Klasifikasi Citra Game Batu Kertas Gunting Menggunakan *Convolutional Neural Network*

Image Classification of Rock Paper Scissor Game Using Convolutional Neural Network

Mohammad Farid Naufal[1], Solichul Huda[2], Aryo Budilaksono[3], Wisnu Aria Yustisia[4], Astri Agustina Arius[5], Fania Alya Miranti[6], Farrel Arghya Tito Prayoga[7]
[1,3,4,5,6,7]Program Studi Teknik Informatika, Universitas Surabaya
[2]Program Studi Teknik Informatika, Universitas Dian Nuswantoro

E-mail: [1]faridnaufal@staff.ubaya.ac.id,
[2]solichul.huda@dsn.dinus.ac.id,[3]s160417131@student.ubaya.ac.id,
[4]s160417120@student.ubaya.ac.id,[5]s160418123@student.ubaya.ac.id,
[6]s160418127@student.ubaya.ac.id,[7]s160418143@student.ubaya.ac.id

Abstrak

Permainan batu, gunting, dan kertas sangat populer di seluruh dunia. Permainan ini biasanya dimainkan saat sedang berkumpul untuk mengundi ataupun hanya bermain untuk mengetahui yang menang dan yang kalah. Namun, perkembangan zaman dan teknologi mengakibatkan orang dapat berkumpul secara *virtual*. Untuk bisa melakukan permainan ini secara *virtual*, penelitian ini membuat model klasifikasi citra untuk membedakan objek tangan yang menunjuk batu, kertas, dan gunting. Performa metode klasifikasi merupakan hal yang harus diperhatikan dalam kasus ini. Salah satu metode klasifikasi citra yang populer adalah *Convolutional Neural Network* (CNN). CNN adalah salah satu jenis neural network yang biasa digunakan pada data klasifikasi citra. CNN terinspirasi dari jaringan syaraf manusia. Algoritma ini memiliki 3 tahapan yang dipakai, yaitu *convolutional layer, pooling layer,* dan *fully connected layer*. Uji coba 5-*Fold cross validation* klasifikasi objek tangan yang menunjuk citra batu, kertas, dan gunting menggunakan CNN pada penelitian ini menghasilkan rata-rata akurasi sebesar 97.66%.

Kata kunci: Permainan batu, gunting, dan kertas, *Convolutional neural network (CNN).* Klasifikasi Citra

Abstract

Rock, scissors and paper games are very popular all over the world. This game is usually played when gathering to draw or just playing to find out who won and who lost. However, the times and technology have resulted in people being able to gather virtually. To be able to play this game virtually, this study creates an image classification model to distinguish hand objects pointing at rocks, paper, and scissors. The performance of the classification method should be considered in this case. One of the popular image classification methods is Convolutional Neural Network (CNN). CNN is a type of neural network that is commonly used in image classification data. CNN is inspired by human neural networks. This algorithm has 3 stages, namely the convolutional layer, the pooling layer, and the fully connected layer. The 5-Fold cross validation trial of the classification of hand objects pointing to images of rocks, paper and scissors using CNN in this study resulted in an average accuracy of 97.66%.

Keywords: The rock, paper, and scissor games, Convolutional neural network (CNN), Image Classification

1. PENDAHULUAN

Bermain merupakan kegiatan yang secara tidak langsung mengasah pikiran dan kreativitas kita, di mana kita sebagai pemain akan memiliki perasaan untuk memenangkan game yang kita mainkan. Salah satu permainan tradisional yang memiliki jalan permainan yang sederhana adalah permainan batu, kertas, dan gunting. Permainan ini merupakan salah satu permainan yang paling terkenal di seluruh dunia karena hampir seluruh orang pernah memainkannya. Permainan batu gunting kertas ini tidak memerlukan alat apapun karena hanya perlu menggunakan tangan kita sebagai alat untuk bermain.

Pengembangan game untuk mendeteksi objek gambar batu, kertas, dan gunting perlu adanya tahap klasifikasi citra. Terdapat banyak metode klasifikasi citra yang populer untuk klasifikasi citra, diantaranya, *K Nearest Neighbors* (KNN), *Support Vector Machine* (SVM), dan *Deep learning* (DL). DL mengungguli KNN dan SVM jika dataset yang digunakan memiliki skala besar dan fitur yang kompleks [1]. Selain itu, DL dapat meraih akurasi yang lebih bagus pada kasus klasifikasi citra, *semantic segmentation, object detection,* dan *Simultaneous Localization and Mapping* (SLAM). Hal ini dikarenakan jaringan *neural* yang digunakan di DL lebih terlatih daripada diprogram, aplikasi yang menggunakan pendekatan ini sering kali memerlukan analisis dan penyempurnaan yang lebih sedikit [2].

Convolutional Neural Network (CNN) adalah salah satu algoritma DL yang populer untuk klasifikasi citra. CNN sama seperti Jaringan Syaraf Tiruan tradisional karena terdiri dari syaraf yang dapat mengoptimalkan diri sendiri melalui setiap pembelajaran yang diberikan. Setiap syaraf akan menerima sebuah input, melakukan operasi dan setiap jaringan yang menghubungkan setiap syaraf mengandung *weight* untuk dioperasikan pada syaraf tertentu [3]. CNN memiliki beberapa tahapan yaitu *Convolutional Layers, Pooling Layers, Fully Connected Layers*. CNN didesain untuk menghandel input dengan bentuk 2 dimensi. Setiap *layer* di jaringannya berkomposisikan multi 2-dimensional planes, dan setiap *planes* terdiri dari *multi neuron* yang *independent*. *Neuron* yang terdapat di 2 *layer* berdekatan akan saling terhubung [4]. CNN telah menjuarai banyak kompetisi dalam bidang pengenalan citra [5].

Citra gestur tangan manusia untuk *game rock paper scissors* memiliki karakteristik yang unik, yaitu tangan manusia memiliki bentuk yang berbeda saat merepresentasikan *rock, paper,* dan *scissors*. Salah satu fitur yang dapat diekstrak adalah *edge* pada tangan. CNN memiliki tahapan *convolution* yang secara otomatis akan melakukan ekstraksi pada citra [6]. Sehingga CNN akan cocok digunakan untuk klasifikasi *gesture* tangan manusia untuk *game rock, paper,* dan *scissors*. Penelitian oleh Thema et al. [7] melakukan klasifikasi postur tangan *Rock Paper Scissors* menggunakan CNN dengan arsitektur VGG dan menghasilkan performa klasifikasi 81.53%. Fitur yang digunakan adalah hanya intensitas *pixel* bukan RGB. *Accuracy training* juga semakin berkurang dengan semakin banyaknya *epoch* yang digunakan. Hal ini menunjukkan bahwa model yang dibuat mengalami *overfitting*. Penelitian oleh Kim et al [8] melakukan klasifikasi menggunakan CNN dan melakukan *binarization* untuk membedakan area tangan. Namun data *training* yang digunakan hanya sebanyak 100 dan *accuracy* yang dihasilkan adalah 90%. Penelitian ini juga tidak menerapkan metode data *augmentation*, sehingga data *training* tidak dapat diperkaya dengan berbagai macam variasi. Semakin kaya data *training*, maka *accuracy* dapat meningkat. Dapat disimpulkan bahwa *gap* penelitian adalah belum adanya tahapan data *augmentation* pada CNN untuk melakukan klasifikasi citra *gestur* tangan yang merepresentasikan *game rock, paper, dan scissors.*

Penelitian ini menggunakan CNN untuk klasifikasi *gesture* tangan *rock, paper, dan scissors* dengan menambahkan tahapan data *augmentation*. Data *augmentation* berguna untuk menghindari terjadinya *overfitting* dan memperkaya data training sehingga dapat meningkatkan *accuracy* klasifikasi. Tahapan pada penelitian ini adalah pengumpulan dataset, pembentukan model arsitektur CNN, melakukan uji coba menggunakan 5-*fold cross validation,* dan melakukan penghitungan performa. Dari uji coba yang dilakukan penghitungan performa menggunakan rata-rata dari tiap *cross validation.*

2. METODE PENELITIAN

Metode penelitian yang dilakukan dalam penelitian ini terdiri dari empat langkah, yaitu: pengumpulan dataset, pembentukan model CNN, *training* dan *testing*, dan perhitungan performa. Gambar 1 menunjukkan tahapan metode penelitian.

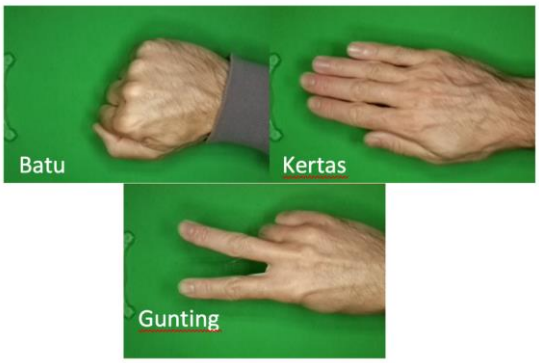
Gambar 1 Metode Penelitian

2.1 Pengumpulan Dataset

Dataset yang digunakan dalam penelitian berupa sampel gambar *Rock*, *Paper* dan *Scissor* yang didapat dari Kaggle [9]. Dataset terbagi menjadi 3 folder dengan total masing - masing adalah 726 gambar batu, 712 gambar kertas, dan 750 gambar gunting. Semua dataset tersimpan dalam format .png. Tabel 1 menunjukkan detail dataset dari tiap *class*. Pembagian *dataset* untuk proses *training* dan *testing* akan dibahas di subbab 2.3. Gambar 2 menunjukkan contoh data gambar untuk *Rock*, *Paper* dan *Scissors*

Tabel 1 Detail Jumlah *Dataset* dari Tiap *Class*

Class	**Jumlah**
Batu	726
Kertas	712
Gunting	750



Gambar 2 Contoh Dataset Gambar Batu, Gunting, dan Kertas

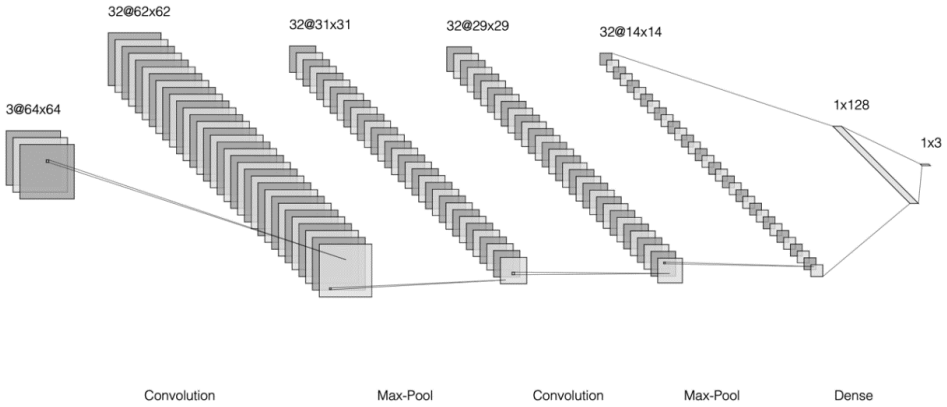
2.2 Pembentukan Model CNN

Pada model CNN dari *library* keras mirip dengan MLP namun terdapat 2 lapis *layer* konvolusi dan *max pooling*. Fungsi dari *layer* konvolusi adalah untuk mengekstraksi atribut pada gambar. Sedangkan *layer max pooling* berguna untuk mereduksi resolusi gambar. Pooling digunakan bertujuan untuk mengurangi dimensi *(downsampling)* sehingga mempercepat komputasi [10].

Tabel 2 Model Arsitektur CNN

Layer (type)	**Output Shape**	**Deskripsi**
conv2d (Conv2D)	(None, 62, 62, 32)	Filter_size = 3x3. Act = ReLu
max_pooling2d (MaxPooling2D)	(None, 31, 31, 32)	Pool_size = 2
conv2d_1 (Conv2D)	(None, 29, 29, 32)	Filter_size = 3x3. Act = ReLu

max_pooling2d_1 (MaxPooling2D)	(None, 14, 14, 32)	Pool_size = 2
flatten (Flatten)	(None, 6272)	-
dense (Dense)	(None, 128)	Act = ReLu
dense_1 (Dense)	(None, 3)	Act= Softmax



Gambar 2 Arsitektur CNN

Dapat dilihat model CNN pada Tabel 2 dan Gambar 3, yaitu terdapat dua tahapan konvolusi dengan masing-masing ukuran filter 3x3 dengan *activation function ReLu* [11], dua tahapan *Max Pooling* dengan ukuran *pool* 2x2, satu *Dense Layer* dengan ukuran 128 menggunakan *activation function ReLu*, satu *Dense Layer output* dengan ukuran 3 dan *activation function Softmax*. Output layer dipilih *Softmax* dengan ukuran 3 dikarenakan terdapat 3 jenis gambar (Batu, Kertas dan Gunting). Data *pooling* yang dihasilkan masih dalam bentuk *multi array* karena itu perlu dilakukan "*flatten*" yaitu dengan menambahkan *layer Flatten* agar dapat mengkonversi data menjadi array 1 dimensi sehingga dapat dihubungkan dengan semua lapisan.

Persamaan (1) menunjukkan activation function dari ReLu, z adalah nilai *input activation function*. Persamaan (2) menunjukkan *activation function Softmax* dengan *input* x_i, jumlah label class n, dan label kelas ke-j.

$$R(z) = (0, z) \quad (1)$$

$$S(x_i) = \frac{e^{x_i}}{\sum_j^n \quad e^{x_j}} \quad (2)$$

2.3 Training dan *Testing*

Pada tahapan *training* dan *testing*, dataset gambar dilakukan *resize* dengan ukuran 64x64. Untuk menghindari *overfitting*, pada tahapan *training* dilakukan 3 proses *data augmentation*, yaitu dengan menambahkan data gambar yang dilakukan *counter clockwise* sebesar 20 derajat, *zoom* gambar sebesar 20%, dan *horizontal flip* secara *random. Overfitting* sendiri terjadi saat semua data yang telah ditrain persentasenya tidak sesuai pada proses *testing*. Tabel 3 menunjukkan tipe proses *data augmentation* yang digunakan. Proses *data augmentation* dilakukan menggunakan *library* Keras [12].

Tabel 3 Tipe Proses Data Augmentation

Data Augmentation	**Nilai**
Counter Clockwise	20 derajat
Zoom	20%
Horizontal Flip	-

Training pada model CNN menggunakan jumlah *epochs* sebanyak 10 dan *batch size* sebanyak 32 dengan menggunakan *callbacks* untuk menyimpan model dengan *metric accuracy* terbaik setiap literasinya. *Optimizer* yang digunakan untuk perbaikan model adalah adam [13] sedangkan loss function adalah *categorical cross entropy* [14]. Persamaan 3 menunjukkan *loss function categorical cross entropy*, di mana t_i dan s_i adalah nilai *ground truth* dan *score output* dari *softmax* untuk tiap *class* i di C. Tabel 4 menunjukkan *detail* parameter yang digunakan untuk *training* model CNN.

$$CE = -\sum_{i}^{C} t_i log(s_i) \quad (3)$$

Testing dilakukan dengan menggunakan metode *K-Fold cross validation* dengan nilai K sebanyak 5. *Dataset* dibagi menjadi dua bagian, yaitu 80% untuk *training* dan 20% untuk *testing.* Tahapan ini dilakukan sebanyak 5 kali secara acak dan diambil nilai rata-rata *accuracy, precision, recall, dan f1 score*. Perhitungan performa dijelaskan lebih detail pada sub bab 2.4.

Tabel 4 Parameter Training Model CNN

Parameter	**Nilai**
Epoch	10
Batch Size	32
Loss Function	Categorical Cross Entropy
Optimizer	Adam
Optimizer Metric	Accuracy

2.4 Perhitungan Performa

Perhitungan performa yang digunakan adalah *metric Accuracy, Precision, Recall dan F1 Score.* Persamaan (5) hingga (8) berturut-turut merupakan perhitungan dari *Accuracy, Precision, Recall, dan F1 Score*. Perhitungan *metric Precision, Recall,* dan *F1 Score* mempertimbangkan permasalahan klasifikasi *multiclass* dengan cara memberikan bobot yang merupakan *instance* dari tiap *class* dan mengalikannya dengan tiap nilai *metric* tiap *class*. Perhitungan ini disebut weighted metric [15]. Hal ini dapat memberikan perhitungan performa dengan lebih baik dikarenakan frekuensi dari tiap class mungkin berbeda. Persamaan 9 menunjukkan perhitungan *weighted metric,* di mana m_i adalah *metric Precision, Recall,* atau *F1 Score* untuk *class* i, j adalah jumlah *class, dan* c_i adalah frekuensi dari kelas i

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

$$F1\ Score = \frac{Precision\ x\ Recall}{Precision + Recall} \quad (8)$$

$$W_m = \frac{\sum_i^j m_i c_i}{\sum_i^j c_i} \quad (9)$$

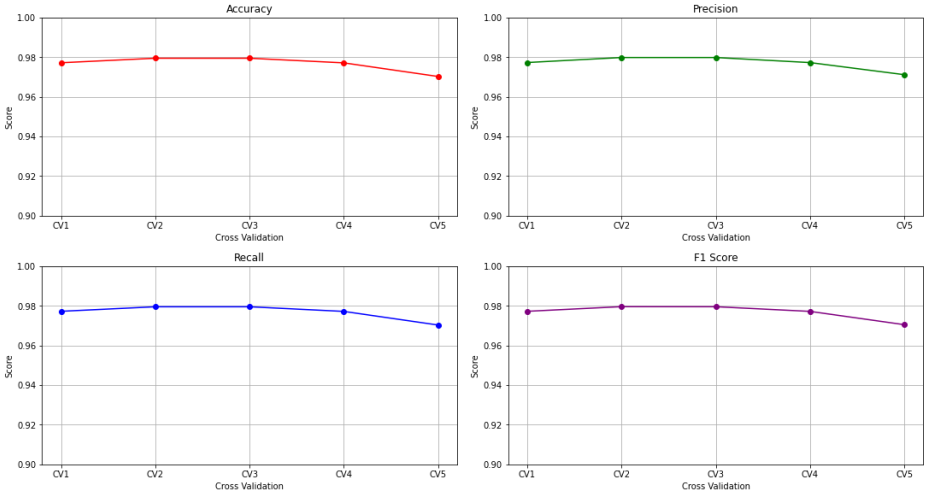
3. HASIL DAN PEMBAHASAN

Tahapan ini menjelaskan hasil penerapan dari metode penelitian. Nilai dari tiap *metric* yaitu *Accuracy, Precision, Recall, dan F1 Score.* Tabel 5 menunjukkan detail dari tiap rata-rata metric di setiap 5-fold cross validation.

Tabel 5 Rata-Rata Nilai Metric di Setiap K-Fold

K-Fold	Accuracy	Precision	Recall	F1 Score
CV1	0.9771	0.9772	0.9771	0.9771
CV2	0.9794	0.9798	0.9794	0.9795
CV3	0.9794	0.9798	0.9794	0.9794
CV4	0.9771	0.9772	0.9771	0.9771
CV5	0.9702	0.9771	0.9702	0.9704
Rata-Rata	**0.9766**	**0.9782**	**0.9766**	**0.976**

Performa yang dihasilkan dari perhitungan rata-rata *metric Accuracy, Precision, Recall*, dan F1 *Score* menunjukkan nilai yang cukup siginifikan yaitu *Accuracy* sebesar 97.66%, *Precision* sebesar 97.82%, *Recall* sebesar 97.66% dan F1 *Score* sebesar 97.6%. hal ini menunjukkan performa model CNN cukup stabil di setiap *K-Fold*. Gambar 3 menunjukkan grafik *metric* dari setiap *K-Fold*.

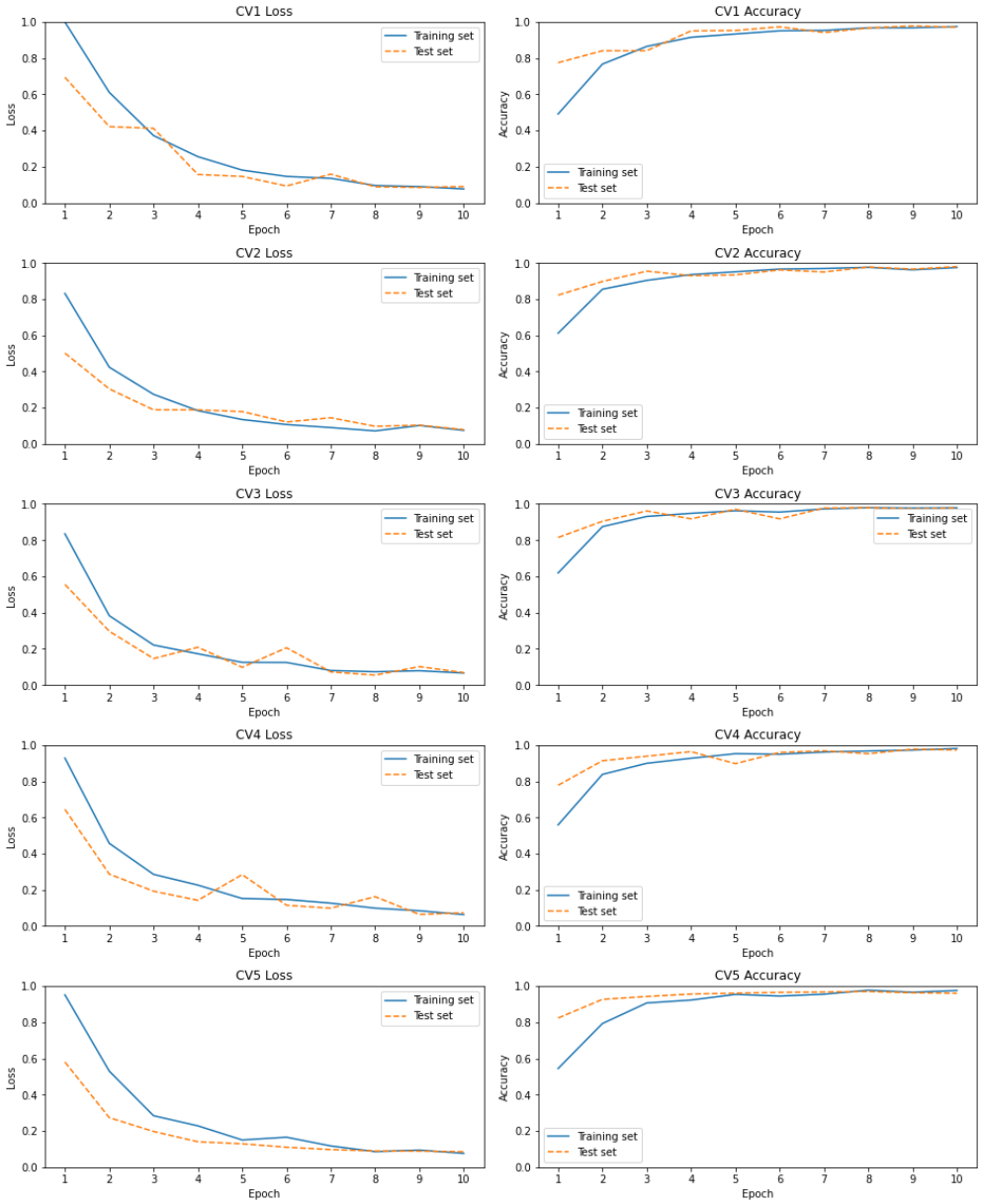


Gambar 3 Grafik Metric di setiap K-Fold

Dalam pembentukan model CNN juga dianalisa *metric Loss* dan *Accuracy* di setiap *epoch*. Model yang terbentuk saat *training* akan diujicoba ke *data testing* di setiap *epoch*. Namun hasil akurasi yang dihasilkan saat proses testing tidak akan dioptimasi untuk memperbarui bobot tiap neuron. Hanya akurasi yang dihasilkan saat *training* yang digunakan untuk optimasi. Hal ini dilakukan untuk melihat apakah model yang dibuat mengalami *overfitting* atau tidak. Jika terdapat perbedaan yang siginifikan antara akurasi saat proses *training* dan *testing*, maka dapat dikatakan terjadi *overfitting* dari model yang dibuat. Tabel 6 menunjukkan perbandingan akurasi saat proses *training* dan *testing* di setiap *epoch*. Gambar 4 menunjukkan grafik *loss* dan *accuracy* dari setiap *epoch*. Dari 5-*Fold* yang diuji coba, perbedaan antara accuracy di tahapan *training* dan *testing* tidak berbeda jauh kecuali saat *epoch* pertama, sehingga dapat dikatakan model CNN yang dibuat tidak *overfitting*.

Tabel 6 Perbandingan Akurasi Saat Proses Training dan Testing di Setiap Epoch

Epoch	CV1		CV2		CV3		CV4		CV5	
	Train Acc	Test Acc	Train Acc	Test Acc	Train Acc	Test Acc	Train Acc	Test Acc	Train Acc	Test Acc
1	0.4909	0.7740	0.6114	0.8219	0.6189	0.8151	0.5591	0.7780	0.5448	0.8238
2	0.7663	0.8402	0.8543	0.8973	0.8749	0.9041	0.8378	0.9130	0.7933	0.9268
3	0.8640	0.8402	0.9034	0.9543	0.9480	0.9612	0.8989	0.9382	0.9069	0.9428
4	0.9143	0.9498	0.9354	0.9292	0.9623	0.9178	0.9269	0.9634	0.9229	0.9565
5	0.9326	0.9521	0.9509	0.9338	0.9549	0.9703	0.9520	0.8970	0.9543	0.9611
6	0.9503	0.9726	0.9657	0.9612	0.9726	0.9178	0.9492	0.9588	0.9452	0.9657
7	0.9526	0.9406	0.9691	0.9498	0.9789	0.9772	0.9612	0.9680	0.9555	0.9680
8	0.9669	0.9658	0.9754	0.9772	0.9771	0.9795	0.9669	0.9519	0.9777	0.9703
9	0.9669	0.9772	0.9617	0.9658	0.9783	0.9726	0.9720	0.9771	0.9663	0.9634
10	0.9737	0.9703	0.9743	0.9795	0.6189	0.9772	0.9812	0.9725	0.9760	0.9611



Gambar 4 Grafik Loss dan Accuracy untuk Setiap Epoch

4. KESIMPULAN DAN SARAN

Berdasarkan hasil uji coba yang telah dilakukan dapat dikatakan bahwa CNN yang dilengkapi tahapan data augmentation memiliki performa yang sangat baik dalam melakukan klasifikasi citra game Batu, Kertas, dan Gunting. Nilai semua *metric performance* menunjukkan nilai di atas 97%. Model yang telah dibentuk pada penelitian ini ke depannya dapat diterapkan untuk game batu, gunting, dan kertas dengan memanfaatkan kamera. Walaupun telah terdapat aplikasi game batu, gunting, dan kertas, namun dengan memanfaatkan kamera, nantinya pemain dapat bermain dengan lebih nyata karena masih menggunakan tangan. Selain itu, dapat juga diterapkan arsitektur CNN lain seperti LeNet-5 [16], AlexNet [17], ZFNet [18], GoogLeNet [19], VGGNet [20], ResNet [21] yang diharapkan adanya peningkatan performa. Untuk meningkatkan kecepatan proses *training* ke depannya juga dapat digunakan metode *transfer learning* [22].

DAFTAR PUSTAKA

[1] L. Zhu and P. Spachos, "Towards Image Classification with Machine Learning Methodologies for Smartphones," *Mach. Learn. Knowl. Extr.*, vol. 1, no. 4, pp. 1039–1057, 2019, doi: 10.3390/make1040059.

[2] N. O'Mahony *et al.*, "Deep Learning vs. Traditional Computer Vision," *Adv. Intell. Syst. Comput.*, vol. 943, no. Cv, pp. 128–144, 2020, doi: 10.1007/978-3-030-17795-9_10.

[3] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," pp. 1–11, 2015, [Online]. Available: http://arxiv.org/abs/1511.08458.

[4] A. A. M. Al-Saffar, H. Tao, and M. A. Talab, "Review of deep convolution neural network in image classification," *Proceeding - 2017 Int. Conf. Radar, Antenna, Microwave, Electron. Telecommun. ICRAMET 2017*, vol. 2018-Janua, pp. 26–31, 2017, doi: 10.1109/ICRAMET.2017.8253139.

[5] M. Pak and S. Kim, "A review of deep learning in image recognition," *Proc. 2017 4th Int. Conf. Comput. Appl. Inf. Process. Technol. CAIPT 2017*, vol. 2018-Janua, pp. 1–3, 2018, doi: 10.1109/CAIPT.2017.8320684.

[6] M. A., Y. A., and M. A., "Automated Edge Detection Using Convolutional Neural Network," *Int. J. Adv. Comput. Sci. Appl.*, vol. 4, no. 10, pp. 10–17, 2013, doi: 10.14569/ijacsa.2013.041003.

[7] Z. Thema and R. Gupta, "Applying Deep Learning for Classifying Images of Hand Postures in the Rock-Paper-Scissors Game," no. 1839122, 2017, [Online]. Available: http://www.cogsys.wiai.uni-bamberg.de/theses/gupta/Masterarbeit_Gupta.pdf.

[8] B. Kim and K. Lee, "Study on Recognition of Hand Gestures Using Convolutional Neural Network," *Int. Conf. Futur. Inf. Commun. Eng.*, vol. 11, no. 1, pp. 236–239, 2019, Accessed: Jan. 16, 2021. [Online]. Available: https://www.dbpia.co.kr/Journal/articleDetail?nodeId=NODE08747453.

[9] "Rock-Paper-Scissors Images | Kaggle." https://www.kaggle.com/drgfreeman/rockpaperscissors (accessed Dec. 08, 2020).

[10] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," *Proc. 2017 Int. Conf. Eng. Technol. ICET 2017*, vol. 2018-Janua, pp. 1–6, 2018, doi: 10.1109/ICEngTechnol.2017.8308186.

[11] A. F. M. Agarap, "Deep Learning using Rectified Linear Units (ReLU)," *arXiv*, no. 1, pp. 2–8, 2018.

[12] "Keras: the Python deep learning API." https://keras.io/ (accessed Dec. 08, 2020).

[13] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–15, 2015.

[14] S. Mannor, B. Peleg, and R. Rubinstein, "The cross entropy method for classification," *ICML 2005 - Proc. 22nd Int. Conf. Mach. Learn.*, pp. 561–568, 2005, doi: 10.1145/1102351.1102422.

[15] C. Goutte and E. Gaussier, "A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation," *Lect. Notes Comput. Sci.*, vol. 3408, no. April, pp. 345–359, 2005, doi: 10.1007/978-3-540-31865-1_25.

[16] Y. Lecun, L. Bottou, Y. Bengio, and P. Ha, "LeNet," *Proc. IEEE*, no. November, pp. 1–46, 1998.

[17] T. F. Gonzalez, "Handbook of approximation algorithms and metaheuristics," *Handb. Approx. Algorithms Metaheuristics*, pp. 1–1432, 2007, doi: 10.1201/9781420010749.

[18] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 8689 LNCS, no. PART 1, pp. 818–833, 2014, doi: 10.1007/978-3-319-10590-1_53.

[19] C. Szegedy *et al.*, "Going deeper with convolutions," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June, pp. 1–9, 2015, doi: 10.1109/CVPR.2015.7298594.

[20] 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.

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[22] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11141 LNCS, pp. 270–279, 2018, doi: 10.1007/978-3-030-01424-7_27.