pembekuan model

Proses pembekuan atau yang lebih dikenal dengan nama *freezing* dilakukan menggunakan *TensorFlow* pada *python*. Seperti yang terlihat pada gambar diatas, hasil akhir dari proses pembekuan adalah Protobuf *Graph*. Protobuf (Protocol Buffer) adalah sebuah file yang dapat diexport ke Android Studio.

III. HASIL DAN PEMBAHASAN

Hasil dan pembahasan penelitian ini akan membahas mengenai implementasi *interface* aplikasi, implementasi metode *Backpropagation* dan protobuf ke Android Studio, serta pengujian aplikasi.

A. *Implemetasi Interface Aplikasi*

Pembuatan *interface* aplikasi menggunakan aplikasi Android Studio. Implementasi dilakukan menggunakan contoh prototype desain yang telah disiapkan.

Berikut tampilan implementasi aplikasi yang diambil dari tangkapan layar handphone penulis.

1. Tampilan Halaman Utama Aplikasi

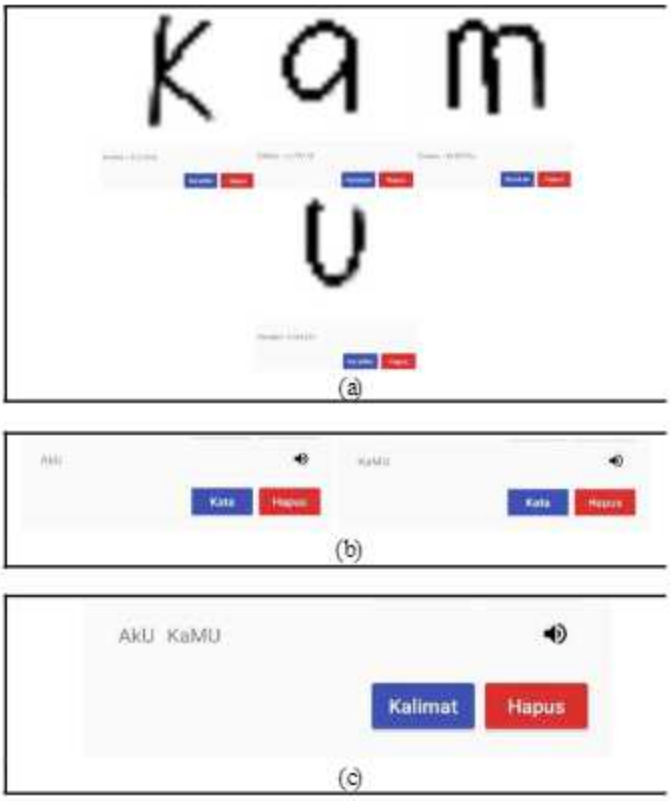
   Pada tahap implementasi *interface* dibuat berdasarkan rancangan yang telah ada sebelumnya. Tampilan halaman utama ini menggunakan layout constrain yang memungkinkan cocok untuk digunakan di banyak resolusi layar handphone. Tampilan halaman utama aplikasi ini ditunjuukan oleh Gambar 9.

Gambar. 9 Tampilan halaman utama

2. Tampilan Hasil Deteksi

   Aplikasi ini mendeteksi recognition tiap karakter, sehingga dalam mengubahnya menjadi sebuah kata dan kalimat membutuhkan penyesuaian tambahan, sehingga penulis memisahkan antara hasil deteksi tiap karakter, tiap kata, dan karakter. Berikut adalah tampilan dari hasil deteksi karakter, kata, dan kalimat dengan contoh kalimat "Aku kamu" yang ditunjukkan oleh Gambar 10.

ISSN : 2686-2220



Gambar. 10 (a) Tampilan hasil deteksi karakter, (b) Tampilan hasil deteksi kata, (c) Tampilan hasil deteksi kalimat

B. *Implementasi Metode Backpropagation*

Pada penelitian ini implementasi metode *backpropagation* menggunakan *python*. Tahap dalam pengimplementasiannya sebagai berikut :

1. *Load dataset*.
   Proses awal, yaitu memasukkan *dataset*. *Dataset* disini adalah *dataset* yang diambil dari *MNIST Handwritten Digit Classification Dataset*.
2. *Pre-processing*.
   Tahap ini merupakan tahapan persiapan mengolah *dataset*, dimana *dataset* yang di reshape menjadi 28 x 28 pixel.
3. *Modeling Data*.
   Pada tahap ini dilakukan backpro initial funion, yakni menginisialisasi fungsi *backpropagation* dan *modeling* data, yaitu pengelompokan data berdasarkan hasil reshape dan dibedakan untuk pelatihan dan pengujian.
4. *Train and Evaluation Data*.
   Disinilah tahap pengolahan data menggunakan *backpropagation*. Meliputi tahapan *feed forward*, *backward*, dan *update weight* untuk mengasilkan training layer dan training result. Setelah proses pengolahan kemudian dilanjutkan dengan proses evaluasi, dimana pada poses ini didapatkan hasil akurasi model sebesar 83,22%. Hasil dari tarining layer dan training result dapat dilihat dalam Tabel 1 dan Tabel 2.

TABEL I
TRAINING LAYER AND SHAPE

Model : sequential_3		
Layer (type)	***Output* Shape**	**Param #**
Reshape	(None, 28, 28, 1)	0
Conv2D	(None, 24, 24, 32)	832
MaxPooling2D	(None, 12, 12, 32)	0
Flatten	(None, 4608)	0
dense_2 (Dense)	(None, 512)	2359808
Dropout	(None, 512)	0
dense_3 (Dense)	(None, 62)	31806
Total params : 2.392.446		
Trainable params : 2.392.446		
Non-trainable params : 0		

Seperti pada Tabel 1, menunjukkan bahwa hasil reshape citra sebesar 28 x 28 pixel dapat menghasilkan pada layer dense_2 memiliki nilai paramaters sebanyak 2359808 berasal dari 4608 nilai input dikalikan 512 neuron layer awal ditambah 512 nilai bias. Juga pada layer dense_3 memiliki nilai parameters sebanyak 31806 berasal dari 512 nilai input dikali 62 neuron layer kedua ditambah 62 nilai bias layer kedua. Sehingga menghasilkan total parameters sebanyak 2359808 + 31806 = 2392446.

TABEL II
TRAINING RESULT

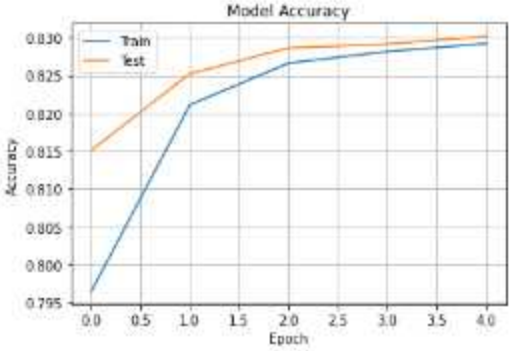
Learning rate = 0,1					
No.	Epoch	Loss	Accuracy	Val Loss	Val Accuracy
1.	Epoch 1/5	0.6214	0.7964	0.5465	0.8151
2.	Epoch 2/5	0.5250	0.8211	0.5127	0.8252
3.	Epoch 3/5	0.5059	0.8266	0.5014	0.8286
4.	Epoch 4/5	0.4994	0.8282	0.4952	0.8292
5.	Epoch 5/5	0.4958	0.8292	0.4926	0.8302

Pada Tabel 2 ditunjukkan hasil dari training result dengan *learning rate* 0,1 adalah nilai max diperoleh saat epoch ke 5 dari 5, yakni nilai min loss sebesar 0,4958, nilai min val_loss sebesar 0,4926, nilai max accuracy sebesar 0,8292 dan nilai max val_accuracy sebesar 0,8302.

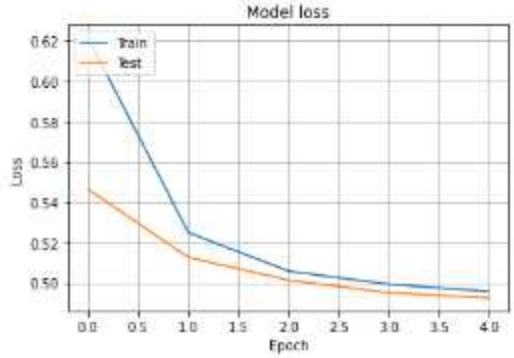
5. *Modeling Graph*.
   Setelah semua model selesai diolah, maka di tahap ini, hasil olahan sebelumnya dijadikan atau

 ISSN : 2686-2220

diekspor kedalam bentuk *Graph*. Merujuk dari hasil training data sebelumnya, maka didapat hasil *modeling graph* yang ditunjukkan oleh Gambar 11 dan Gambar 12.



Gambar. 11 Grafik model accuracy



Gambar. 12 Grafik model loss

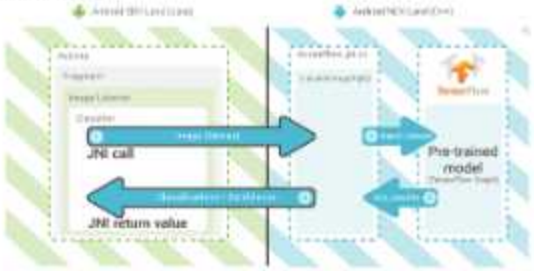
Hasil dari grafik graph seperti pada Gambar 11 dan Gambar 12, menunjukkan bahwa hasil model accuracy untuk train dan test terbaik didapat pada epoch ke 5 ditunjukkan dengan grafik yang meningkat dengan nilai accuracy sebesar 0,8292 dan nilai val_accuracy sebesar 0,8302. Hasil model loss untuk train dan test terbaik didapat pada epoch ke 5 ditunjukkan dengan grafik yang menurun dengan nilai loss sebesar 0,4958, nilai val_loss sebesar 0,4926.

6. *Freeze* The *Graph*.

Tahap ini merupakan tahapan persiapan *graph* sebelum diekspor ke Android. *Graph* yang diperoleh dari hasil sebelumnya dibebukan atau disebut dengan istilah *freeze* dengan mengganti variabel sebelumya menjadi variabel global. Bentuk file protobuf menjadi (.pb), dalam hal ini file disimpan dengan nama "HW-*Graph*.pb".

C. *Implementasi Protobuf ke Android Studio*

Implementasi protobuf ke Android Studio menggunakan bantuan library *TensorFlow*. Cara kerja menggunakan library *TensorFlow* dalam proses implementasi protobuf ke Android Studio seperti Gambar 13.



Gambar. 13 Cara kerja library tensorflow

Seperti Gambar 13, aplikasi ini menggunakan Java untuk pengembangan pada Android Sudio. *Java Native Interface* (JNI) digunakan untuk memanggil dari Java Android SDK ke *TensorFlow* dan menjembatani ke Android NDK. Diawali dengan pengambilan sampel data gambar bentuk bitmap dari Java Android SDK dikirimkan ke *TensorFlow* menjadi sebuah masukan. Kemudian, diproses oleh *TensorFlow Graph* untuk menghasilkan top result clasification yang menjadi hasil akhir dan dikirimkan kembali kepada Java Android SDK.

Grafik *TensorFlow* (.pb) dengan nama file "HW-*Graph*.pb" berada di "app\assets", dihubungkan dengan antarmuka "MainActivity". Pada MainActivity menginisialisasi, mendefinisikan model, dan menjalankan proses klasifikasi menggunakan *TensorFlow* Inference *Interface*.

Penambahan fitur text-to-speech pada aplikasi yang dibuat menggunakan bantuan library TextToSpeech dari android studio. Hal ini membuat hasil klasifikasi dapat didengar dalam bentuk suara.

D. *Pengujian Aplikasi*

Tahapan pengujian aplikasi dilakukan untuk menguji kinerja aplikasi agar dapat berfungsi sesuai dengan harapan penulis. Pada pengujian aplikasi ini, penulis menggunakan dua metode pengujian, yakni pengujian blackbox testing dan pengujian menggunakan karakter masukan dari pengguna.

1. Pengujian Blackbox

TABEL III
PENGUJIAN BLACKBOX TESTING

No	Skenario Pengujian	Hasil yang diharapkan	Kesimpulan
1	Membuka aplikasi	Menuju halaman utama aplikasi	Diterima
2	Menggambar pada kanvas	Muncul karakter pada kanvas sesuai goresan tangan pengguna	Diterima
3	Klik tombol karakter	Menampilkan pengklasifikasian terbaik hasil	Diterima

ISSN: 2686-2220

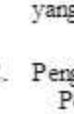
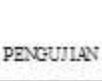
		deteksi karakter yang digambar	
4	Klik tombol kata	Menampilkan hasil deteksi kata yang diambil dari penggabungan karakter dan menghasilkan suara dari kata tersebut	Diterima
5	Klik tombol kalimat	Menampilkan hasil deteksi kalimat yang diambil dari penggabungan kata dan menghasilkan suara dari kalimat tersebut	Diterima
6	Klik tombol hapus karakter	Menghapus karakter	Diterima
7	Klik tombol hapus kata	Menghapus kata	Dterima
8	Klik tombol hapus kalimat	Menghapus kalimat	Diterima
9	Menutup aplikasi	Keluar dari aplikasi	Diterima

Berdasarkan hasil dari pengujian blackbox testing yang telah dilakukan seperti pada Tabel 3, dapat dilihat bahwa aplikasi yang dibuat dapat berfungsi dengan baik secara keseluruhan sesuasi yang diharapkan.

2. Pengujian Karakter

Pengujian menggunakan karakter masukan dari pengguna dilakukan untuk mengetahui akurasi penggunaan metode *backpropagation* dalam pengklasifikasian karakter. Pengujian dilakukan sebanyak 67 skenario.

TABEL IV
PENGUJIAN MENGGUNAKAN KARAKTER (ABJAD DAN ANGKA)

No	Karakter masukan	Hasil deteksi	Hasil yang diharapkan	Akurasi (%)	Kesimpulan
1		A	A	90,2	Benar
2		B	B	98,2	Benar
3		C	C	85,7	Benar
4		D	D	86,5	Benar
5		E	E	99,2	Benar
6		F	F	82,3	Benar
7		G	G	67,4	Benar
8		H	H	74,0	Benar
9		I	I	90,4	Benar
10		J	J	87,3	Benar
11		K	K	62,0	Benar
12		L	L	65,7	Benar
13		M	M	97,9	Benar
14		N	N	93,1	Benar
15		0	O	85,9	Salah
16		P	P	99,3	Benar
17		Q	Q	99,3	Benar
18		R	R	88,9	Benar
19		5	S	88,3	Salah
20		T	T	99,8	Benar
21		U	U	81,3	Benar
22		V	V	63,5	Benar
23		W	W	100,0	Benar
24		X	X	74,4	Benar
25		Y	Y	86,0	Benar
26		Z	Z	82,7	Benar
27		a	a	90,1	Benar

ISSN : 2686-2220

28		b	b	94,4	Benar
29		C	c	93,1	Salah
30		d	d	99,2	Benar
31		e	e	86,1	Benar
32		F	f	87,8	Salah
33		g	g	74,0	Benar
34		h	h	95,4	Benar
35		i	i	51,2	Benar
36		j	j	92,5	Benar
37		k	k	45,1	Benar
38		l	l	46,3	Benar
39		m	m	52,9	Benar
40		n	n	76,3	Benar
41		O	o	55,4	Salah
42		P	p	68,4	Salah
43		q	q	90,2	Benar
44		r	r	99,2	Benar
45		s	s	17,4	Benar
46		t	t	96,7	Benar
47		U	u	33,6	Salah
48		v	v	56,9	Benar
49		w	w	46,8	Benar
50		t	x	54,1	Salah
51		Y	y	68,6	Salah
52		t	z	75,1	Salah
53		0	0	42,7	Benar
54		1	1	61,7	Benar
55		2	2	77,9	Benar
56		3	3	99,8	Benar
57		4	4	92,6	Benar
58		5	5	92,6	Benar
59		6	6	65,0	Benar
60		7	7	99,2	Benar
61		8	8	79,1	Benar
62		9	9	98,8	Benar

TABEL V
PENGUJIAN MENGGUNAKAN KARAKTER (TULISAN TANGAN BERANTAKAN)

1		a	a	73,9	Benar
2		e	e	19,7	Benar
3		h	h	55,4	Benar
4		y	g	53,5	Salah
5		m	m	36,8	Benar

Berdasarkan uji coba dengan 67 karakter masukan pengguna yang mewakili dari huruf abjad besar, abjad kecil, dan angka pada Tabel 4 dan Tabel 5, hasil yang didapatkan yaitu ada 57 karakter yang berhasil di deteksi dengan benar dan 10 karakter yang dideteksi salah. Hasil akurasi model yang didapat pada proses pelatihan sebesar 83,22% dan hasil perhitungan akurasi tiap karakter seperti pada tabel diperoleh dari proses pelatihan model yang sebelumnya dibuat pada *python*.

IV. KESIMPULAN

Merangkum dari hasil penelitian yang telah dilakukan maka dapat disimpulkan sebagai berikut :

A. Metode *backpropagation* dapat digunakan dalam kasus klasifikasi karakter tulisan tangan bebrbasis android dengan bantuan dua bahasa pemrogaman dalam pengembangannya, yakni pengembangan model pada *python* dan pengembangan *interface* pada Android Studio. Library *TensorFlow* juga membantu menjembatani kedua bahasa pemrogaman tersebut agar aplikasi dapat berjalan dengan baik.

B. Aplikasi *handwriting recognition* ini dapat berjalan dengan baik terbukti dari hasil pengujian blackbox didapatkan hasil sempurna dengan semua fitur berfungsi sesuai harapan penulis.

C. Hasil akurasi model yang didapat pada proses pelatihan pada *python* sebesar 83,22%. Pengujian berdasarkan karakter masukan dari pengguna mendapatkan hasil sebanyak 57 karakter berhasil dideteksi dengan benar dan 10 karakter dideteksi salah dari 67 karakter yang mewakili dari huruf abjad besar, huruf abjad kecil, dan angka.

V. SARAN

Berdasarkan hasil penelitian dan implementasi program yang telah dibuat aplikasi *handwriting recognition* ini perlu pengembangan lebih lanjut, oleh sebab itu penulis menyarankan penelitian lanjutan dengan metode yang berbeda dan memperbanyak *dataset* dan epoch pada proses pelatihan sehingga nantinya aplikasi dapat berfungsi lebih optimal.

UCAPAN TERIMA KASIH

Puji syukur kepada Allah SWT yang telah memberikan rahmat dan ridha-Nya kepada penulis sehingga penulis dapat menyelesaikan penelitian ini. Terimakasih juga penulis ucapkan kepada orang tua dan orang-orang terdekat yang memberikan semangat, masukan, dan dukungan kepada penulis. Serta terimakasih kepada dosen pembimbing yang sudah membimbing penelitian ini hingga akhir.

REFERENSI

[1] Wiranda, N., & Putro, A. E. (2019). Model Identifikasi Kata Ucapan Tuna Wicara. *IJEIS (Indonesian Journal of Electronics and Instrumentation Systems)*, *9*(2), 131. https://doi.org/10.22146/ijeis.47609

[2] Hardien, V. S., Syafwan, & Trinanda, R. (2018). *PERANCANGAN BOOKLET MEDIA KOMUNIKASI TUNA RUNGU DI SEKOLAH LUAR BIASA (SLB) YPAC SUMBAR*. *151*(2), 10–17.

[3] Areni, I. S., Wahyini, S., Indrabayu, I., & Anugrahyani, A. (2017). Solution to abbreviated words in text messaging for personal assistant application. *Proceedings - 2017 International Seminar on Application for Technology of Information and Communication: Empowering Technology for a Better Human Life, ISemantic 2017, 2018-Janua*, 238–241. https://doi.org/10.1109/ISEMANTIC.2017.8251876

[4] Hietania, F., Santiyasa, W., & Dwidasmara, I. B. G. (2014). Implementasi Backpropgation Dalam Pengolahan Citra Teks Tulisan Tangan Menjadi Teks Digital. *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana)*, *1*(1), 1–10.

[5] Ocariadi, B. C., & Brianorman, Y. (2020). PENGENALAN POLA TANDA TANGAN MENGGUNAKAN METODE JARINGAN SYARAF TIRUAN *BACKPROPAGATION*. *Jurnal TEKNOINFO*, *14*(1), 15–21. https://doi.org/https://doi.org/10.33365/jti.v14i1.462

[6] Vemala, H. T., Puspitaningrum, D., & Setiawan, Y. (2013). *PENGENALAN POLA HURUF HIJAIYAH TULISAN TANGAN MENGGUNAKAN LOGIKA FUZZY DENGAN JARINGAN SYARAF TIRUAN BACKPROPAGATION* (Vol. 12).

[7] Andana, A., Widyati, R., & Irzal, M. (2018). Pengenalan Citra Tulisan Tangan Dengan Metode *Backpropagation*. *Jurnal Matematika Terapan*, *2*(1), 36–44. http://journal.unj.ac.id/unj/index.php/jmt/article/view/7166

[8] Anggraini, N., Kurniawan, A., Wardhani, L. K., & Hakiem, N. (2018). Speech Recognition Application for the Speech Impaired using the Android-based Google Cloud Speech API. *TELKOMNIKA*, *16*(6). https://doi.org/10.12928/TELKOMNIKA.v16i6.9638

[9] Apriyanti, K., & Widodo, T. W. (2016). Implementasi Optical Character Recognition Berbasis *Backpropagation* untuk Text to Speech Perangkat Android. *IJEIS*, *6*(1), 13–24.

[10] Ghozali, I., & Adikara, P. P. (2018). Implementasi Metode *Backpropagation* untuk Mengenali Teks pada Natural Scene Image. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, *2*(8), 2527–2533.

[11] Masrani, H., Ruslianto, I., & Ilhamsyah. (2018). Aplikasi Pengenalan Pola Pada Huruf Tulisan Tangan Menggunakan Jaringan Saraf Tiruan Dengan Metode Ekstraksi Fitur Geometri. *Coding, Sistem Komputer Untan*, *06*(02), 69–78. http://jurnal.untan.ac.id/index.php/jcskommipa/article/view/26674

[12] Mehdi, S. A., & Sciences, E. (2007). Sign Language Recognition using Sensor Gloves Yasir Niaz Khan. *Neural Information Processing*, *5*, 2204–2206.

[13] Novandra, G., Nafan, M. Z., & Laksana, T. G. (2018). Perancangan aplikasi android identifikasi tanda tangan menggunakan multi layer perceptron. *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *03*(01), 76–83.

[14] Putri, F. P., & Kusnadi, A. (2018). Pengenalan Tulisan Tangan Offline Dengan Metode Generalized Hough Transform dan *Backpropagation*. *Jurnal ULTIMA Computing*, *10*(1), 5–12. https://doi.org/10.31937/sk.v10i1.890