Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 8, Number 1, January 2024
DOI : https://doi.org/10.33395/sinkron.v9i1.13176

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Analysis and Implementation of CNN in Real-time Classification and Translation of Kanji Characters

Putri Annisa[1)*], **Zuli Agustina Gultom**[2)], **Yoshida Sary**[3)]
[1,2,3)] Muhammadiyah University of North Sumatra, Indonesia
[1)]putriannisa@umsu.ac.id , [2)]zuliagustina@umsu.ac.id, [3)]yoshidasary@umsu.ac.id

Submitted : Nov 14, 2023 | **Accepted** : Nov 30, 2023 | **Published** : Jan 1, 2024

Abstract: The Kanji characters are essential elements in the Japanese language. In the current era of globalization, improved access to technology capable of recognizing Kanji characters can support cross-cultural communication, facilitate trade, education, and the exchange of information worldwide. Based on these reasons, the development of models capable of identifying Kanji characters is currently a crucial focus, considering that Japan is one of the destination countries for cultural exchange, education, trade, and various other fields in Indonesia.This research explores the training outcomes of the Convolutional NeuralNetwork (CNN) algorithm applied to Kanji character recognition. It employs a CNNarchitecture with 10 layers for recognizing digital image of Kanji characters from N5to N1 levels. The training of the CNN model reveals varying accuracies, influencedby factors such as architecture, training data size, and data quality. The lowest accuracy, occurring at the beginning of training, highlights challenges like poor random weight initialization and suboptimal architecture. Conversely, the highest accuracy demonstrates the optimal predictive ability of the model after multiple training iterations. The iterative training process refines the model's accuracy over time, with initial challenges paving the way for a better understanding of the data forfuture improvements. In the subsequent analysis and system development phase, two algorithms are compared to assess the effectiveness of the CNN algorithm in Kanji character recognition. Testing is conducted at various complexity levels to systematically evaluate accuracy. This testing involves complexity levels at each Kanji character level to assess system accuracy. The developed model shows potential for real-time classification of Kanji characters, with a focus on error rate and accuracy during training and testing. The built model can essentially be used toevaluate the ability to recognize Kanji characters effectively. The model's performance can be assessed based on the error rate and accuracy achieved during the training and testing processes. Based on the CNN training that has been conducted, a training accuracy of 0.6944 and a validation accuracy of 0.6250 were obtained. These results indicate a good identification performance; however, there is still a limitation in identifying Kanji characters with high similarity.

Keywords: CNN, Kanji Characters, OCR, Real-time classification, Digital Image

INTRODUCTION

Kanji characters are ideographic symbols with meaning and narratives embedded in each of their components (Ambarastuti & Savitri, 2021). Essentially, Kanji characters use symbols to convey specific

*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 8, Number 1, January 2024
DOI : https://doi.org/10.33395/sinkron.v9i1.13176
e-ISSN : 2541-2019
p-ISSN : 2541-044X

meanings. In the Japanese language, Kanji is considered challenging, both when standing alone and when combined with other Kanji characters, as the resulting meanings can vary.

With technological advancements, the difficulty in identifying and classifying Kanji characters can be automated using computer devices. By utilizing digital image media, computers can identify Kanji characters within those images. Computer vision is a field of study that can fulfill this need.

The identification or classification of objects in digital images using computer vision involves several stages and models. Models used in the process of identifying and classifying objects in digital images can range from simple mathematical equations to artificial neural networks. With the development of artificial intelligence technology, artificial neural networks have been widely applied compared to other classification models. Some research on identifying letters in digital images, such as that conducted by Riyadi et al., exploring the recognition of Latin letters using backpropagation (Riyadi, et al., 2021), has shown reasonably good results, albeit within a controlled environment. Similar findings are observed in research by Setiani, who also employed backpropagation but focused on a different set of letters, specifically the hiragana script (Setiani, 2020).

The classification of Kanji characters using artificial neural networks continues to evolve with the emergence of new methods in the field. For instance, Vo and Truong applied ANFIS to Kanji character classification (Vo & Truong, 2021). Adole et al. applied RCNN Inception ResNet V2 to handwritten Kanji character classification, achieving a commendable accuracy of 85% recognition of available Kanji characters (Adole, et al., 2020). Despite the relatively high accuracy, Adole et al.'s research was conducted in a controlled environment similar to previous studies, limiting its application in real-world or scene image environments.

Convolutional Neural Network (CNN) is a model that combines convolutional operations and artificial neural networks, consisting of several convolutional sub-stages aimed at feature extraction to be used as input for the neural network (Li, et al., 2021). CNN has been widely applied in the classification of objects in digital images. In the case of letter identification, several studies have applied CNN, yielding satisfactory identification results under specified conditions (Bora, et al., 2020). This study will apply CNN to Kanji character identification, following a similar approach to previous research using CNN but with the potential for application in real-world or scene image environments.

LITERATURE REVIEW

The Japanese language recognizes three types of writing: kanji (漢字), kana (仮名), and Latin letters (ローマ字). Kana and Latin letters represent sounds. Kana represents syllables or more, while Latin letters represent phonemes. On the other hand, kanji characters represent words or morphemes and meanings (ideographs). In other words, each kanji character not only represents a sound but also conveys the meaning of the written word. Kanji plays a crucial role and can be considered the "backbone" of Japanese vocabulary. Without kanji, it would be challenging to interpret vocabulary. Conversely, with kanji, we can understand the meaning of a word precisely, reducing the risk of misinterpretation.

Kanji is a system of written characters or Hanzi script used in the Japanese language. These Kanji characters are adapted from Chinese Hanzi characters and comprise a vast number, exceeding 50,000 characters. Each Kanji character holds a distinct meaning and pronunciation, serving the purpose of writing words in Japanese. The significance of Kanji characters in the Japanese language is paramount, as many Japanese words incorporate these characters. Therefore, it is crucial for learners of the Japanese language to have a thorough understanding of Kanji characters. To study Kanji characters effectively, a foundational grasp of the meaning and pronunciation of each character, as well as recognition of its written form, is essential. Learning Kanji characters also involves understanding their usage in words and phrases, as well as employing them appropriately within specific contexts.

Machine Learning is an algorithm that processes data, learns from it and applies what it has learned to make decisions. Deep Learning is a subset of Machine Learning designed to continuously analyze data with a logical structure similar to how humans make decisions. Convolutional Neural Network (CNN) is one of the Deep Learning methods used for detecting and recognizing objects in digital images. CNN's ability is claimed to be the best model for solving Object Detection and Object Recognition problems. In 2012, research onCNN showed the capability of digital image recognition with accuracy rivaling that of humans on specific datasets.

*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X

Initially, Convolutional Neural Network (CNN) was developed by a researcher named Kunihiko Fukushima at NHK Public Japan Broadcasting Science Research Laboratories in Tokyo, Japan, under the name NeoCognitron (B. K. Triwijoyo, 2020). In a study conducted by Shobhit Bhatnagar from the Indian Institute of Technology Patna, a comparison was made between the CNN algorithm and the Support Vector Classifier (SVC) algorithm for fashion product classification. The research revealed that the CNN algorithm had a higher accuracy rate compared to SVC, with respective figures of 91.17% for CNN and 89.70% for SVC.

Another study conducted by Tjokorda Agung Budi used Random Forest as an alternative method in recognizing Hangul's Korean character in his thesis. Random Forest can accept various types of input data and produce good accuracy values. The test results of Random Forest with 10 trees and projection-based feature extraction were able to classify Hangul syllables based on KS5602 very effectively.

METHOD

Convolutional Neural Network (CNN) is a deep learning architecture method designed for processing structured grid data, such as images. It is particularly powerful in tasks related to computer vision, image recognition, and pattern detection. The CNN architecture is characterized by several key components:

Convolutional Layers:
Convolutional layers are the core building blocks of CNNs.
They use convolutional operations to detect patterns, features, or objects in the input data.
These layers have learnable filters or kernels that slide over the input to perform convolutions.

Pooling Layers:
Pooling layers follow convolutional layers.
They reduce the spatial dimensions (width and height) of the input volume.
Common pooling operations include max pooling and average pooling.

Activation Function:
Activation functions introduce non-linearity to the network.
Common activation functions include ReLU (Rectified Linear Unit) to add non-linearity to the model.

Fully Connected Layers (Dense Layers):
Fully connected layers connect every neuron in one layer to every neuron in the next layer.
They are typically found towards the end of the network and are responsible for making predictions based on the learned features.

Flattening:
Before the fully connected layers, the 2D feature maps are flattened into a 1D vector.
This prepares the data for input into the dense layers.

Dropout:
Dropout is a regularization technique used to prevent overfitting.
During training, random neurons are "dropped out" or ignored, forcing the network to learn more robust features.

*name of corresponding author

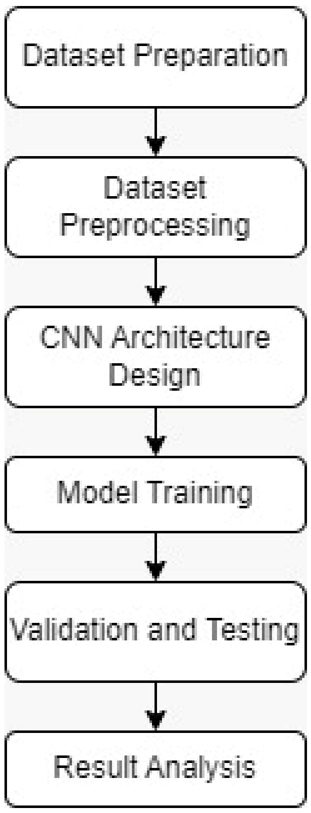


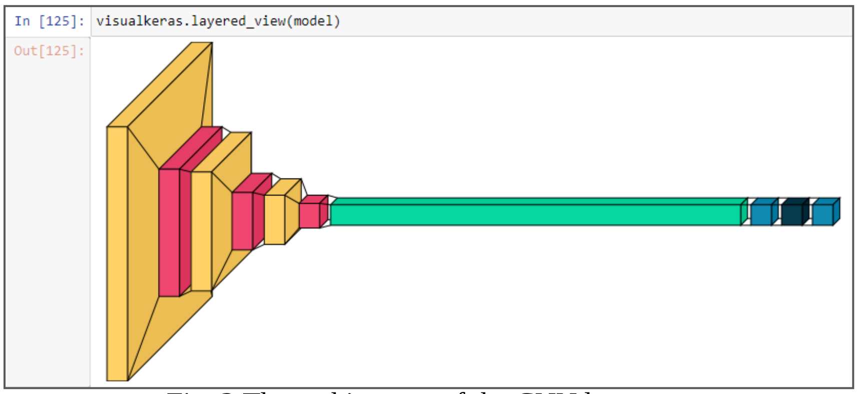
Fig. 1 CNN Step in Kanji Recognition

1. Dataset Selection: Choose a dataset that includes Kanji characters from levels N5 to N1. Ensure the dataset covers a variety of characters, has adequate image resolution, and clear labels for each character.
2. Data Preprocessing: Preprocess the dataset by normalizing images, resizing them to a consistent size, converting them to a format suitable for the CNN model (e.g., array format), and splitting the dataset into training, validation, and testing sets.
3. CNN Architecture Design: Select or design a CNN architecture suitable for Kanji character recognition. This involves determining the number of layers, types of layers (convolutional, pooling, fully connected), activation functions, and other parameters.
4. Model Training: Use the training data to train the CNN model. Conduct the training process iteratively to adjust weights and optimize the model. Evaluate the model using validation data to monitor performance during training.
5. Validation and Testing : Use validation data to fine-tune the model's hyperparameters, such as learning rate, number of layers, or batch size, aiming to improve the model's performance and prevent overfitting. After training, test the model using separate testing data. Evaluate the model's performance based on relevant metrics such as accuracy.
6. Results Analysis: Analyze the testing results to understand the model's strengths and weaknesses. Identify characters that are difficult to recognize and factors influencing their recognition.

The typical CNN architecture follows a pattern of alternating convolutional and pooling layers, followed by fully connected layers. This hierarchical structure enables the network to learn increasingly complex features in the input data. CNNs have been highly successful in various image-related tasks, including image classification, object detection, and image segmentation. Their ability to automatically learn hierarchical features makes them well-suited for tasks where understanding spatial hierarchies is crucial. The CNN architecture used in this researchconsists of 10 layers, comprising 3 convolutional layers, 3 sample-based discretization process layers, 1 matrix transformation layer, 1 dropout layer, and 2 neural network layers.

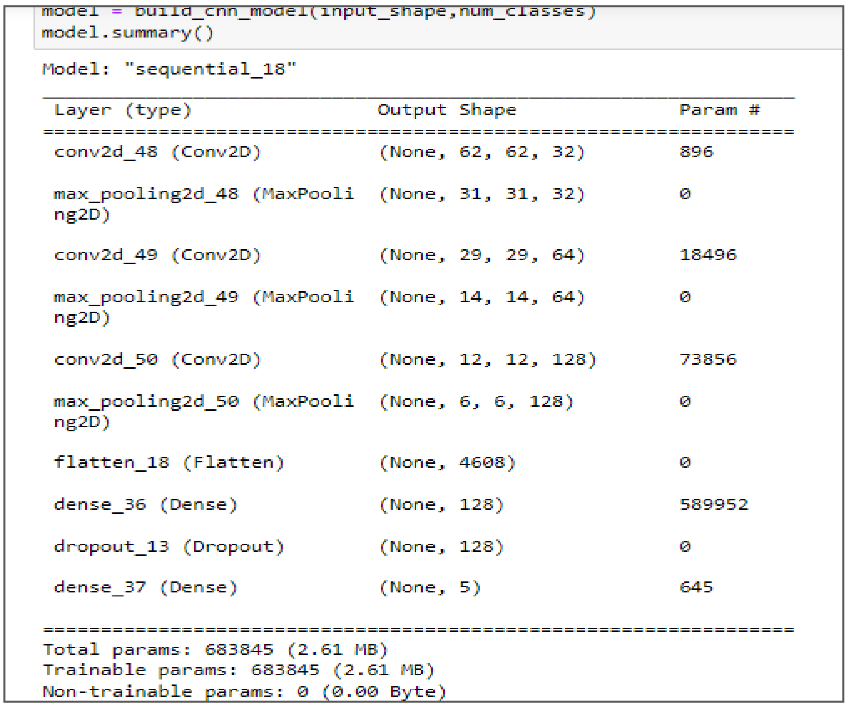
*name of corresponding author

e-ISSN : 2541-2019
p-ISSN : 2541-044X



```
In [125]: visualkeras.layered_view(model)
Out[125]:
```

Fig. 2 The architecture of the CNN layers



```
model = build_cnn_model(input_shape,num_classes)
model.summary()

Model: "sequential_18"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_48 (Conv2D)          (None, 62, 62, 32)        896

 max_pooling2d_48 (MaxPooli  (None, 31, 31, 32)        0
 ng2D)

 conv2d_49 (Conv2D)          (None, 29, 29, 64)        18496

 max_pooling2d_49 (MaxPooli  (None, 14, 14, 64)        0
 ng2D)

 conv2d_50 (Conv2D)          (None, 12, 12, 128)       73856

 max_pooling2d_50 (MaxPooli  (None, 6, 6, 128)         0
 ng2D)

 flatten_18 (Flatten)        (None, 4608)              0

 dense_36 (Dense)            (None, 128)               589952

 dropout_13 (Dropout)        (None, 128)               0

 dense_37 (Dense)            (None, 5)                 645

=================================================================
Total params: 683845 (2.61 MB)
Trainable params: 683845 (2.61 MB)
Non-trainable params: 0 (0.00 Byte)
```

Fig. 3 CNN Models

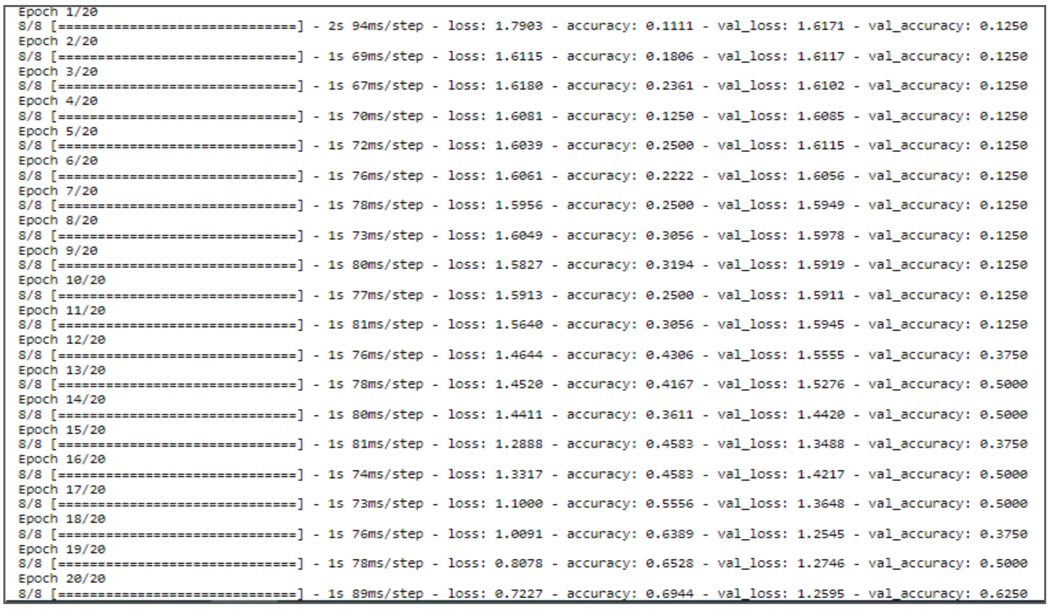
Training Results:
The lowest and highest accuracies obtained during the training of a Convolutional Neural Network (CNN) algorithm on a dataset can vary significantly depending on various factors, including the CNN architecture used, the size of the training data, data quality, and many other factors. However, in a general context, here is a brief explanation of the concepts of lowest and highest accuracy in the training of a CNN model:

Lowest Accuracy:
a. Lowest accuracy is the lowest level of accuracy achieved by the CNN model during the training process.
b. This typically occurs at the beginning of training when the CNN model has not learned well, and itspredictions are highly inaccurate.
c. The lowest accuracy can be caused by various issues such as poor random weight initialization,inappropriate training data, or an ill-suited architecture.

*name of corresponding author

Highest Accuracy:

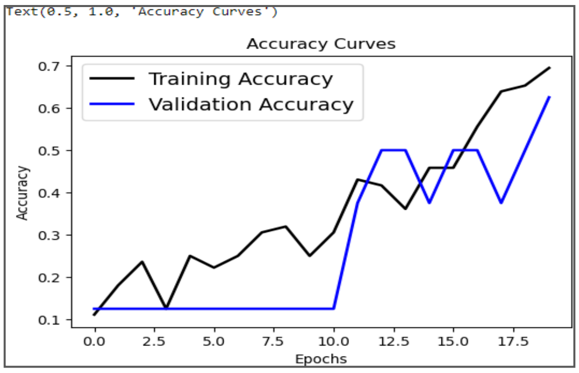
a. Highest accuracy is the highest level of accuracy achieved by the CNN model during the training process.
b. It reflects the best accuracy level that the CNN model can achieve on a specific training dataset after multiple training iterations.
c. The highest accuracy reflects the CNN model's ability to understand patterns in the data and make accurate predictions.



```
Epoch 1/20
8/8 [==============================] - 2s 94ms/step - loss: 1.7903 - accuracy: 0.1111 - val_loss: 1.6171 - val_accuracy: 0.1250
Epoch 2/20
8/8 [==============================] - 1s 69ms/step - loss: 1.6115 - accuracy: 0.1806 - val_loss: 1.6117 - val_accuracy: 0.1250
Epoch 3/20
8/8 [==============================] - 1s 67ms/step - loss: 1.6180 - accuracy: 0.2361 - val_loss: 1.6102 - val_accuracy: 0.1250
Epoch 4/20
8/8 [==============================] - 1s 70ms/step - loss: 1.6081 - accuracy: 0.1250 - val_loss: 1.6085 - val_accuracy: 0.1250
Epoch 5/20
8/8 [==============================] - 1s 72ms/step - loss: 1.6039 - accuracy: 0.2500 - val_loss: 1.6115 - val_accuracy: 0.1250
Epoch 6/20
8/8 [==============================] - 1s 76ms/step - loss: 1.6061 - accuracy: 0.2222 - val_loss: 1.6056 - val_accuracy: 0.1250
Epoch 7/20
8/8 [==============================] - 1s 78ms/step - loss: 1.5956 - accuracy: 0.2500 - val_loss: 1.5949 - val_accuracy: 0.1250
Epoch 8/20
8/8 [==============================] - 1s 73ms/step - loss: 1.6049 - accuracy: 0.3056 - val_loss: 1.5978 - val_accuracy: 0.1250
Epoch 9/20
8/8 [==============================] - 1s 80ms/step - loss: 1.5827 - accuracy: 0.3194 - val_loss: 1.5919 - val_accuracy: 0.1250
Epoch 10/20
8/8 [==============================] - 1s 77ms/step - loss: 1.5913 - accuracy: 0.2500 - val_loss: 1.5911 - val_accuracy: 0.1250
Epoch 11/20
8/8 [==============================] - 1s 81ms/step - loss: 1.5640 - accuracy: 0.3056 - val_loss: 1.5945 - val_accuracy: 0.1250
Epoch 12/20
8/8 [==============================] - 1s 76ms/step - loss: 1.4644 - accuracy: 0.4306 - val_loss: 1.5555 - val_accuracy: 0.3750
Epoch 13/20
8/8 [==============================] - 1s 78ms/step - loss: 1.4520 - accuracy: 0.4167 - val_loss: 1.5276 - val_accuracy: 0.5000
Epoch 14/20
8/8 [==============================] - 1s 80ms/step - loss: 1.4411 - accuracy: 0.3611 - val_loss: 1.4420 - val_accuracy: 0.5000
Epoch 15/20
8/8 [==============================] - 1s 81ms/step - loss: 1.2888 - accuracy: 0.4583 - val_loss: 1.3488 - val_accuracy: 0.3750
Epoch 16/20
8/8 [==============================] - 1s 74ms/step - loss: 1.3317 - accuracy: 0.4583 - val_loss: 1.4217 - val_accuracy: 0.5000
Epoch 17/20
8/8 [==============================] - 1s 73ms/step - loss: 1.1000 - accuracy: 0.5556 - val_loss: 1.3648 - val_accuracy: 0.5000
Epoch 18/20
8/8 [==============================] - 1s 76ms/step - loss: 1.0091 - accuracy: 0.6389 - val_loss: 1.2545 - val_accuracy: 0.3750
Epoch 19/20
8/8 [==============================] - 1s 78ms/step - loss: 0.8078 - accuracy: 0.6528 - val_loss: 1.2746 - val_accuracy: 0.5000
Epoch 20/20
8/8 [==============================] - 1s 89ms/step - loss: 0.7227 - accuracy: 0.6944 - val_loss: 1.2595 - val_accuracy: 0.6250
```

Fig. 4 The training process of the CNN model

The training process of a CNN model typically involves many iterations in which the model's weights and parameters are updated repeatedly. The model's accuracy usually improves over time during training, and the lowest accuracy typically occurs at the beginning of training before the model begins to better "understand" the data.

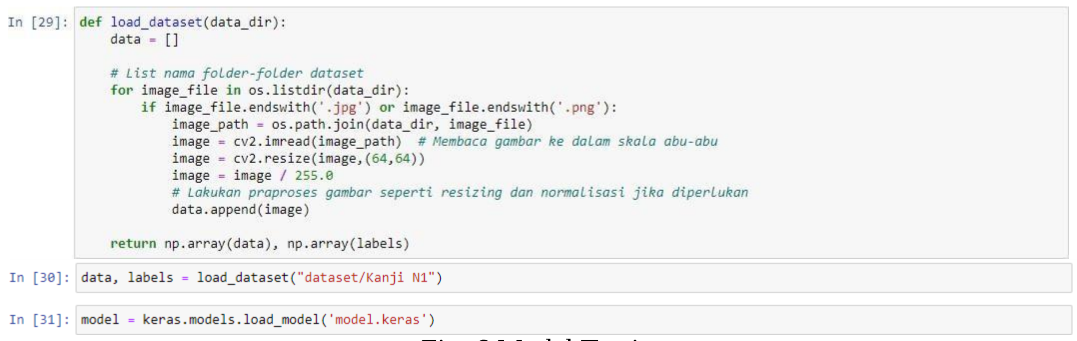


```
Text(0.5, 1.0, 'Accuracy Curves')
```

Fig. 5 Accuracy Curves of the CNN model

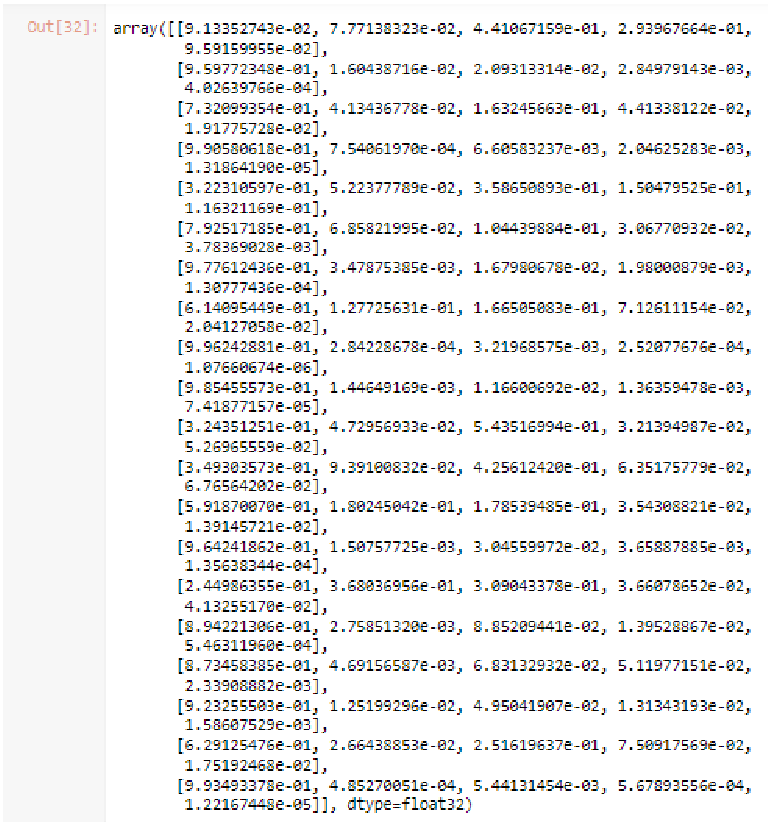
*name of corresponding author

Sinkron : Jurnal dan Penelitian Teknik Informatika
Volume 8, Number 1, January 2024 e-ISSN : 2541-2019
DOI : https://doi.org/10.33395/sinkron.v9i1.13176 p-ISSN : 2541-044X



```
In [29]: def load_dataset(data_dir):
             data = []

             # List nama folder-folder dataset
             for image_file in os.listdir(data_dir):
                 if image_file.endswith('.jpg') or image_file.endswith('.png'):
                     image_path = os.path.join(data_dir, image_file)
                     image = cv2.imread(image_path)  # Membaca gambar ke dalam skala abu-abu
                     image = cv2.resize(image,(64,64))
                     image = image / 255.0
                     # Lakukan praproses gambar seperti resizing dan normalisasi jika diperlukan
                     data.append(image)

             return np.array(data), np.array(labels)

In [30]: data, labels = load_dataset("dataset/Kanji N1")

In [31]: model = keras.models.load_model('model.keras')
```

Fig. 6 Model Testing



```
Out[32]: array([[9.13352743e-02, 7.77138323e-02, 4.41067159e-01, 2.93967664e-01,
                  9.59159955e-02],
                 [9.59772348e-01, 1.60438716e-02, 2.09313314e-02, 2.84979143e-03,
                  4.02639766e-04],
                 [7.32099354e-01, 4.13436778e-02, 1.63245663e-01, 4.41338122e-02,
                  1.91775728e-02],
                 [9.90580618e-01, 7.54061970e-04, 6.60583237e-03, 2.04625283e-03,
                  1.31864190e-05],
                 [3.22310597e-01, 5.22377789e-02, 3.58650893e-01, 1.50479525e-01,
                  1.16321169e-01],
                 [7.92517185e-01, 6.85821995e-02, 1.04439884e-01, 3.06770932e-02,
                  3.78369028e-03],
                 [9.77612436e-01, 3.47875385e-03, 1.67980678e-02, 1.98000879e-03,
                  1.30777436e-04],
                 [6.14095449e-01, 1.27725631e-01, 1.66505083e-01, 7.12611154e-02,
                  2.04127058e-02],
                 [9.96242881e-01, 2.84228678e-04, 3.21968575e-03, 2.52077676e-04,
                  1.07660674e-06],
                 [9.85455573e-01, 1.44649169e-03, 1.16600692e-02, 1.36359478e-03,
                  7.41877157e-05],
                 [3.24351251e-01, 4.72956933e-02, 5.43516994e-01, 3.21394987e-02,
                  5.26965559e-02],
                 [3.49303573e-01, 9.39100832e-02, 4.25612420e-01, 6.35175779e-02,
                  6.76564202e-02],
                 [5.91870070e-01, 1.80245042e-01, 1.78539485e-01, 3.54308821e-02,
                  1.39145721e-02],
                 [9.64241862e-01, 1.50757725e-03, 3.04559972e-02, 3.65887885e-03,
                  1.35638344e-04],
                 [2.44986355e-01, 3.68036956e-01, 3.09043378e-01, 3.66078652e-02,
                  4.13255170e-02],
                 [8.94221306e-01, 2.75851320e-03, 8.85209441e-02, 1.39528867e-02,
                  5.46311960e-04],
                 [8.73458385e-01, 4.69156587e-03, 6.83132932e-02, 5.11977151e-02,
                  2.33908882e-03],
                 [9.23255503e-01, 1.25199296e-02, 4.95041907e-02, 1.31343193e-02,
                  1.58607529e-03],
                 [6.29125476e-01, 2.66438853e-02, 2.51619637e-01, 7.50917569e-02,
                  1.75192468e-02],
                 [9.93493378e-01, 4.85270051e-04, 5.44131454e-03, 5.67893556e-04,
                  1.22167448e-05]], dtype=float32)
```

Fig. 7 Results of Model Testing

Analysis and System :

After the analysis and design phase, system development is carried out as an implementation to assist in testingthe comparison of the two algorithms in determining the capability of the CNN algorithm in Kanji character recognition. Testing of the system is conducted to assess the system's accuracy and to test the dataset created earlier. Testing is conducted at each level of complexity of Kanji character writing and recognition, ranging fromN1 to N5.

Based on the test results in Table 1, it is stated that 3 out of 5 Kanji characters can be identified well, while theother 2 characters are not suitable. This proves that the achieved results for detecting highly complex characters, especially at the N1 level, are good.

Table 1 Testing Kanji Class N1

No.	Image	Classification Result	Classification Class	Status
1	僕	[9.13352743e-02, 7.77138323e-02, 4.41067159e-01,2.93967664e-01, 9.59159955 e-02]	N-3	Not Suitable
2	杉	[9.59772348e-01, 1.60438716e-02, 2.09313314e-02, 2.84979143e -03, 4.02639766e-04]	N-1	Suitable
3	氏	[7.32099354e-01, 4.13436778e-02, 1.63245663e-01, 4.41338122e -02, 1.91775728e-02]	N-1	Suitable
4	幹	[9.90580618e-01, 7.54061970e-04, 6.60583237e-03, 2.04625283e -03, 1.31864190e-05]	N-1	Suitable
5	絹	[3.22310597e-01, 5.22377789e-02, 3.58650893e-01, 1.50479525e -01, 1.16321169e-01]	N-3	Not Suitable

RESULT

The model built can essentially be used to assess the ability to recognize Kanji characters effectively. The model's performance can be evaluated based on the error rate and accuracy achieved during the training and testingprocesses. In the next stages, it is expected that this model can be implemented in an application capable of real- time Kanji character classification.

1. Model Training Process:
   a. The model training process involves using training data (examples of Kanji characters) to teach the modelto recognize these characters.
   b. During training, the model undergoes repeated iterations where its weights and internal parameters areupdated with the goal of reducing prediction errors and improving accuracy.
   c. The accuracy level of the training data can provide initial insights into how well the model can recognizethe Kanji characters used in the training process.
2. Model Testing Process:
   a. After training, the model is tested on a separate dataset that was not used during training (testing data). Thisis done to evaluate how well the model can generalize and classify Kanji characters it has never seen before.
   b. The accuracy rate and error rate on the testing data provide a better understanding of the model's performancein real-world situations beyond the training data.
3. Implementation in a Real-time Application:
   a. Once the model is well-trained and achieves adequate accuracy, it can be integrated into an application thatallows users to classify Kanji characters in real-time.
   b. This application can accept input in the form of Kanji character images and use the model to quickly andaccurately identify these characters.
   c. The classification results provided by the model can be displayed to users in real-time, enabling users todetect and identify Kanji characters with the assistance of technology.

*name of corresponding author

The accuracy levels obtained from the training results can be observed, where in the first iteration, an accuracy of 0.1111 was achieved, and it continued to increase in each epoch, resulting in an accuracy of 0.6944. Based on the conducted testing, an accuracy of 72% was obtained, with several Kanji characters being well-recognized.

In the research on Kanji character recognition using a Convolutional Neural Network (CNN) model, several challenges may be encountered including the availability of adequate data, the need for significant computationalresources, complex image processing techniques, and thorough evaluation and validation. Additionally, interpreting research results is also a crucial aspect to consider. All of these challenges must be handled carefully to ensure the success of the research and effective implementation in real-world applications.

As for the Kanji character data itself, it is obtained by directly writing each character at each Kanji level in digital form. Meanwhile, the computational resources available to the researcher are limited to a laptop with sufficient computing power, which results in a relatively lengthy training process for the dataset.

DISCUSSIONS

In the research on Kanji character recognition using a Convolutional Neural Network (CNN) model, several challenges may be encountered including the availability of adequate data, the need for significant computational resources, complex image processing techniques, and thorough evaluation and validation. Additionally, interpreting research results is also a crucial aspect to consider. All of these challenges must be handled carefully to ensure the success of the research and effective implementation in real-world applications.

As for the Kanji character data itself, it is obtained by directly writing each character at each Kanji level in digital form. Meanwhile, the computational resources available to the researcher are limited to a laptop with sufficient computing power, which results in a relatively lengthy training process for the dataset.

CONCLUSION

The study's outcomes reveal the model's effectiveness in recognizing Kanji characters, with a focus on error rates and accuracy across various testing scenarios. Future implementation in real-time applications is envisioned,building upon the well-trained model. The training process, characterized by iterative weight adjustments, resultsin enhanced accuracy over time, overcoming initial challenges. Acknowledging challenges such as limited data and computational resources, the research establishes a foundation for advanced Kanji character recognition technology. The model's potential for practical applications invites further exploration and refinement. Discussionsemphasize the importance of careful handling of challenges for successful real-world implementation, positioningthe research as a valuable contribution to the evolving landscape of Kanji character recognition.

In the context of the Convolutional Neural Network (CNN) training, the obtained results provide insights into the model's performance. The training accuracy of 0.6944 indicates how well the model learned from the training dataset. This value suggests that, during the training process, the model correctly predicted the outcomes for nearly 69.44% of the data it was exposed to.

The validation accuracy, which is another critical metric, is recorded at 0.6250. This metric is essential for evaluating how well the model generalizes to new, unseen data. A validation accuracy of 0.6250 indicates that the model maintains reasonably good predictive performance when applied to data it hasn't encountered during the training phase.

The testing accuracy of 72% further reinforces the model's reliability and effectiveness. This metric is derived from evaluating the model's performance on an independent dataset not used in either training or validation. The 72% accuracy on the test set suggests that the model can make accurate predictions on new and unseen data, demonstrating its ability to generalize well beyond the training dataset.

In summary, the overall performance of the CNN model, with a training accuracy of 0.6944, validation accuracy of 0.6250, and testing accuracy of 72%, is deemed quite satisfactory. These metrics collectively suggest that the model has learned meaningful patterns from the training data, can generalize well to new data, and performs with a commendable level of accuracy in identifying the target features or classes.

ACKNOWLEDGMENT

The author would like to thank the Research Institute and Publication Institute of the Muhammadiyah University of North Sumatera, Medan, Indonesia for supporting the dissemination of novice lecturer research (PDP) with number: 11/II.3-AU/UMSU-LP2M/C/2023.

REFERENCES

Afif, M., Fawwaz, A., Ramadhani, K. N., & Sthevanie, F. (2020). Breed Classification in Cats using Convolutional Neural Network (CNN). Journal of Final Project Faculty of Informatics, 8 (1), 715–730.

Andrian, R., Naufal, M. A., Hermanto, B., Junaidi, A., & Lumbanraja, F. R. (2019). K-Nearest Neighbor (K-NN)Classification for Recognition of the Batik Lampung Motifs. Journal of Physics: Conference Series, 1338 (1),012061. doi:10.1088/1742-6596/1338/1/012061.

Hanin, M. A., Patmasari, R., & Nur, R. Y. (2021). Skin Disease Classification System Using Convolutional NeuralNetwork (CNN). E-Proceeding Engineering, 8 (1), 273–281.

Jamshed Memon, Maira Sami, Rizwan Ahmed Khan, and Mueen Uddin. (2020). Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR). Digital Object Identifier 10.1109/ACCESS.2020.3012542

Setiawan, W. (2020). Deep Learning using Convolutional Neural Network. Media Nusa Creative.

Tan Chiang Wei, U. U. Sheikh, and Ab Al-Hadi Ab Rahman. (2018). Improved Optical Character Recognitionwith Deep Neural Network. International Colloquium on Signal Processing & its Applications.

Shrawan Ram, Shloak Gupta & Basant Agarwal. (2018). Devanagri Character Recognition Model Using Deep Convolution Neural Network. Journal of Statistics and Management Systems ISSN: 0972-0510 (Print) 2169-0014

Goutam Sarker. (2020). A Survey on Convolution Neural Networks. IEEE REGION 10 CONFERENCE (TENCON) Osaka, Japan

Ghada Sokar, Prof. Elsayed E. Hemayed, and Dr. Mohamed Rehan. (2018). A Generic OCR Using Deep Siamese Convolution Neural Networks. 978-1-5386-7266-2/18 IEEE

Shriansh Srivastava, J. Priyadarshini, Sachin Gopal, Sanchay Gupta, and Har Shobhit Dayal. (2019). Optical Character Recognition on Bank Cheques Using 2D Convolution Neural Network. Applications of Artificial Intelligence Techniques in Engineering, Advances in Intelligent Systems and Computing https://doi.org/10.1007/978-981-13-1822-1_55

Meduri Avadesh and Navneet Goyal, (2018), Optical Character Recognition for Sanskrit using Convolution Neural Networks IAPR International Workshop on Document Analysis Systems 2018 13th IAPR International Workshop on Document Analysis Systems

*name of corresponding author