

Penerapan Convolutional Neural Network untuk Handwriting Recognition pada Aplikasi Belajar Aritmatika Dasar Berbasis Web

Jayaku Brilliantio¹, Nico Santosa², Garvin Ardian³, Lukman Hakim⁴

^{1,2,3,4}Program Studi Teknik Informatika, Universitas Bunda Mulia, Jakarta

e-mail: 1jayakubriliantio@gmail.com, 2nico.santosa1@gmail.com, 3garvinardian132@gmail.com,
4lhakim2710@gmail.com

Abstrak

Aritmatika merupakan cabang ilmu matematika yang berhubungan dengan angka, pengukuran, dan komputasi numerik seperti penjumlahan, pengurangan, perkalian, dan pembagian. Mengajar aritmatika memiliki tantangan tersendiri bagi pengajar. Pada umumnya pengajaran aritmatika bersifat satu arah, sehingga bersifat monoton dan kemampuan dalam mengingat materi menjadi rendah. Salah satu cara untuk meningkatkan daya ingat dan penyerapan materi yang disampaikan adalah dengan menulis. Pada penelitian ini dirancang suatu aplikasi berbasis web belajar aritmatika dengan menulis. Untuk mengenali tulisan digital berupa angka dan operator aritmatika dibutuhkan handwriting recognition system. Convolutional Neural Network (CNN) dapat melakukan pengenalan tulisan tangan dengan tepat, baik yang bersifat off-line maupun online. Dataset diperlukan dalam training model CNN untuk mampu mengenal tulisan. Bobot yang diperoleh dari hasil training model CNN akan diintegrasikan dengan aplikasi. Melalui penelitian ini, dapat diketahui bahwa CNN memiliki tingkat akurasi yang baik dalam mengenali dan mengklasifikasikan tulisan tangan. Tingkat akurasi dari CNN dalam mengenal tulisan tangan yang diperoleh dari hasil pengujian adalah 95.36%.

Kata kunci— pengenalan tulisan tangan, convolutional neural network, jaringan saraf tiruan

Abstract

Arithmetic is a branch of mathematics that related with number, measurement, and numerical computations such as addition, subtraction, multiplication, and division. Teaching arithmetic is also a challenge for teachers. In general, arithmetic is taught by the academics in one-way teaching, so it is monotonous and the ability to memorize the given material is decreased. There is one way to increased the memory or retention ability and theory understanding is by writing. In this research, a web-based application to learn arithmetic by writing was designed. To recognize digital writing in the form of numbers and arithmetic operators, a handwriting recognition system is needed. Convolutional Neural Network(CNN) can carry out precise handwriting recognition for both off-line and online handwriting. The dataset is needed to train CNN Model to be able to recognize the handwriting. The weight from CNN model training will be integrated with the application. Through this research, it can be seen that CNN has a good level of accuracy in classifying the handwriting. The accuracy rate of CNN in recognizing handwriting obtained from the test result is 95.63%.

Keywords— handwriting recognition, convolutional neural network, neural network

1. PENDAHULUAN

Aritmatika merupakan salah satu cabang ilmu matematika. Menurut C.C. MacDuffee, aritmatika merupakan cabang ilmu matematika yang berhubungan dengan angka, relasi antar angka, dan observasi pada angka yang dipelajari dan digunakan untuk menyelesaikan suatu permasalahan. Aritmatika merujuk kepada aspek mendasar dari teori angka, pengukuran, dan komputasi numerik seperti penjumlahan, pengurangan, perkalian, pembagian [1].

Pendidikan di bidang aritmatika sudah dikenalkan sejak dini ketika memasuki dunia pendidikan formal [2]. Pemahaman mendasar terhadap aritmatika menjadi sangat penting karena aritmatika digunakan dalam kehidupan sehari-hari. Selain itu, dengan memiliki pemahaman aritmatika yang baik dapat meningkatkan pemahaman konsep matematika yang lebih rumit seperti aljabar [3], [4]. Dibalik

pentingnya pemahaman tentang aritmatika, tidak sedikit penyampaian aritmatika dalam proses belajar mengajar menjadi sebuah tantangan tersendiri bagi pengajar [2]. Proses belajar mengajar di bidang aritmatika dan matematika pada umumnya bersifat informatif dan satu arah yang menyebabkan kemampuan mengingat terhadap materi menjadi rendah [5].

Salah satu alternatif yang dapat digunakan untuk meningkatkan daya ingat adalah menulis. Menulis merupakan salah satu teknik yang dapat meningkatkan kemampuan daya ingat jangka panjang (*long term memory*) dan kemampuan dalam mengenali sesuatu hal jika dibandingkan dengan membaca maupun mendengarkan secara lisan [6]. Teknik pembelajaran yang bersifat interaktif juga dapat meningkatkan semangat dan minat dalam mempelajari aritmatika [7], [8]. Seiring dengan perkembangan teknologi, metode pembelajaran yang bersifat interaktif dengan teknik menulis dapat diimplementasikan melalui gawai dan internet.

Sistem pengenalan karakter dibutuhkan untuk mengenali karakter yang dituliskan oleh pengguna melalui gawai. Penggunaan *deep learning* pada zaman ini sangatlah meningkat. Salah satu arsitektur *deep learning* yaitu *Convolutional Neural Network* (CNN) dapat digunakan untuk pengenalan pola seperti karakter, angka, dan penyelesaian kasus-kasus *computer vision* [9]. CNN dapat digunakan sebagai basis dalam membangun sistem pengenalan karakter yang ditulis oleh manusia [10].

2. METODE PENELITIAN

Dalam penelitian ini dibangun suatu aplikasi belajar aritmatika dasar berbasis web dengan sistem pengenalan tulisan tangan (*handwriting recognition system*) dengan algoritma *Convolutional Neural Network* (CNN).

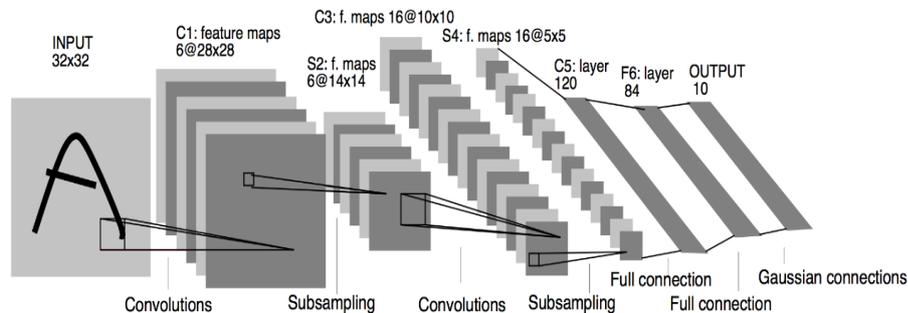
2.1 Handwriting Recognition System

Handwriting recognition adalah sebuah proses untuk mengkonversikan tulisan tangan ke dalam bentuk yang dapat dibaca dan dipahami oleh mesin [11]. *Handwriting recognition* dibagi menjadi dua jenis yaitu *offline handwriting recognition* dan *online handwriting recognition*. *Online handwriting recognition* adalah jenis tulisan tangan yang ditulis dengan menggunakan *touch pad* atau menggunakan *stylus pen*, sedangkan *offline handwriting recognition* adalah jenis tulisan yang tersimpan dalam format gambar, seperti hasil scan tulisan tangan dari manusia [11], [12].

2.2. Convolutional Neural Network (CNN)

CNN merupakan salah satu arsitektur model pada *deep learning* yang terinspirasi dari kemampuan visual natural dari makhluk hidup. Neocognitron yang dikembangkan oleh Kunihiko Fukushima pada tahun 1980 dianggap sebagai pendahulu dari CNN [13]. Kemudian dilanjutkan pada tahun 1990, LeCun et al. mempublikasikan sebuah penelitian yang menjadi salah satu fundamental dari CNN [14].

Pada tahun 1998, LeCun et al., mengembangkan LeNet-5 yang digunakan dalam pengembangan sistem pengenalan dokumen [15]. Pada tahun 2012, Alex Krizhevsky berhasil memenangkan lomba *ImageNet Large Scale Visual Recognition Challenge 2012* [16]. Hal ini menjadi bukti nyata kiprah dari metode *Deep Learning* seperti *CNN*, dapat mengungguli metode *Machine Learning* lainnya.



Gambar 1 Arsitektur LeNet-5, salah satu CNN oleh Yan LeCunn

Hingga saat ini, terdapat berbagai macam arsitektur dari CNN, dimulai dari tahun 1998 yaitu LeNet-5, AlexNet pada tahun 2012, VGGNet, GoogLeNet, ResNet. Namun, pada umumnya

keseluruhan arsitektur CNN dibangun dengan menggunakan component yang serupa. Berdasarkan pada LeNet-5, CNN secara umum mencakup tiga bagian utama (*layers*) [17],[15] sebagai berikut:

1. *Convolution Layer*

Lapisan konvolusi bertujuan untuk mempelajari representasi fitur yang dapat diperoleh dari *input*. Sebuah lapisan konvolusi merupakan hasil penggabungan dari beberapa kernel konvolusi dimana setiap dari kernel tersebut dapat menangkap fitur yang berbeda sehingga menghasilkan matriks fitur hasil operasi konvolusi yang beragam. Matriks fitur diperoleh dari berbagai macam kernel. Secara matematis, nilai fitur pada koordinat $[i, j]$ dari fitur k pada lapisan l , direpresentasikan sebagai berikut:

$$z_{i,j,k}^l = w_k^{lT} x_{i,j}^l + b_k^l \quad (1)$$

Setelah melakukan proses operasi konvolusi, maka dilakukan proses aktivasi dengan menggunakan fungsi yang bersifat non-linear. Proses aktivasi dengan fungsi *non-linear* bertujuan untuk menangkap fitur yang bersifat non-linear. Fungsi aktivasi yang digunakan pada umumnya adalah *sigmoid*, *ReLU* [18], dan *tanh* [19]. Nilai aktivasi dari fitur pada koordinat $[i, j]$ dinotasikan dengan $a_{i,j,k}^l$ dikomputasikan dengan:

$$a_{i,j,k}^l = g(z_{i,j,k}^l) \quad (2)$$

2. *Subsampling Layer*

Subsampling atau operasi *pooling* akan dioperasikan pada setiap matriks fitur setelah proses aktivasi untuk melakukan reduksi dimensi. Pada umumnya layer *pooling* diletakkan diantara dua lapisan konvolusi. Terdapat dua jenis operasi *pooling* secara garis besar yaitu *Average Pooling* dan *Max Pooling*. Secara matematis, operasi *pooling* direpresentasikan sebagai berikut:

$$y_{i,j,k}^l = \text{pool}(a_{m,n,k}^l), \forall (m, n) \in R_{i,j} \quad (3)$$

dimana $R_{i,j}$ adalah area sekitar koordinat $[i, j]$.

3. *Fully-Connected Layer*

Setelah melalui operasi konvolusi dan subsampling, matrix fitur akan di-*flatten* sehingga dapat digunakan sebagaimana seperti *multilayer perceptron* (MLP), atau dikenal sebagai *fully-connected neural network* [20]. Proses ini akan menarik semua neuron dari lapisan sebelumnya dan menghubungkan setiap neuron ke lapisan saat ini untuk menghasilkan informasi semantik yang bersifat global dan dapat digunakan untuk melakukan klasifikasi.

Proses CNN dimulai dengan adanya proses operasi konvolusi (*affine transformation*). Setelah itu akan dilakukan aktivasi terhadap hasil operasi konvolusi tersebut. Kemudian dilanjutkan dengan proses *subsampling* atau *pooling*. Fitur matriks hasil *subsampling* akan dilakukan proses *flatten* agar sehingga membentuk suatu *fully-connected layer* (atau *multilayer perceptron*) [20]. Dalam kasus klasifikasi pada umumnya digunakan fungsi *softmax*. Fungsi *softmax* berfungsi untuk menghasilkan probabilitas dari masing-masing kelas target, Fungsi *softmax* didefinisikan sebagai berikut [17]:

$$p_j^{(i)} = \frac{e^{z_j^{(i)}}}{\sum_{l=1}^K e^{z_l^{(i)}}} \quad (4)$$

$p_j^{(i)}$ merupakan prediksi pada kelas ke-j dengan data *training* ke-i, K merupakan jumlah kelas pada y target, dan $z_j^{(i)} = w_j^T a^{(i)} + b_j$.

Fungsi *softmax* mengubah prediksi menjadi nilai negatif dan melakukan normalisasi untuk mendapatkan probabilitas yang terdistribusi dari keseluruhan kelas yang ada pada y target. Untuk melakukan *backpropagation* pada *neural network*, diperlukan nilai *loss* pada *output layer* antara hasil prediksi dan nilai yang nyata. Persamaan *softmax loss* sebagai cara untuk melakukan perhitungan *multinomial logistic loss* atau *cross entropy loss* adalah sebagai berikut [17]:

$$L_{softmax} = - \frac{1}{N} [\sum_{i=1}^N \sum_{j=1}^K 1 \{y^{(i)} = j\} \log p_j^{(i)}] \quad (5)$$

N merupakan jumlah baris data *training*. K merupakan jumlah kelas yang pada y target, $p_j^{(i)}$ merupakan prediksi pada kelas ke- j dengan data *training* ke- i , $y^{(i)}$ adalah kelas target atau label dari data *training* ke- i .

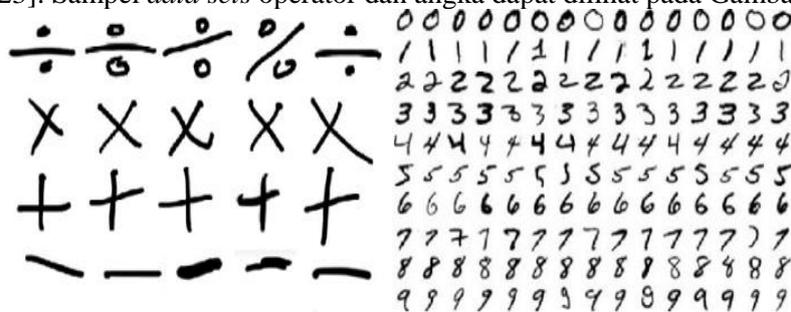
Persoalan tentang *overfitting* sering terjadi dalam melakukan *training* model CNN. Untuk mengurangi *overfitting* dan meningkatkan performa dari model CNN dapat digunakan sebuah algoritma regularisasi yang bernama *Dropout* [21], [22]. Regularisasi *Dropout* memiliki performa yang baik dalam mengurangi *overfitting* pada kasus *computer vision*. Pada regularisasi *dropout*, *neuron* yang terdapat pada CNN akan dinonaktifkan secara acak pada tahap *training* dengan mengacu pada hukum distribusi *Bernoulli*. *Neuron* ini kemudian akan diaktifkan kembali pada tahap *testing* [20].

2.3. Rancangan Pengembangan Handwriting Recognition System

Perancangan sistem pengenalan tulisan tangan ini dilakukan secara paralel bersama dengan pengembangan aplikasi. Pengembangan sistem pengenalan tulisan tangan menggunakan framework Tensorflow. Berikut adalah tahapan yang dibutuhkan dalam mengembangkan sistem ini:

1. Pengumpulan data

Pada tahap ini dilakukan pengumpulan *data sets* berupa *image* dari tulisan tangan angka 0-9 dan operator aritmatika. *Data sets* angka yang digunakan adalah *data sets* tulisan tangan MNIST dari Yan LeCun [23]. Sampel *data sets* operator dan angka dapat dilihat pada Gambar 3.



Gambar 2 Sampel *data sets* operator aritmatika (kiri) dan *data sets* MNIST (kanan)

2. Preprocessing data

Pada tahap ini, data *image* yang telah terkumpulkan akan dilakukan *preprocessing data*. *Preprocessing data* yang dilakukan meliputi *data augmentation* dan *normalization*. Selain itu data juga akan dibagi menjadi *training data* yang akan digunakan pada pelatihan CNN, sedangkan *test data* akan digunakan sebagai validasi.

3. Perancangan arsitektur CNN

Pada tahap ini, akan dilakukan penentuan arsitektur dari CNN, yang berkaitan dengan jenis model, jumlah lapisan konvolusi, lapisan *subsampling*, dan jumlah lapisannya. *Hyperparameter* dari CNN seperti *padding*, *stride*, jenis *activation function* juga harus ditentukan. Arsitektur CNN yang digunakan untuk penelitian ini dapat dilihat pada Gambar 4 dan Gambar 5.

Tabel 1 Arsitektur CNN untuk klasifikasi tulisan angka

Layer	Ukuran	Dimensi Keluaran
Input	(28, 28, 1)	-
Convolutional 2D + ReLu	32 (3 x 3) filters	(None, 28, 28, 32)
Max Pooling 2D	(2 x 2) filter	(None, 14, 14, 32)
Convolutional 2D + ReLu	48 (3 x 3) filters	(None, 14, 14, 48)
Max Pooling 2D	(2 x 2) filter	(None, 7, 7, 48)
Flatten	-	(None, 7 * 7 * 48)
Dense + ReLu	256 perceptrons	(None, 256)
Dropout (rate=0.5)	-	(None, 256)
Dense + Softmax	10 perceptrons	(None, 10)

Tabel 2 Arsitektur CNN untuk klasifikasi tulisan operator

Layer	Ukuran	Dimensi Keluaran
Input	(48, 48, 1)	-
Convolutional 2D + ReLu	32 (3 x 3) filters	(None, 48, 48, 32)
Max Pooling 2D	(2 x 2) filter	(None, 24, 24, 32)
Convolutional 2D + ReLu	48 (3 x 3) filters	(None, 22, 22, 48)
Max Pooling 2D	(2 x 2) filter	(None, 11, 11, 48)
Flatten	-	(None, 11 * 11 * 48)
Dense + ReLu	256 perceptrons	(None, 256)
Dropout (rate=0.2)	-	(None, 256)
Dense + Softmax	4 perceptrons	(None, 4)

4. *Training* model CNN

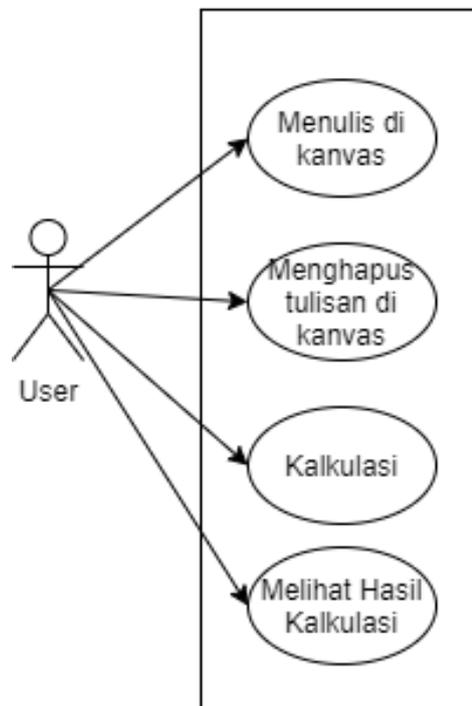
Pada tahap ini, akan ditentukan *learning rate*, jumlah *epoch*, jenis optimasi.

5. Pengujian model CNN

Pada tahap ini, dapat dilakukan pengujian terhadap kemampuan klasifikasi dari model dengan menggunakan *test cases* yang ditentukan, melihat apakah model CNN yang telah dilatih mengalami *overfitting* atau *underfitting*. Jika terjadi *overfitting* atau *underfitting*, maka perlu dilakukan *hyperparameter tuning* atau menambah jumlah dataset, disesuaikan dengan kasus yang terjadi.

2.4. Use Case Aplikasi

User dapat melakukan *input* dengan menulis angka dan operator di kanvas dimana terdapat 3 kanvas yaitu kanvas untuk angka pertama, operator, dan angka kedua. *User* dapat melakukan penghapusan terhadap kanvas yang sudah diisi. *User* dapat melakukan kalkulasi dari angka dan operasi aritmatika yang telah ditulis di kanvas. *User* dapat melihat hasil kalkulasi dari operasi aritmatika yang telah dilakukan sebelumnya. Gambar 6 adalah *use case* dari aplikasi.

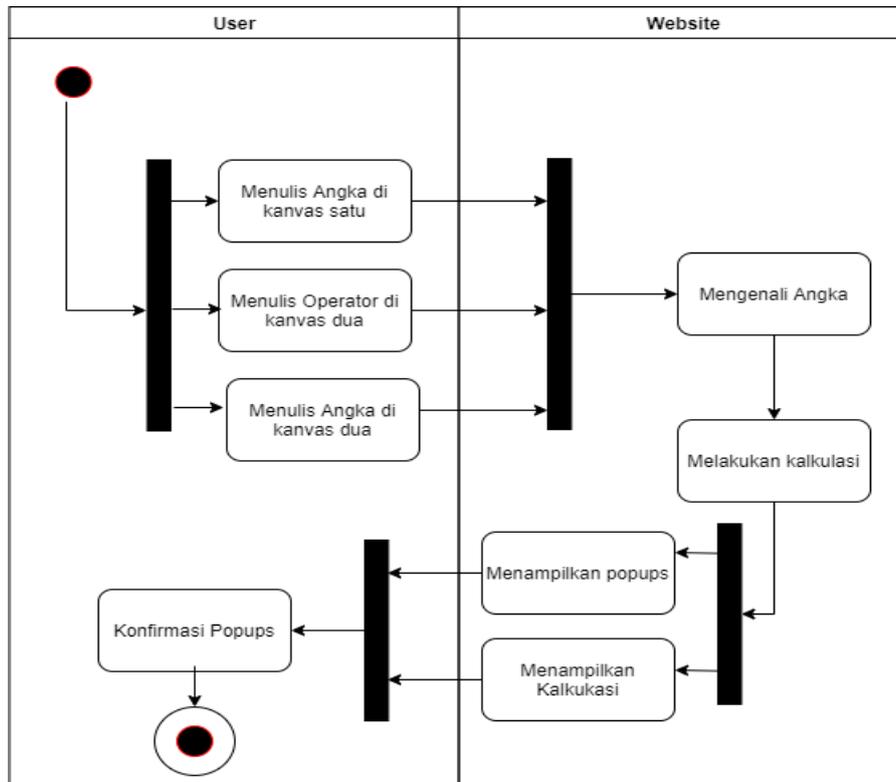


Gambar 3 Use Case Aplikasi

2.5. Activity Diagram Aplikasi

User melakukan masukan berupa tulisan terhadap ketiga kanvas, kanvas pertama untuk angka, kanvas kedua untuk operator, kanvas ketiga untuk angka. Setelah melakukan masukan berupa tulisan maka website akan menerima masukan dari *user* dan dilakukan proses pengenalan tulisan *user* kemudian dilanjutkan dengan melakukan kalkulasi sesuai dengan apa yang berhasil dikenalkan oleh sistem. Sistem

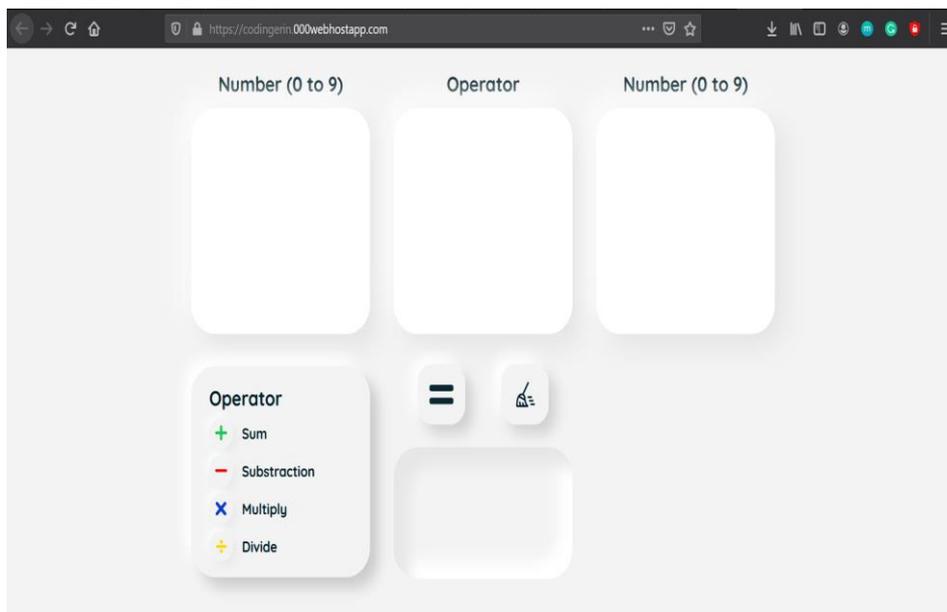
kemudian memberikan *pop ups* berupa hasil pengenalan angka dan operasi yang dimasukan kedalam sistem serta menampilkan hasil kalkulasi ke *user*. *User* kemudian melakukan konfirmasi terhadap *pop ups*. Activity diagram dapat dilihat pada Gambar 7.



Gambar 4 Activity Diagram Interaksi User dan Aplikasi

4. HASIL DAN PEMBAHASAN

3.1. Implementasi User Interface



Gambar 5 User Interface Aplikasi Belajar Aritmatika Dasar

Perancangan desain dari *user interface* menggunakan desain *Neumorphism*. Pada *user interface* terdapat 3 kanvas utama yang berbentuk persegi, dimana *user* dapat menuliskan angka dan operator

Tabel 3 Hasil perhitungan akurasi kebenaran klasifikasi dari *test* data

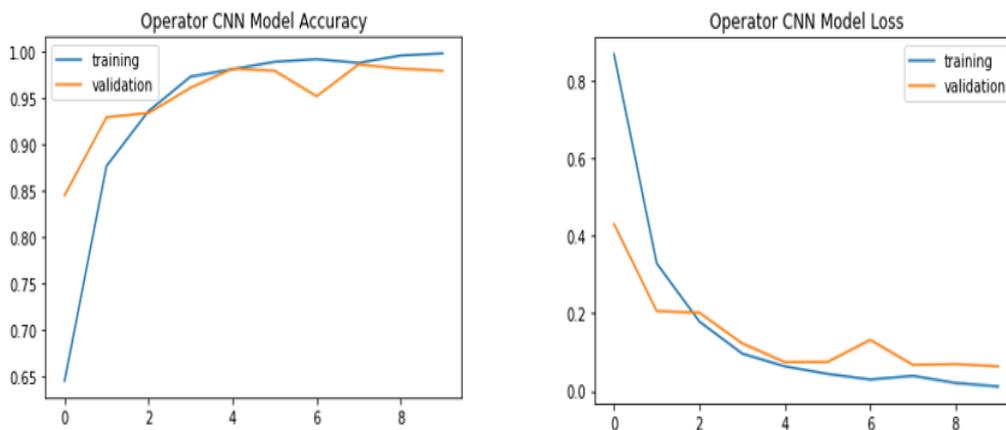
Karakter yang diuji	Berhasil	Gagal	Akurasi
0	19	1	95%
1	18	2	90%
2	19	1	95%
3	19	1	95%
4	20	0	100%
5	18	2	90%
6	19	1	95%
7	18	2	90%
8	18	2	90%
9	19	1	95%
+	20	0	100%
-	20	0	100%
*	20	0	100%
/	20	0	100%
Total Akurasi	280	13	95.36%

Tabel 4 Hasil uji coba perhitungan aritmatika aplikasi

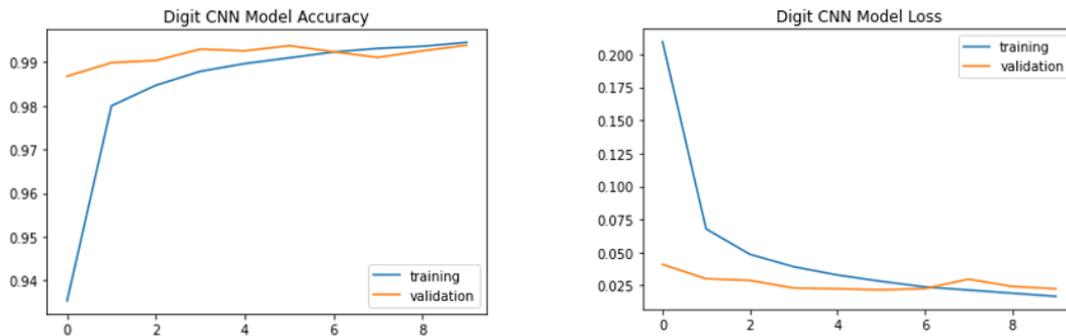
Operasi	Hasil Yang diinginkan	Hasil Program	Kesimpulan
1 / 2	0.5	0.5	Benar
4 * 3	12	12	Benar
5 / 6	0.833	0.83	Benar
7 - 8	-1	-1	Benar
9 - 0	9	9	Benar
7 * 0	0	0	Benar
5 + 6	11	11	Benar
4 + 1	5	5	Benar
9 * 3	27	27	Benar
0 / 0	Tidak Terdefiniskan	<i>Undefined or Infinity</i>	Benar

3.3 Pembahasan Handwriting Recognition System

Sistem *Handwriting Recognition* menggunakan arsitektur *deep learning Convolutional Neural Network* (CNN). Dalam proses pelatihan CNN, hal penting yang perlu diperhatikan adalah bagaimana performa pada proses training CNN dan proses testing atau validasi. Berikut adalah hasil visualisasi dari akurasi dan *cross entropy loss* pada proses training training dan validasi dari model CNN untuk klasifikasi operator (Gambar 11) dan klasifikasi angka (Gambar 12).



Gambar 8 Akurasi dan Loss dari Model CNN klasifikasi operator



Gambar 9 Akurasi dan Loss dari Model CNN klasifikasi angka

Meninjau grafik akurasi dari model CNN untuk klasifikasi operator dan klasifikasi angka, tingkat akurasi pada data training dan data validasi memiliki *gap* (perbedaan jarak) yang kecil dan memiliki nilai akurasi yang tinggi diatas 99%. Hal ini menunjukkan model yang bagus, tidak *underfitting* maupun *overfitting*.

Meninjau grafik *loss* dari model CNN klasifikasi operator dan klasifikasi angka, *loss* dari *training* dan akurasi juga memiliki *gap* yang kecil dan memiliki nilai *loss* yang cukup rendah yaitu dibawah 2,5% sehingga tidak terjadi *underfitting* maupun *overfitting*. Performa dari kedua model CNN menunjukkan hasil yang cukup baik karena tidak adanya perbedaan yang signifikan jika ditinjau dari akurasi dan *loss* dari *training* dan validasinya.

4. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan oleh peneliti, diperoleh kesimpulan sebagai berikut:

1. Algoritma CNN sangat baik dalam melakukan pengenalan tulisan tangan dengan tingkat akurasi rata-rata pada proses training adalah 99% sedangkan pada proses pengujian aplikasi, keberhasilan pengenalan tulisan tangan dari *user* adalah 95.36%.
2. Kegagalan dalam mengenali tulisan tangan dapat disebabkan oleh beberapa hal selain daripada bentuk dari tulisannya, seperti jumlah data dan variasi dari data yang digunakan dalam *training* CNN dan kurang tepatnya penentuan *hyperparameter* dari CNN.

DAFTAR PUSTAKA

- [1] C. C. MacDuffee, "Arithmetic," *Encyclopedia Britannica, inc.*, 2019. [Online]. Available: <https://www.britannica.com/science/arithmetic>. [Accessed: 07-Oct-2020].
- [2] G. Swalaganata, "Pengembangan Media Pembelajaran Game Aritmatika (GAMETIKA) Menggunakan Adobe Flash CS6," *Jurnal Tadris Matematika*, vol. 1, no. 1, May 2018, doi: 10.21274/jtm.2018.1.1.65-76.
- [3] W. D. Pratiwi and E. Kurniadi, "Transisi Kemampuan Berpikir Aritmatika ke Kemampuan Berpikir Aljabar pada Pembelajaran Matematika," *Jurnal Gantang*, vol. 3, no. 1, pp. 1–8, Mar. 2018, doi: 10.31629/jg.v3i1.388.
- [4] A. Wahyuni, "Analisis Kesalahan Siswa dalam Menyelesaikan Soal Aritmatika Sosial," *Jurnal Pendidikan Matematika*, 2020, doi: 10.36709/jpm.v11i1.10022.
- [5] O. R. Dila and L. S. Zanthi, "Identifikasi Kesulitan Siswa dalam Menyelesaikan Soal Aritmatika Sosial," *Teorema: Teori dan Riset Matematika*, vol. 5, no. 1, p. 17, Mar. 2020, doi: 10.25157/teorema.v5i1.3036..
- [6] A. M. Silva and R. Limongi, "Writing to Learn Increases Long-term Memory Consolidation: A Mental-chronometry and Computational-modeling Study of 'Epistemic Writing,'" *Journal of Writing Research.*, vol. 11, no. 1, pp. 211–243, Jun. 2019, doi: 10.17239/jowr-2019.11.01.07.
- [7] F. A. Rizal, B. Suyanto, and T. R. Yudiantoro, "Aplikasi Game Edukasi Matematika Dengan Konsep Aritmatika Anak," *Jurnal Teknik Elektro Terapan*, vol. 5, no. 1, pp. 45–50, 2016.

- [8] M. S. Islam and S. Fahmi, "Pengembangan Media Pembelajaran Interaktif Matematika dengan Menggunakan Macromedia Flash 8 Pada Materi Aritmatika Sosial untuk Siswa SMP Kelas VII Semester Genap," *Seminar Nasional Pendidikan Matematika Ahmad Dahlan*, 2019.
- [9] S. S. Rosyda and T. W. Purboyo, "A Review of Various Handwriting Recognition Methods," *International Journal of Applied Engineering Research*, vol. 13, no. 2, pp. 1155–1164, 2018.
- [10] R. Ptucha, F. Petroski Such, S. Pillai, F. Brockler, V. Singh, and P. Hutkowski, "Intelligent Character Recognition using Fully Convolutional Neural Networks," *Pattern Recognition*, vol. 88, pp. 604–613, Apr. 2019, doi: 10.1016/j.patcog.2018.12.017.
- [11] R. Plamondon and S. N. Srihari, "Online and Off-line Handwriting Recognition: a Comprehensive Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, 2000, doi: 10.1109/34.824821
- [12] A. Priya, S. Mishra, S. Raj, S. Mandal, and S. Datta, "Online and Offline Character Recognition: A Survey," in 2016 *International Conference on Communication and Signal Processing (ICCSP)*, 2016, pp. 0967–0970, doi: 10.1109/ICCSP.2016.7754291.
- [13] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, Apr. 1980, doi: 10.1007/BF00344251.
- [14] Y. LeCun *et al.*, "Handwritten Digit Recognition with a Back-Propagation Network," in *Advances in Neural Information Processing Systems*, 1989, vol. 2, pp. 396–404.
- [15] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, doi: 10.1109/5.726791.
- [16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.
- [17] J. Gu *et al.*, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, May 2018, doi: 10.1016/j.patcog.2017.10.013.
- [18] V. Nair and G. E. Hinton, "Rectified linear units improve Restricted Boltzmann machines," in *ICML 2010 - Proceedings, 27th International Conference on Machine Learning*, 2010.
- [19] Y. A. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, "Efficient backprop," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2012, doi: 10.1007/978-3-642-35289-8-3.
- [20] C. K. Dewa, A. L. Fadhilah, and A. Afiahayati, "Convolutional Neural Networks for Handwritten Javanese Character Recognition," *IJCCS (Indonesian Journal of Computing and Cybernetics Systems)*, vol. 12, no. 1, p. 83, Jan. 2018, doi: 10.22146/ijccs.31144..
- [21] G. E. Dahl, T. N. Sainath, and G. E. Hinton, "Improving deep neural networks for LVCSR using rectified linear units and dropout," in 2013 *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2013, pp. 8609–8613, doi: 10.1109/ICASSP.2013.6639346.
- [22] S. Nitish, H. Geoffrey, K. Alex, S. Ilya, and S. Ruslan, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [23] Y. LeCun, C. Cortes, and C. J. C. Burges, "MNIST handwritten digit database," *AT&T Labs [Online]*. Available <http://yann.lecun.com/exdb/mnist>, vol. 2, 2010.