PENGENALAN KARAKTER HIJAIYAH MENGGUNAKAN METODE CONVOLUTIONAL NEURAL NETWORK

Ryan Eka Wiratna, Aghil Sahputro, Benny Danendra H, Eva Yulia Puspaningrum
[1234]Universitas Pembangunan Nasional "Veteran" Jawa Timur
Email: 17081010082@student.upnjatim.ac.id

Abstrak. *Penulisan karakter Hijaiyah secara manual menemui permasalahan dimana terdapat banyak variasi penulisan tangan oleh setiap manusia. Solusi dari permasalahan tersebut dapat diatasi dengan melakukan klasifikasi penulisan tangan secara otomatis. Dalam klasifikasi tersebut tentunya membutuhkan metode yang tepat agar dapat dilakukan secara cepat dan tepat. Dalam karya ini, kami memodelkan arsitektur pembelajaran yang dapat mengenali penulisan karakter arab yang variatif dari tangan manusia. Convolutional Neural Network (CNN) adalah salah satu algoritma Deep Learning yang dapat menerima input gambar dan menentukan aspek atau obyek apa yang terdapat pada sebuah gambar. Algoritma CNN dilatih menggunakan dataset. Pada penelitian ini menggunakan dataset yang terdiri dari 16.800 karakter dari 60 partisipan. Database yang digunakan terbagi menjadi 2 set yaitu training set (13.400 karakter terbagi menjadi 480 gambar per class) dan test set (3.360 karakter terbagi menjadi 120 gambar per class). Arsitektur yang kami buat dapat menghasilkan nilai akurasi 94,94% pada epoch ke 100.*

Kata Kunci: Penulisan Karakter Hijaiyah, Deep Learning, Convolutional Network

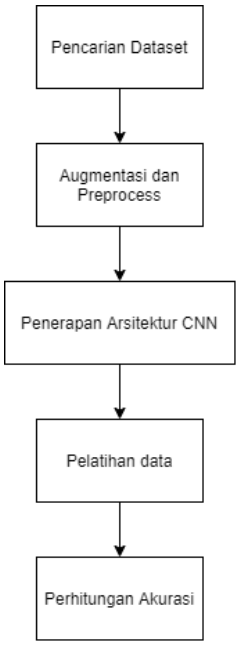
Pengenalan merupakan bidang yang meliputi banyak bidang. Seperti pengenalan wajah, pengenalan sidik jari, pengenalan gerak, pengenalan gambar, pengenalan angka, pengenalan karakter dll. Sistem pengenalan karakter tulisan tangan merupakan sistem cerdas yang mampu mengklasifikasikan karakter tulisan tangan seperti yang dilihat manusia[1]. Pengenalan karakter tulisan tangan merupakan sebuah proses yang sulit dikarenakan terdapat banyak gaya penulisan yang mempengaruhi data karakter yang akan dikenali. Penulisan karakter Arab merupakan salah satu tantangan yang sampai saat ini masih terjadi. Dikarenakan terdapat banyak model penulisan karakter Arab yang ada di dunia. Penulisan karakter hijaiyah digunakan pada Bahasa Arab yang merupakan bahasa yang paling banyak digunakan di dunia pada urutan ke-5[2]. Hal tersebut dapat menunjukkan bahwa karakter hijaiyah terdapat banyak variasi yang dibuktikan dengan jumlah penggunaan Bahasa Arab yang dimana menggunakan karakter hijaiyah. Klasifikasi penulisan karakter hijaiyah dapat dilakukan secara otomatis yang dimana nantinya akan mempercepat waktu klasifikasi dan menghasilkan hasil yang tepat. Proses klasifikasi yang dilakukan secara otomatis mendapat tantangan berupa penggunaan metode/algoritma yang digunakan dapat mempengaruhi waktu dan hasil proses klasifikasi. Pemilihan metode/algoritma yang tepat dapat memberikan hasil yang baik. Pada karya ini menggunakan algoritma *convolutional neural network* (CNN) untuk melakukan klasifikasi penulisan karakter hijaiyah.

Penggunaan algoritma CNN pada penelitian ini berdasarkan penggunaan algoritma CNN dalam beberapa penelitian yang sudah dilakukan untuk melakukan klasifikasi citra. Pada penelitian pengenalan pola karakter bahasa jepang hiragana menghasilkan akurasi sebesar 96,2%[3]. Pada penelitian klasifikasi motif batik menggunakan convolutional network menghasilkan rata-rata akurasi sebesar 70%[4]. Pada penelitian klasifikasi Bati Riau dengan menggunakan Convolutional Network (CNN) menghasilkan akurasi sebesar 65%[5].

Pada penelitian ini kami menggunakan 3 arsitektur yang dimana 2 arsitektur buatan kami yang hampir sama dan juga arsitektur terkenal yaitu VGG-16 yang dibuat oleh Karen Simonyan dan Andrew Zisserman[6]. Pada penelitian ini kami menggunakan *dataset* AHCD yang digunakan pada penelitian yang dilakukan oleh Ahmad El-Sawy, Mohamed Loey, dan Hazem EL-Bakry[1]. Penelitian ini menghasilkan akurasi dengan arsitektur VGG-16 sebesar 3,33% dan dengan arsitektur buatan kami versi 2 dengan epoch terbaik sebesar 94,94%.

I. Metodologi

Alur metodologi yang digunakan pada penelitian ini yaitu pencarian dataset, augmentasi dan preprocess, penerapan arsitektur CNN, pelatihan data dan perhitungan akurasi berdasarkan hasil dari pelatihan yang dapat digambarkan pada Gambar 1.



Gambar 1. Metode Penelitian

Pencarian *Dataset*

Dataset yang digunakan pada penelitian ini terdiri dari 16.800 karakter yang ditulis oleh 60 orang yang berbeda. Rentang usia dari penulis yaitu 19 tahun hingga 40 tahun. 90% dari keseluruhan penulis yaitu penulis dengan tangan kanan. Setiap peserta menulis setiap karakter arab sejumlah 28 (tanpa hamzah dan lam alif) sebanyak 10 kali. *Dataset* ini dibuat oleh Ahmed El-Sawy, Mohamed Loey, Hazem EL-Bakry pada penelitian mereka[1]. Data yang terkumpul terbagi menjadi 2 bagian, yaitu 1 set pelatihan (13.440 karakter yang terdiri dari 480 gambar per kelas) dan 1 set pengujian (3.360 karakter yang terdiri 120 gambar per kelas). Data yang terkumpul memiliki keberagaman diantaranya gaya penulisan, ketebalan tulisan, dan posisi dari penulisan.

Augmentasi dan Preprocess

Augmentasi merupakan sebuah proses pengolahan data gambar yang dimana terdapat proses mengubah atau memodifikasi gambar yang menghasilkan gambar yang berbeda saat dibaca komputer, namun sebenarnya gambar tersebut merupakan gambar yang sama apabila dilihat oleh mata manusia[7]. Augmentasi dapat meningkatkan akurasi dari hasil penerapan arsitektur CNN dikarenakan terdapat data tambahan yang dapat melatih arsitektur agar dapat mengenali lebih banyak variatif gambar.

Pada penelitian ini terdapat augmentasi data berupa pergeseran rentang lebar, pergeseran rentang tinggi, membalik gambar secara horizontal, perputaran gambar, pergeseran gambar, perbesaran gambar. Pergeseran rentang lebar menggunakan *argument width_shift_range* dengan nilai 0,1. *Width_shift_range* berguna untuk melakukan pemindahan semua piksel ke arah horizontal dengan tetap menjaga dimensi gambar tetap sama[8]. Nilai 0,1 berarti melakukan pergeseran sebesar -100px hingga +100px secara acak. Pergeseran rentang tinggi menggunakan *argument height_shift_range* dengan nilai 0,1. *Height_shift_range* berguna untuk melakukan pemindahan semua piksel ke arah vertikal dengan tetap menjaga dimensi gambar tetap sama[8].

Membalik gambar secara horizontal menggunakan *argument horizontal_flip*. Perputaran gambar dilakukan dengan *argument rotation_range* dengan nilai 40 yang berarti perputaran dilakukan dengan besara 40 derajat. Pergeseran gambar menggunakan argument *shear*_range dengan nilai 0,1. *Shear_range* digunakan untuk memperbaiki sudut persepsi[9]. Digunakan untuk menambah gambar sehingga komputer dapat melihat bagaimana manusia melihat gambar tersebut dari sudut yang berbeda. Perbesaran gambar menggunakan *argument zoom_range* dengan nilai 0,1. Berarti dilakukan perbesaran gambar antara 90% perbesaran hingga 110% pengecilan. Pengisian area kosong (*filling)* menggunakan metode *nearest* yang berarti mengisi piksel yang kosong dengan piksel yang terdekat.

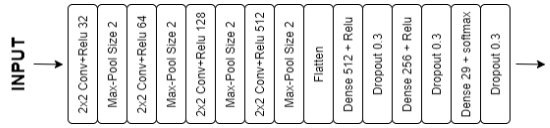
Penerapan Arsitektur CNN

CNN merupakan salah satu metode *deep learning* yang digunakan untuk mengklasifikasikan citra. Peran CNN dipegang oleh *convolution layer* yang digunakan untuk mengekstrak fitur dari sebuah citra dengan *filter* tertentu yang memiliki banyak variasi[10].

Pada CNN terdapat banyak arsitektur terkenal yang dapat digunakan. Salah satunya yang digunakan pada penelitian ini yaitu arsitektur VGG-16. VGG merupakan kepanjangan dari *Visual Geometric* Group yang dikembangkan oleh Oxford University dan VGG-16 memiliki arti terdapat 16 layer yang terdapat pada *Visual Geometric Group*[11]. Pada arsitektur VGG16 terdapat 16 *convolution layer* dengan *kernel* sebesar 3x3, *stride* sebesar

1, dan ReLu untuk setiap *hidden layer*. *Convolution layer* akan diikuti dengan 5 *Max Pooling layer*. *Max Pooling layer* yang digunakan memiliki *kernel* sebesar 2x2 dan *stride* sebesar 2. Kemudian akan diikuti dengan 3 *Fully Connected layer* dan *layer* terakhir menggunakan *softmax activation* untuk proses klasifikasi. Selain itu pada penelitian ini juga menggunakan arsitektur buatan sendiri yang memiliki 2 variasi berbeda. Pembeda dari 2 arsitektur tersebut terletak pada nilai *filters* yang digunakan pada *convolution layer*.

Pada arsitektur buatan kami versi pertama terdapat 8 *convolution layer* dengan *kernel* sebesar 2x2 dan ReLu untuk setiap *hidden layer*. *Convolution layer* akan diikuti dengan 4 *Max Pooling layer*. *Max Pooling layer* yang digunakan memiliki *kernel* sebesar 2x2 dengan *pool size* 2x2. Kemudian akan diikuti dengan 4 *fully connected layer* dan *layer* terakhir menggunakan *softmax activation* untuk proses klasifikasi. Pada arsitektur versi kedua terdapat perbedaan dimana terdapat penambahan pada *convolution layer* sebanyak 2, pada *Max Pooling layer* sebanyak 1 dan pada *fully connected layer* sebanyak 1. Penggambaran arsitektur versi 1 buatan dapat digambarkan pada Gambar 2 sedangkan versi 2 pada Gambar 3.



Gambar 2. Arsitektur Buatan Versi 1

Gambar 3. Arsitektur Buatan Versi 2.

Pelatihan Data

Pada proses pelatihan data ini terdapat inisialisasi parameter yaitu fungsi *loss* menggunakan *categorical cross-entropy*, menggunakan *optimizer* yaitu *adam*, metrik berupa akurasi, menggunakan *batch size* 128, dan di penelitian ini *epoch* yang digunakan bervariatif yang dimana bertujuan agar dapat menemukan epoch yang akan memberikan hasil yang optimal. *Categorical Cross-entropy* merupakan fungsi *loss* yang digunakan untuk masalah klasifikasi multi-kelas. Cross-entropy akan menghitung skor yang merangkum perbedaan rata-rata antara distribusi probabilitas aktual dan prediksi untuk semua kelas dalam masalah yang diamati[12].

Algoritma optimalisasi Adam merupakan ekstensi dari *stochastic gradient descent* yang baru-baru ini memperoleh adopsi yang lebih luas dimana digunakan untuk *deep learning* dalam visi komputer dan NLP(*Natural Languange Processing*). Pengoptimalisasi Adam dapat digunakan untuk memperbarui *weight network* secara berulang berdasarkan *data training*. Adam merupakan algoritma terbaik yang dimana dapat mencapai hasil yang terbaik dengan cepat[13]. *Batch size* merupakan *hyperparameter* yang digunakan untuk menentukan jumlah sampel yang dikerjakan sebelum memperbarui parameter model internal[14].

Perhitungan Akurasi

Perhitungan akurasi pada penelitian ini digunakan dengan cara menampilkan sebaran hasil perhitungan melalui matriks. Penampilan matriks menggunakan 4 grafik yang dimana dari masing-masing grafik menampilkan perbandingan hasil perhitungan antara lain *loss* pada pelatihan data dengan *epoch*, akurasi pada pelatihan dengan *epoch*, validasi *loss* dengan *epoch* dan validasi akurasi dengan *epoch*.

II. Hasil dan Pembahasan

Pada penelitian ini kami menggunakan *hardware* dengan spesifikasi sistem operasi Windows 11 64-bit, *processor* AMD Ryzen 3 3200U with Radeon Vega Mobile Gfx dan RAM 8GB. Untuk *software* yang digunakan yaitu *visual studio code* sebagai *code editor*, Microsoft Word untuk pengerjaan jurnal dan menggunakan bahasa pemrograman *python*.

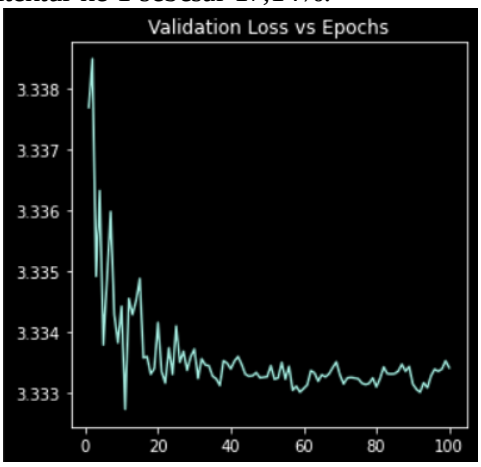
Hasil dari penelitian ini kami tampilkan dalam bentuk tabel yang dimana berisi mengenai hasil akurasi dari 2 variasi arsitektur yang sudah kami buat dan dijelaskan pada sebelumnya. Berikut data hasil uji coba kami :

Tabel 1. Data hasil uji

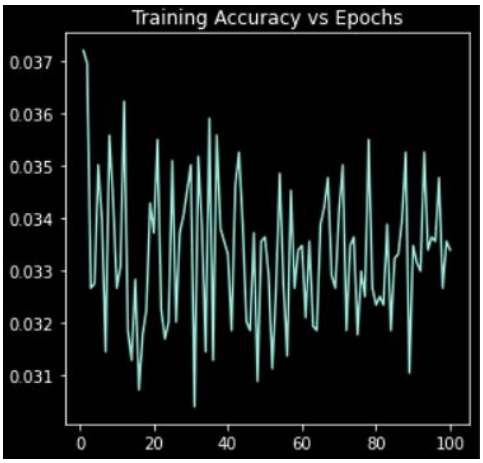
No	Epoch	Variasi	Hasil Akurasi (dalam %)	Total *test lost* (dalam %)
1	10	1	79,25	60,75
		2	74,4	72.8
2	20	1	86,01	39,14
		2	81,87	59,08
3	30	1	89,79	29,95
		2	88,45	36,88

4	40	1	92,47	23,41
		2	84,1	52,43
5	50	1	93	21,34
		2	89,85	31,55
6	60	1	92,65	24,8
		2	94,31	20,2
7	70	1	92,8	20,74
		2	93,12	22,81
8	80	1	92,38	24,83
		2	92,41	24,2
9	90	1	93,3	23,5
		2	94,28	20,91
10	100	1	94,8	17,14
		2	94,94	17,46

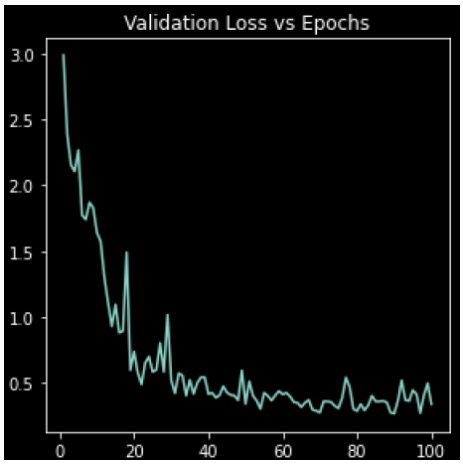
Dapat dilihat pada Tabel 1 dimana menampilkan hasil akurasi dan total *test error* dengan beragaman variasi. Hasil akurasi dan total *test loss* fluktuatif dikarenakan proses augmentasi data sebelumnya yang menghasilkan data pelatihan yang acak. Untuk hasil uji dari arsitektur VGG-16 menghasilkan akurasi sebesar 3,57% dan total *test loss* 333,23% yang dimana berarti arsitektur variasi ke-2 buatan penulis memiliki hasil lebih baik dengan angka akurasi 94.94% dan total *test loss* 17,46%. Berikut untuk visualisasi pada grafik untuk perhitungan akurasi dan total *test loss* dengan *epoch*. Dapat dilihat apabila hasil akurasi tertinggi terjadi pada epoch ke-100 menggunakan variasi arsitektur ke-2 dengan hasil akurasi sebesar 94,94% dengan total *test loss* sebesar 17,46%. Untuk total *test loss* terkecil terdapat pada *epoch* 100 dengan variasi arsitektur ke-1 sebesar 17,14%.



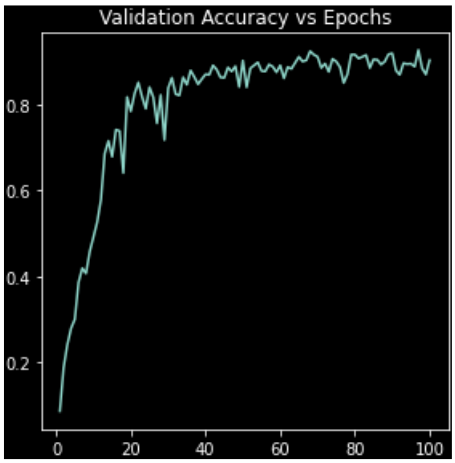
Gambar 4. Grafik *Loss test* dengan *epoch* pada arsitektur VGG-16.



Gambar 5. Grafik akurasi dengan *epoch* pada arsitektur VGG-16.



Gambar 6. Grafik *loss test* dengan *epoch* pada *epoch* 100 dengan arsitektur versi 2.



Gambar 7. Grafik akurasi dengan *epoch* pada *epoch* 100 dengan arsitektur versi 2.

III. Kesimpulan

Penggunaan arsitektur CNN dalam pengenalan karakter atau gambar sudah banyak dilakukan. Terdapat banyak variasi arsitektur yang sudah dibuat. Pada penelitian ini berhasil membuat arsitektur CNN dengan sedikit penambahan pada arsitektur nya yang menghasilkan akurasi terbesar sebesar 94,94% dan juga *loss test* terkecil sebesar 17,14%. Kemungkinan besar hasil akurasi dan juga *test loss* masih dapat menghasilkan yang lebih baik apabila terdapat pengaturan arsitektur dan juga pemilihan jumlah *epoch* yang lebih bervariatif dan juga dapat dilakukan penambahan data yang dikumpulkan melalui orang yang nyata sehingga tidak perlu melakukan augmentasi yang pada akhirnya murni pengenalan dari tulisan tangan manusia

IV. Daftar Pustaka

[1] H. E.-B. Ahmed El-Sawy, Mohamed Loey, "Arabic Handwritten Characters Recognition Using Convolutional Neural Network," *2021 12th Int. Conf. Inf. Commun. Syst. ICICS 2021*, no. January, pp. 182–188, 2021, doi: 10.1109/ICICS52457.2021.9464596.

[2] C. AnnurMutia, "Inilah Bahasa yang Paling Banyak Dipakai di Dunia, Bagaimana Bahasa Indonesia?," *Katadata*, 2021. https://databoks.katadata.co.id/datapublish/2021/11/01/inilah-bahasa-yang-paling-banyak-dipakai-di-dunia-bagaimana-bahasa-indonesia.

[3] M. M. Susilo, D. M. Wonohadidjojo, and N. Sugianto, "Pengenalan Pola Karakter Bahasa Jepang Hiragana Menggunakan 2D Convolutional Neural Network," *J. Inform. dan Sist. Inf. Univ. Ciputra*, vol. 03, no. 02, pp. 28–36, 2017.

[4] R. Mawan, "Klasifikasi motif batik menggunakan Convolutional Neural Network," *Jnanaloka*, pp. 45–50, 2020, doi: 10.36802/jnanaloka.2020.v1-no1-45-50.

[5] H. Fonda, "Klasifikasi Batik Riau Dengan Menggunakan Convolutional Neural Networks (Cnn)," *J. Ilmu Komput.*, vol. 9, no. 1, pp. 7–10, 2020, doi: 10.33060/jik/2020/vol9.iss1.144.

[6] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[7] K. H. Mahmud, Adiwijaya, and S. Al Faraby, "Klasifikasi Citra Multi-Kelas Menggunakan Convolutional Neural Network," *e-Proceeding Eng.*, vol. 6, no. 1, pp. 2127–2136, 2019.

[8] J. Brownlee, "How to Configure Image Data Augmentation in Keras," *Machine Learning Mastery*, 2019. https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/.

[9] Stackoverflow, "What exactly the shear do in ImageDataGenerator of Keras?," 2020. https://stackoverflow.com/questions/57301330/what-exactly-the-shear-do-in-imagedatagenerator-of-keras.

[10] S. ilham Pradika, B. Nugroho, and E. Y. Puspaningrum, "Pengenalan Tulisan Tangan Huruf Hijaiyah Menggunakan Metode Convolutional Neural Network," *Semin. Nas. Inform. Bela Negara*, vol. 1, p. 98, 2020.

[11] D. Subroto and L. Liliana, "Deteksi Aktivitas Manusia Berdasarkan Data Skeleton dengan Menggunakan Modifikasi VGG16," *J. Infra*, vol. 9, no. 1, pp. 122–128, 2021, [Online]. Available: http://publication.petra.ac.id/index.php/teknik-informatika/article/view/10933.

[12] J. Brownlee, "How to Choose Loss Functions When Training Deep Learning Neural Networks," *Machine Learning Mastery*, 2019. https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/.

[13] J. Brownlee, "Gentle Introduction to the Adam Optimization Algorithm for Deep Learning," *Machine Learning Mastery*, 2017. https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/.

[14] J. Brownlee, "Difference Between a Batch and an Epoch in a Neural Network," *Machine Learning Mastery*, 2018. ttps://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/.