INCODING: JOURNAL OF INFORMATIC AND COMPUTER SCIENCE ENGINEERING
https://journal.mahesacenter.org/index.php/incoding/index || ISSN 2776-432X (online)
5 (2) 2025: 134-148
DOI:10.34007/incoding.v5i2.828

Pengenalan Tulisan Tangan Angka Pada Dataset MNIST Menggunakan Arsitektur SqueezeNet

Handwritten Digit Recognition on MNIST Dataset Using SqueezeNet Architecture

Elva Andrian & Susilawati*
Program Studi Teknik Informatika, Fakultas Teknik, Universitas Medan Area, Indonesia

Diterima: 16 April 2025; Direview: 20 April 2025; Disetujui: 24 April 2025

*Coresponding Email: susilawati@staff.uma.ac.id

Abstrak

Pengenalan tulisan tangan angka merupakan suatu proses mengenali dan mengidentikasi angka menggunakan algoritma kecerdasan buatan seperti Convolutional Neural Network (CNN). Penerapan pengenalan tulisan tangan angka bisa dikembangkan dan digunakan dalam identifikasi nomor kode pos pada surat, identifikasi nominal pada cek bank dan lainnya. Namun sebelum melakukan penerapan tersebut dibutuhkan pelatihan model pada algoritma yang akan digunakan supaya pengenalan angka menjadi tepat karena permasalahan yang dihadapi dalam mengenali tulisan tangan angka ialah gambar atau data tulisan yang beragam dan sulit diidentifikasi. Pada penelitian ini digunakan algoritma CNN dengan arsitektur SquueezeNet dengan dataset MNIST (Modified National Institute of Standards and Techology) yang terbagi atas 60000 data latih dan 10000 data uji. Platform yang digunakan untuk melakukan proses pelatihan dan pengujian yaitu Google Colab. Pelatihan dilakukan sebanyak 12 kali menggunakan hyperparameter seperti Optimizer yaitu Adam, SGD, dan RMSprop, Learning rate yaitu 0.1, 0.01, 0.001, 0.0001 dan Batch Size 64. Berdasarkan hasil penelitian dari 12 model yang dilatih diperoleh 1 model dengan hasil terbaik pada Optimizer yaitu Adam, Learning rate yaitu 0.0001 dan Batch Size 64 menghasilkan akurasi sebesar 99.11%.
Kata Kunci: CNN; MNIST; Squueze Net; Pengenalan Tulisan Tangan Angka.

Abstract

Handwritten digit recognition is a process of recognizing and identifying numbers using artificial intelligence algorithms such as Convolutional Neural Network (CNN). The application of handwritten number recognition can be developed and used in identifying postal code numbers on letters, identifying nominal amounts on bank checks and others. However, before carrying out the application, model training is needed on the algorithm that will be used so that number recognition is accurate because the problem faced in recognizing handwritten numbers is that images or written data are diverse and difficult to identify. In this study, the CNN algorithm was used with the SquueezeNet architecture with the MNIST (Modified National Institute of Standards and Technology) dataset which is divided into 60,000 training data and 10,000 test data. The platform used to carry out the training and testing process is Google Colab. Training was carried out 12 times using hyperparameters such as Optimizer namely Adam, SGD, and RMSprop, Learning rate namely 0.1, 0.01, 0.001, 0.0001 and Batch Size 64. Based on the research results from 12 trained models, 1 model was obtained with the best results on the Optimizer namely Adam, Learning rate namely 0.0001 and Batch Size 64 resulting in an accuracy of 99.11%.
Keywords: *CNN; MNIST; Squeeze Net; Handwritten Digit Recignition.*

PENDAHULUAN

Image classification atau klasifikasi gambar adalah hal yang mendasar dalam *computer vision* yang penerapannya dalam dunia nyata meliputi diagnosis medis, agrikultur, *e-commerce,* kontrol kualitas dan termasuk juga pada pengenalan tulisan angka yang menjadi fokus pada penelitian ini. *Image classification* melibatkan pemberian label pada gambar, kemudian melakukan proses pelatihan model pembelajaran untuk mengklasifikasikan atau mendeteksi objek yang beragam atau pola yang ada pada gambar tersebut [1].

Pengenalan tulisan tangan angka termasuk ke dalam *image classification* yang mana melibatkan proses pemberian label dan melakukan klasifikasi pada data yang akan digunakan. Pengenalan tulisan tangan angka merupakan kemampuan komputer untuk mengidentifikasi dan mengenali tulisan tangan angka manusia yang diambil dari berbagai media seperti gambar, kertas, layer sentuh dan yang lainnya yang kemudian diklasifikasikan menjadi 10 kelas terdiri dari 0-9 [2]. CNN merupakan satu dari sekian banyak algoritma yang bisa digunakan dalam proses pelatihan pengenalan tulisan angka. CNN adalah salah satu algoritma turunan dari *Deep Learning* yang mengambil data masukkan dalam bentuk gambar dan menentukan aspek yang akan dipelajari untuk memprediksi, mengidentifikasi atau membedakan gambar berdasarkan label yang telah ditentukan sebelumnya [3]. CNN memiliki beberapa model arsitektur seperti *LeNet, Alexnet, ResNet, GoogleNet* dan *SqueezeNet* yang mana arsitektur-arsitektur ini akan terus berkembang dan menyesuaikan terhadap kemajuan teknologi yang ada [4]. *SqueezeNet* merupakan salah satu model arsitektur CNN yang dirancang untuk mencapai akurasi setara dengan *AlexNet* namun dengan jumlah parameter yang jauh lebih sedikit dibanding arsitekur CNN lainnya, akan tetapi tetap memiliki performa yang optimal untuk melakukan klasifikasi gambar [5]. *SqueezeNet* pertama kali diperkenalkan pada tahun 2016 oleh tim peneliti yang terdiri dari Forrest N. Iandola, Song Han, Matthew W. Moskewicz, Khalid Ashraf, William J. Dally, dan Kurt Keutzer. Dengan jumlah parameter yang jauh lebih sedikit, *SqueezeNet* memiliki beberapa keuntungan seperti efisiensi komputasi, penghematan *bandwidth,* dan bisa diimplementasikan pada perangkat dengan memori yang terbatas [6].

Permasalahan dan tantangan yang dihadapi pada klasifikasi gambar pengenalan tulisan angka ialah berupa gaya tulisan yang beragam, variasi spasi dan tulisan tangan

yang inkonsisten [7]. Penelitian pengenalan tulisan tangan angka menggunakan dataset MNIST telah banyak dilakukan dan menghasilkan akurasi yang beragam. Penelitian yang dilakukan oleh Yevhen Chychkarov dengan menggunakan algoritma *Support Vector Machine* (SVM), *K-Nearest Neighbour* (KNN), *Random Forest* (RF) dan CNN yang mana dari ke empat algoritma ini menghasilkan akurasi terbaik pada CNN yaitu sebesar 97.6% [8]. Penelitian yang dilakukan oleh Ali Abdullah Yahya menggunakan CNN dengan *batch normalization* dan *optimizer* yaitu RMSprop menghasilkan akurasi sebesar 99.98% dan 99.40% dengan penambahan *noise* 50% pada dataset MNIST [9]. Penelitian dilakukan oleh Firta Panjaitan menggunakan SVM dan *feature extraction* yaitu *Histogram of Oriented Gradients* (HOG) dan *Principal Component Analysis* (PCA) mampu mencapai akurasi sebesar 98-99% [10]. Penelitian terkait lainnya juga dilakukan oleh Ruomin Zhu menggunakan *Neuromorphic Nanowire Network* dengan basis kernel CNN dan penambahan *theoretic metrics* berupa *Mutual Information (MI), Transfe Entropy (TE), dan Active Information Storage (AIS)* untuk menganalisis dinamik pembelajaran pada MNIST dan hasil akurasi yang didapat mencapai 98% [11].

Penelitian ini bertujuan untuk mengevaluasi performa arsitektur CNN *SqueezeNet* pada pengenalan tulisan tangan angka dengan mengukur nilai akurasi dan stabilitas model selama proses pelatihan. Nantinya akan digunakan tiga *optimizer* yang berbeda dan juga empat nilai *learning rate* yang berbeda. Hasil penelitian ini dapat memberikan wawasan mengenai efektivitas arsitektur CNN *SqueezeNet* dengan mengkomparasikannya dengan penelitian terkait

METODE PENELITIAN

Dataset

Pada penelitian ini, akan digunakan dataset MNIST. Dataset MNIST dibangun oleh Institute National Standard Technology (NIST) yang datanya berupa gambar tulisan tangan angka 0 sampai 9 dan dibagi dalam 60000 data latih dan 10000 data uji [12]. Dataset MNIST merupakan data bertipe grayscale (hitam-putih) berukuran 28*28 piksel yang tiap pikselnya memiliki rentang nilai 0 sampai 255 dimana nilai 0 berarti hitam, 255 berarti putih dan keabu-abuan diantara nilai 0 dan 255 tersebut [13].

Tabel 1 Distribusi Dataset MNIST

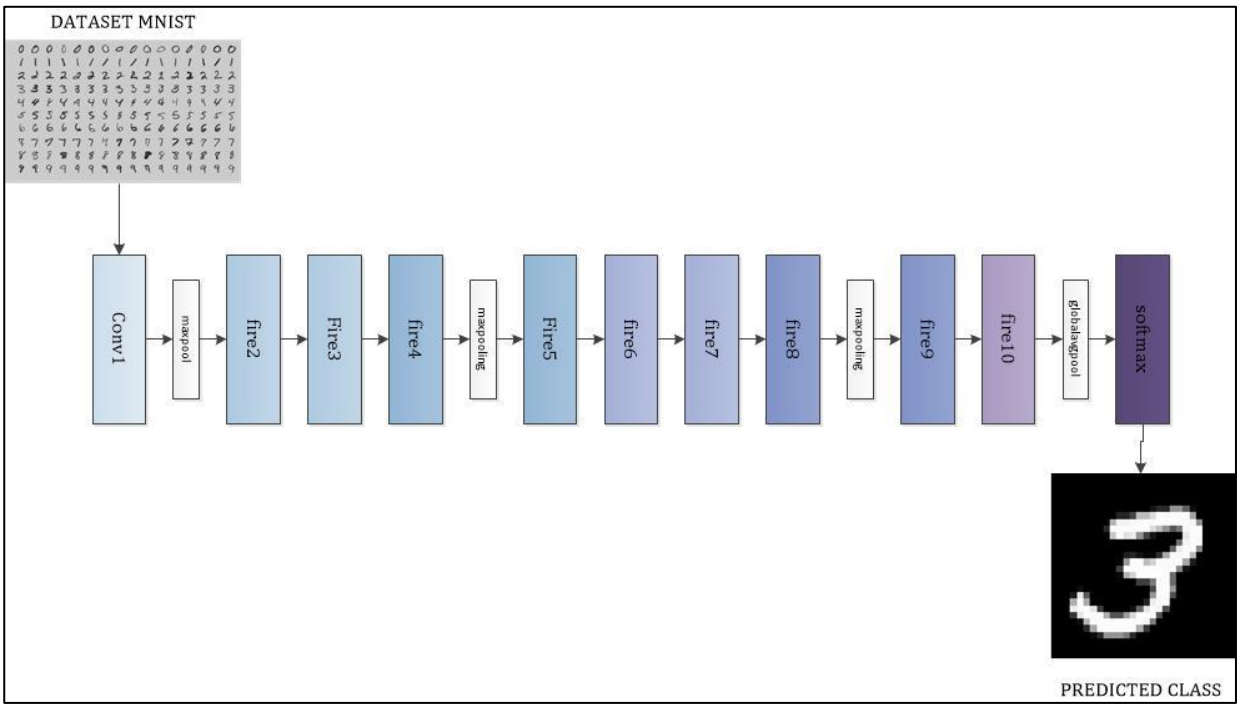
Angka	0	1	2	3	4	5	6	7	8	9	Total
Data Latih	5926	6742	5942	6155	5849	5421	5913	6265	5851	5936	60000
Data Uji	980	1135	1032	1010	982	892	958	1028	974	1009	10000

Gambar 1 Sampel Dataset MNIST

Normalisasi Data

Normalisasi data adalah sebuah proses mengubah nilai piksel gambar ke dalam skala tertentu dan bertujuan mempercepat proses pelatihan model, menghindari masalah numerik dan meningkatkan stabilitas model [14]. Pada dataset MNIST dilakukan normalisasi data dengan mengubah piksel gambar skala 0 sampai 255 menjadi 1 dan 0 dengan membagi 255 pada tiap piksel. Skala nilai pada piksel yang baru dapat membantu performa pelatihan model karena dapat mengurangi *noise* yang tinggi dan relevansi yang rendah [15].

Squeezenet



Gambar 2 Layer pada *SqueezeNet*

Squeezenet memanfaatkan strategi pemisahan konvolusional yang mana mengubah standar konvolusional menjadi *fire modules.* Di dalam *fire modules* terdiri dari *expand layer* dan *squeeze layer* yang setiap modulnya terdapat fungsi aktivasi ReLU yang bertujuan

menambahkan non-linearitas dan membantu model belajar fitur yang kompleks [16]. Pada *squeeze layer* tersusun atas 1x1 konvolusional yang berfungsi mengurangi bobot parameter dan pada *expand layer* tersusun atas gabungan 3x3 dan 1x1 konvolusional yang berfungsi untuk memastikan presisi model arsitektur *squeezenet* [17].

Pada gambar 2 di atas dapat dijelaskan secara berututan:

1. *Input* yang merupakan dataset MNIST berukuran 28x28 piksel yang telah dilakukan normalisasi
2. *Conv1* yaitu lapisan konvolusi awal untuk mengekstraksi fitur dasar dari input dataset MNIST
3. *Fire modules* (*fire2-fire10*) yaitu lapisan yang berfungsi mengekstraksi fitur secara bertahap
4. *Global Average Pooling* yaitu lapisan yang berfungsi mengurangi *overfitting* dan memperkecil ukuran fitur menjadi 1 dimensi
5. *Softmax* yaitu lapisan yang bertujuan menghasilkan probabilitas untuk masing-masing label (0-9)
6. *Output* yaitu berupa prediksi label terdiri dari 0-9

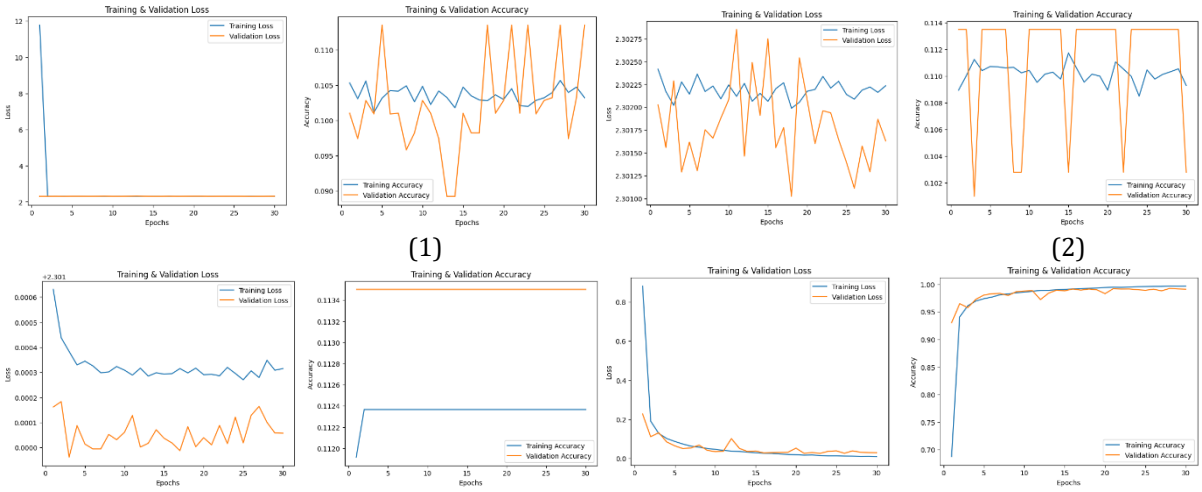
Fine-Tuning Hyperparameter

Fine-tuning hyperparameter ialah proses penyesuaian parameter yang nilainya diberikan oleh *user* sebelum melakukan pelatihan pada model dengan tujuan mengontrol cara model belajar dan mempengaruhi performa yang dihasilkan [18]. Beberapa parameter yang biasa dilakukan penyesuaian tersebut diantaranya adalah *epoch, learning rate, batch size* dan *optimizer* [19]. Pada penelitian ini akan menggunakan 4 nilai *learning rate* yaitu 0.1, 0.01,0.001 dan 0.0001, 3 *optimizer* yaitu SGD, Adam, RMSprop, *batch size 64* dan *epoch* sebanyak 30.

HASIL DAN PEMBAHASAN

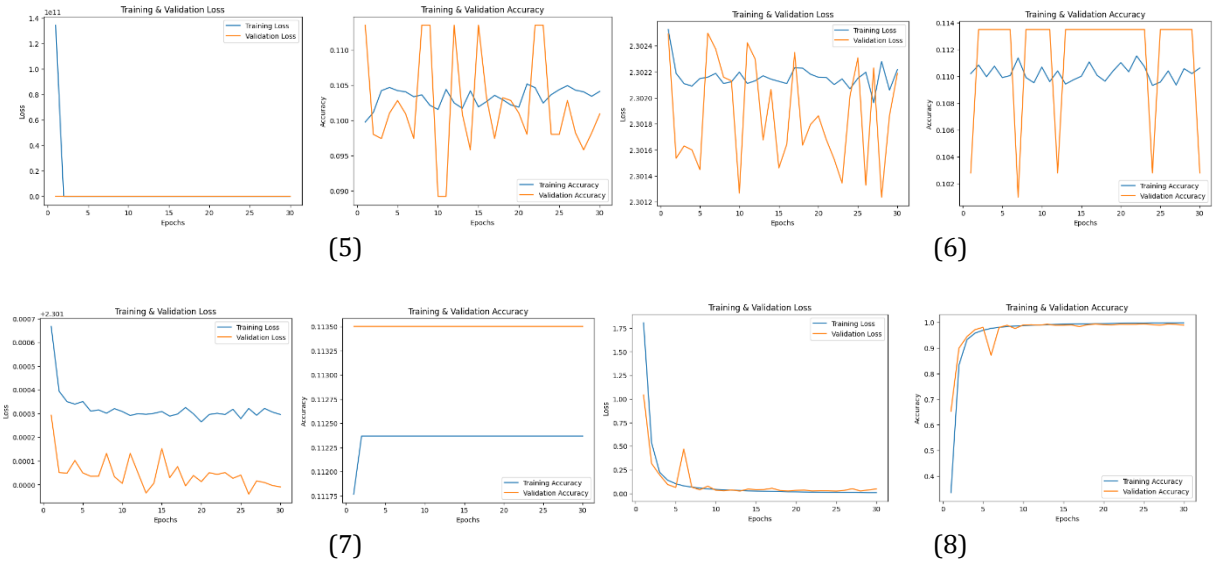
Model Pelatihan

Model dilatih dengan data latih yang dibagi 10% untuk validasi sehingga data menjadi 50000 untuk data latih dan 10000 untuk data validasi serta dilakukan penyesuaian *hyperparameter* terdiri dari 3 *optimizer* dan 4 nilai *learning rate* yang berbeda dengan *batch size* 64 dan *epoch* sebanyak 30. Ada sebanyak 12 model yang dilatih dan berikut grafik akurasi dan nilai *loss* dari hasil pelatihan yang dilakukan:



Gambar 3 Grafik Akurasi dan Loss Pelatihan Model ke 1-4

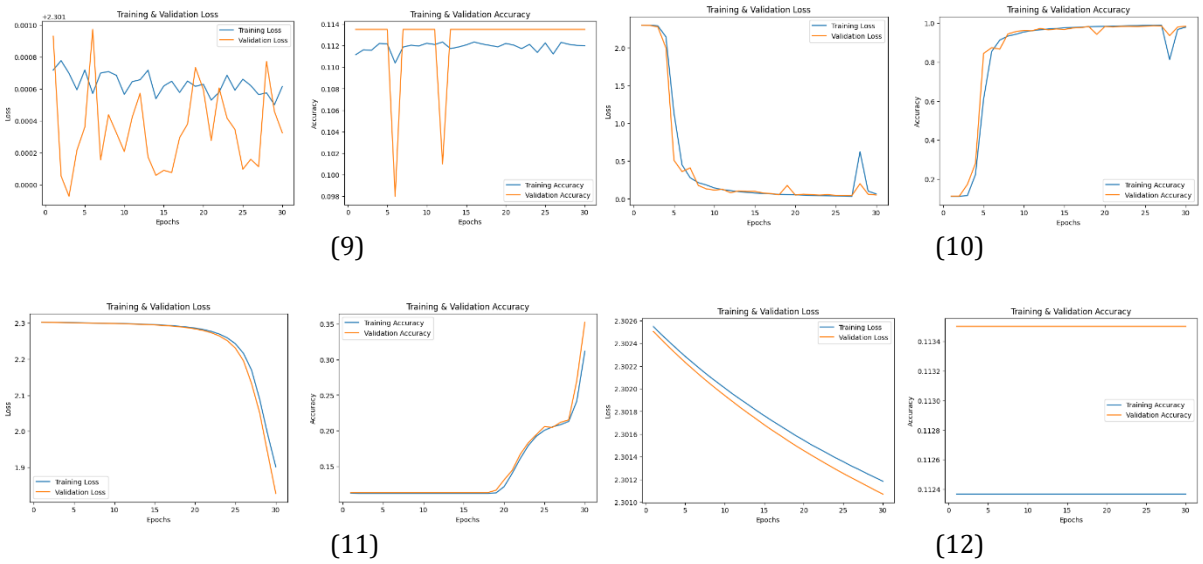
Pada gambar 3 di atas merupakan grafik akurasi dan loss pada pelatihan model ke 1-4 dengan *optimizer* Adam dan *learning rate* 0.1, 0.01, 0.001 dan 0.0001 berurut sesuai nomor yang tertera pada gambar. Dapat dilihat bahwa pada gambar di atas hanya model ke-4 dengan *learning rate* 0.0001 memiliki tingkat akurasi yang terus meningkat dan pada epoch 30 mencapai akurasi sebesar 99%, sementara model lainnya tidak mengalami peningkatan selama 30 epoch dan hanya menetap pada akurasi sekitar sebesar 11%.



Gambar 4 Grafik Akurasi dan Loss Pelatihan Model ke 5-8

Pada gambar 4 di atas merupakan grafik akurasi dan loss pada pelatihan model ke 5-8 dengan *optimizer* RMSprop dan *learning rate* 0.1, 0.01, 0.001 dan 0.0001 berurut sesuai nomor yang tertera pada gambar. Dapat dilihat bahwa pada gambar di atas hanya model

ke-8 dengan learning rate 0.0001 memiliki tingkat akurasi yang terus meningkat dan pada epoch 30 mencapai akurasi sebesar 98%, sementara model lainnya tidak mengalami peningkatan selama 30 epoch dan hanya menetap pada akurasi sekitar sebesar 11%.



Gambar 5 Grafik Akurasi dan Loss Pelatihan model ke 9-12

Pada gambar 5 di atas merupakan grafik akurasi dan loss pada pelatihan model ke 9-12 dengan *optimizer* SGD dan *learning rate* 0.1, 0.01, 0.001 dan 0.0001 berurut sesuai nomor yang tertera pada gambar. Dapat dilihat bahwa pada gambar di atas hanya model ke-10 dengan learning rate 0.01 memiliki tingkat akurasi yang terus meningkat dan pada epoch 30 mencapai akurasi sebesar 98%, sementara model ke-9 dan ke-12 tidak mengalami peningkatan dan hanya menetap pada akurasi sekitar 11% dan untuk model ke-11 cukup mengalami peningkatan akurasi pada epoch 20 sampai 30, namun hanya mencapai akurasi sekitar 35%

Berdasarkan 12 model di atas selama pelatihan model, dapat dilihat bahwa model ke-4 dengan *optimizer* Adam dan *learning rate* 0,0001, model ke-8 dengan optimizer RMSprop dan learning rate 0.0001 dan model ke-10 dengan optimizer SGD dan learning rate 0.01 adalah tiga model terbaik dengan akurasi yang terus mengalami peningkatan dan sebaliknya nilai *loss* terus mengalami penurunan dari epoch 1 hingga epoch 30.

Model Pengujian

Pengujian model dilakukan dengan data uji yang terdiri atas 10000 data. Di bawah ini ditampilkan tabel yang memberikan informasi mengenai akurasi pengujian pada 12 model

beserta data prediksi pada 10 label yaitu angka 0 sampai 9 yang dilengkapi dengan nilai *precision, recall dan f-1 score* di tiap label.

Tabel 2 Perbandingan Hasil Pada 12 Model

Model	Optimizer	Learning Rate	Accuracy	Label	Jumlah Aktual	Jumlah Terprediksi Benar	Accuracy Label	Precision	Recall	F-1 Score
Model 1	Adam	0.1	11.35%	0	980	0	0%	0	0	0
				1	1135	1035	91%	0.1135	0.91189427	0.20187357
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 2	Adam	0.01	11.35%	0	980	0	0%	0	0	0
				1	1135	0	0%	0	0	0
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	1028	100%	0.1028	1	0.18643453
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 3	Adam	0.001	11.35%	0	980	0	0%	0	0	0
				1	1135	1036	91%	0.1036	0.91277533	0.18607993
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 4	Adam	0.0001	11.35%	0	980	973	99%	0.0973	0.99285714	0.17723133
				1	1135	1127	99%	0.1127	0.99295154	0.20242479
				2	1032	1025	99%	0.1025	0.99321705	0.18582306
				3	1010	1008	100%	0.1008	0.9980198	0.18310627
				4	982	969	99%	0.0969	0.98676171	0.17647059
				5	892	882	99%	0.0882	0.98878924	0.16195373

				6	958	954	100%	0.0954	0.99582463	0.17411936
				7	1028	1012	98%	0.1012	0.9844358	0.18353283
				8	974	964	99%	0.0964	0.98973306	0.17568799
				9	1009	997	99%	0.0997	0.98810704	0.18112453
Model 5	RMSprop	0.1	11.35%	0	980	0	0%	0	0	0
				1	1135	0	0%	0	0	0
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	1009	100%	0.1009	1	0.18330457
Model 6	RMSprop	0.01	11.35%	0	980	0	0%	0	0	0
				1	1135	0	0%	0	0	0
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	1028	100%	0.1028	1	0.18643453
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 7	RMSprop	0.001	11.35%	0	980	0	0%	0	0	0
				1	1135	1035	91%	0.1035	0.91189427	0.18590031
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 8	RMSprop	0.0001	11.35%	0	980	977	100%	0.0977	0.99693878	0.17795993
				1	1135	1131	100%	0.1131	0.99647577	0.20314324
				2	1032	1032	100%	0.1032	1	0.1870921
				3	1010	1010	100%	0.101	1	0.18346957
				4	982	982	100%	0.0982	1	0.1788381
				5	892	958	107%	0.0958	1.07399103	0.17590892

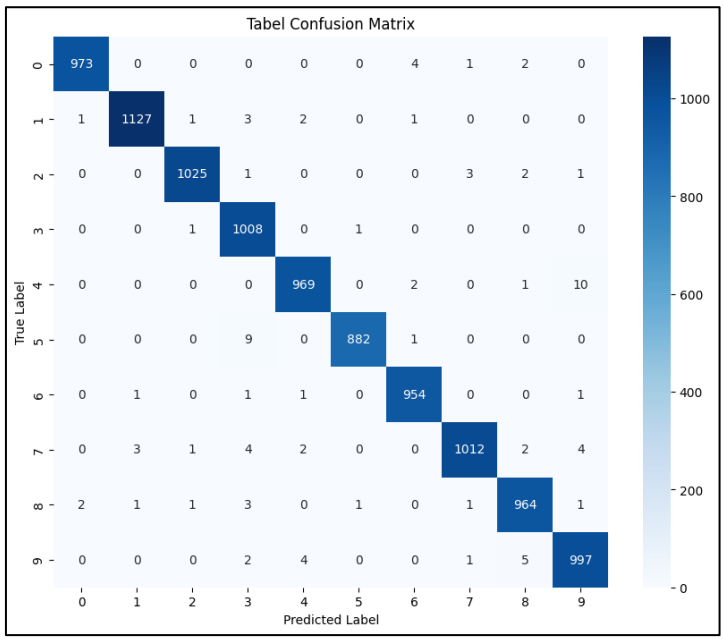
				6	958	1028	107%	0.1028	1.07306889	0.18762548
				7	1028	974	95%	0.0974	0.94747082	0.17664128
				8	974	986	101%	0.0986	1.01232033	0.17969747
				9	1009	986	98%	0.0986	0.97720515	0.17912617
Model 9	SGD	0.1	11.35%	0	980	0	0%	0	0	0
				1	1135	1035	91%	0.1035	0.91189427	0.18590031
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0
Model 10	SGD	0.01	11.35%	0	980	973	99%	0.0973	0.99285714	0.17723133
				1	1135	1116	98%	0.1116	0.98325991	0.20044903
				2	1032	1021	99%	0.1021	0.98934109	0.1850979
				3	1010	993	98%	0.0993	0.98316832	0.18038147
				4	982	974	99%	0.0974	0.99185336	0.17738117
				5	892	880	99%	0.088	0.98654709	0.16158649
				6	958	923	96%	0.0923	0.96346555	0.1684614
				7	1028	1019	99%	0.1019	0.99124514	0.18480232
				8	974	974	100%	0.0974	1	0.17751048
				9	1009	986	98%	0.0986	0.97720515	0.17912617
Model 11	SGD	0.001	11.35%	0	980	0	0%	0	0	0
				1	1135	1123	99%	0.1123	0.98942731	0.20170633
				2	1032	23	2%	0.0023	0.02228682	0.00416969
				3	1010	981	97%	0.0981	0.97128713	0.17820163
				4	982	349	36%	0.0349	0.35539715	0.06355855
				5	892	0	0%	0	0	0
				6	958	331	35%	0.0331	0.34551148	0.06041248
				7	1028	716	70%	0.0716	0.69649805	0.12985129
				8	974	0	0%	0	0	0

				9	1009	0	0%	0	0	0
Model 12	SGD	0.0001	11.35%	0	980	0	0%	0	0	0
				1	1135	1035	91%	0.1035	0.91189427	0.18590031
				2	1032	0	0%	0	0	0
				3	1010	0	0%	0	0	0
				4	982	0	0%	0	0	0
				5	892	0	0%	0	0	0
				6	958	0	0%	0	0	0
				7	1028	0	0%	0	0	0
				8	974	0	0%	0	0	0
				9	1009	0	0%	0	0	0

Pada tabel 2 di atas, dapat diketahui tingkat akurasi 12 model dari pengujian yang dilakukan dengan data uji. Ada beberapa kesamaan tingkat akurasi untuk beberapa model terutama model yang memiliki akurasi rendah yaitu model ke-1 dengan *optimizer* Adam dan *learning rate* 0.1, model ke-3 dengan *optimizer* Adam dan *learning rate* 0.001, model ke-7 dengan *optimizer* RMSprop dan *learning rate* 0.001, model ke-9 dengan *optimizer* SGD dan *learning rate* 0.1 dan model ke-12 dengan *optimizer* SGD dan *learning rate* 0.0001 yang mana kelima model ini memiliki tingkat akurasi sebesar 11.35%. Pada model ke-2 dengan *optimizer* Adam dan learning 0.01 dan model ke-6 dengan *optimizer* RMSprop dan *learning rate* 0.01 juga memiliki kesamaan tingkat akurasi yaitu sebesar 10.28%. Sementara pada model ke-5 dengan *optimizer* RMSprop dan *learning rate* 0.1 dan model ke-11 dengan *optimizer* SGD dan *learning rate* 0.001 masing-masing memiliki tingkat akurasi sebesar 10.09% dan 35.23%. Untuk model dengan akurasi tertinggi ada 3 yaitu model model ke-4 dengan *optimizer* Adam dan *learning rate* 0.0001 dengan akurasi 99.11%, model ke-8 dengan *optimizer* RMSprop dan *learning rate* 0.0001 dengan akurasi 98.90% dan yang terakhir yaitu model ke-10 dengan *optimizer* SGD dan *learning rate* 0.01 dengan akurasi 98.32%.

Evaluasi Akhir

Evaluasi akhir dilakukan dengan menampilkan grafik confusion matrix yang memberikan informasi mengenai kesalahan prediksi pada 10 label yang terdiri dari angka 0 sampai 9. Pada proses pengujian, dapat diketahui bahwa model ke-4 dengan optimizer Adam dan learning rate 0.0001 adalah model yang terbaik dengan tingkat akurasi mencapai 99.11% dan model ke-4 inilah yang kemudian dilakukan evaluasi. Berikut di bawah ini grafik confusion matrix pada model ke-4 tersebut



Gambar 6 Grafik Confusion Matrix

Berdasarkan gambar 6 di atas dapat diketahui informasi label yang terprediksi dengan benar dan terprediksi sebagai label lain. Kesalahan prediksi terjadi paling banyak yaitu pada label angka 9 yang diprediksi sebanyak 5 kali sebagai angka 8, pada label angka 5 yang diprediksi sebanyak 9 kali sebagai angka 3 dan pada label angka 4 yang diprediksi sebanyak 10 kali sebagai angka 9.

SIMPULAN

Penelitian ini melakukan klasifikasi pada dataset MNIST untuk melakukan pengenalan tulisan tangan angka menggunakan arsitektur CNN Squeezenet yang dilakukan *fine-tuning hyperparameter* yaitu menggunakan *learning rate* terdiri dari 0.1, 0.01, 0.001 dan 0.0001, *optimizer* berupa Adam, SGD, dan RMSprop, *batch size 64* dan *epoch* sebanyak 30 yang mana menghasilkan 12 model. Berdasarkan 12 model yang telah dilakukan pengujian, diperoleh 1 model terbaik kombinasi *hyperparameter* yaitu Adam dengan *learning rate* 0.0001 yang menghasilkan nilai akurasi sebesar 99.11%. Pada grafik *confution matrix*, terdapat aktual label dan label yang diprediksi dan model yang terbaik ini banyak terjadi kesalahan prediksi pada label angka 5 yang diprediksi sebanyak 9 kali sebagai label angka 3 dan pada label angka 4 yang diprediksi sebanyak 10 kali sebagai angka 9. Penelitian menggunakan dataset yang sama yaitu MNIST dilakukan oleh Ali Abdullah Yahya menghasilkan akurasi sebesar 99.98% dengan menerapkan algoritma

CNN menggunakan *optimizer* RMSprop. Jika dibandingkan dengan hasil CNN *SqueezeNet* pada penelitian ini hanya berbeda 0.87%, dengan ini dikatakan bahwa CNN *SqueezeNet* mampu untuk mengklasifikasikan dataset MNIST untuk pengenalan tulisan tangan angka dengan baik. Diharapkan penelitian ini dapat menjadi referensi untuk penelitian mendatang dan menambahkan nilai *hyperparameter* yang lebih banyak supaya hasil akurasi meningkat dan menjadi lebih baik lagi.

DAFTAR PUSTAKA

[1] M. Khalil, A. Khalil, and A. Ngom, "A Comprehensive Study of Vision Transformers in Image Classification Tasks." 2023. [Online]. Available: http://arxiv.org/abs/2312.01232

[2] N. F. Abubacker and M. Raheem, "Handwritten Digit Recognition Using Machine Learning," *J. Theor. Appl. Inf. Technol.*, vol. 102, no. 5, pp. 2172–2180, 2024, doi: 10.54254/2755-2721/5/20230613.

[3] S. Dewi, F. Ramadhani, and S. Djasmayena, "Klasifikasi Jenis Jerawat Berdasarkan Gambar Menggunakan Algoritma CNN (Convolutional Neural Network)," *Hello World J. Ilmu Komput.*, vol. 3, no. 2, pp. 68–73, 2024, doi: 10.56211/helloworld.v3i2.518.

[4] M. Kevin Santosa, M. Hanindia Prami Swari, and A. Nugroho Sihananto, "Implementasi Arsitektur Alexnet Dan Resnet34 Pada Klasifikasi Citra Penyakit Daun Kentang Menggunkan Transfer Learning," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 5, pp. 3293–3301, 2024, doi: 10.36040/jati.v7i5.7337.

[5] W. Setiawan, A. Ghofur, F. Hastarita Rachman, and R. Rulaningtyas, *Deep Convolutional Neural Network AlexNet and Squeezenet for Maize Leaf Diseases Image Classification*. KINETIK: Game Technology, Information System, Computer Network, Computing, Electronics and Control, 2021. doi: 10.22219/kinetik.v6i4.1335.

[6] B. Gope, S. Pande, N. Karale, S. Dharmale, and P. Umekar, *Handwritten digits identification using mnist database via machine learning models*, vol. 1022, no. 1. PURPOSE-LED PUBLISHING, 2021. doi: 10.1088/1757-899X/1022/1/012108.

[7] L. Ming Seng, B. Bang Chen Chiang, Z. Arabee Abdul Salam, G. Yih Tan, and H. Tong Chai, "MNIST handwritten digit recognition with different CNN architectures," *J. Appl. Technol. Innov.*, vol. 5, no. 1, pp. 2600–7304, 2021.

[8] Y. Chychkarov, A. Serhiienko, I. Syrmamiikh, and A. Kargin, *Handwritten digits recognition using SVM, KNN, RF and deep learning neural networks*, vol. 2864. CEUR-WS.org, 2021. doi: 10.32782/cmis/2864-44.

[9] A. A. Yahya, J. Tan, and M. Hu, "A novel handwritten digit classification system based on convolutional neural network approach," *Sensors*, vol. 21, no. 18, 2021, doi: 10.3390/s21186273.

[10] F. Panjaitan, R. S. Simbolon, and J. Batubara, "Handwritten Digit Recognition Using Support Vector Machine (SVM) with Radial Basis Function (RBF) Kernel," *J. Basic Sci. Technol.*, 2025.

[11] R. Zhu *et al.*, *MNIST classification using Neuromorphic Nanowire Networks*. Association for Computing Machinery, 2021. doi: 10.1145/3477145.3477162.

[12] Susilawati, "Algoritma Restricted Boltzmann Machines (RBM) untuk Pengenalan Tulisan Tangan Angka," *Semin. Nas. Teknol. Inform.*, pp. 140–148, 2017.

[13] T. Soupizet, Z. Jouni, S. Wang, A. Benlarbi-Delai, and P. M. Ferreira, "Analog Spiking Neural Network Synthesis for the MNIST," *J. Integr. Circuits Syst.*, vol. 18, no. 1, 2023, doi: 10.29292/jics.v18i1.663.

[14] F. Adams, R. A. Dwi Anggoro, M. B. Satria, A. W. Oktavia, and N. Chamidah, "Perbandingan Normalisasi Data untuk Klasifikasi Wine Menggunakan Algoritma Naïve Bayes, Decision Tree, dan Support Vector Machine," *Semin. Nas. Mhs. Ilmu Komput. dan Apl.*, vol. 2, no. 2, pp. 260–268, 2021, [Online]. Available: https://conference.upnvj.ac.id/index.php/senamika/article/view/1744

[15] Gde Agung Brahmana Suryanegara, Adiwijaya, and Mahendra Dwifebri Purbolaksono, "Peningkatan Hasil Klasifikasi pada Algoritma Random Forest untuk Deteksi Pasien Penderita Diabetes Menggunakan Metode Normalisasi," in *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, 2021, pp. 114–122. doi: 10.29207/resti.v5i1.2880.

[16] B. Qiang *et al.*, "SqueezeNet and Fusion Network-Based Accurate Fast Fully Convolutional Network for Hand Detection and Gesture Recognition," *IEEE Access*, vol. 9, pp. 77661–77674, 2021, doi: 10.1109/ACCESS.2021.3079337.

[17] P. Theerthagiri, A. U. Ruby, J. G. C. Chandran, T. H. Sardar, and A. Shafeeq B. M, "Deep SqueezeNet learning model for diagnosis and prediction of maize leaf diseases," *J. Big Data*, vol. 11, no. 1, 2024, doi: 10.1186/s40537-024-00972-z.
[18] M. Sipper, "High Per Parameter: A Large-Scale Study of Hyperparameter Tuning for Machine Learning Algorithms," *Algorithms*, vol. 15, no. 9, 2022, doi: 10.3390/a15090315.
[19] Afis Julianto, Andi Sunyoto, and Ferry Wahyu Wibowo, "Optimasi Hyperparameter Convolutional Neural Network Untuk Klasifikasi Penyakit Tanaman Padi," *Tek. Teknol. Inf. dan Multimed.*, vol. 3, no. 2, pp. 98–105, 2022, doi: 10.46764/teknimedia.v3i2.77.