Pokémon Classification with Deep Learning Applications

Mert Çalışkan, N22233552, mertcaliskan@hacettepe.edu.tr

Hacettepe University, Institute of Informatics, Turkey

Abstract. This study is a final research project conducted as part of the VBM 689 Deep Learning Applications course at Hacettepe University Institute of Informatics. The aim of the project is to classify Pokémon using deep learning techniques based on their visual and type features, utilizing the "Pokemon Image Dataset" containing 809 different Pokémon characters on the Kaggle platform. Within the scope of the study, a total of ten different models were developed using Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN), and their performances were evaluated with metrics such as accuracy, precision, recall, and F1 score. The results revealed that some models experienced overfitting problems and their generalization capabilities were limited. The study investigated how data augmentation and regularization techniques affect model performance and provided improvement suggestions for future work.

Keywords: Pokémon, Classification, Deep Learning, MLP, CNN.

1 Introduction

This study is a final research project conducted as part of the VBM 689 Deep Learning Applications course at Hacettepe University Institute of Informatics. The project aims to classify Pokémon using deep learning techniques based on their visual and type features, utilizing the "Pokemon Image Dataset" containing 809 different Pokémon characters on the Kaggle platform. In the project, a total of ten models were created using various layer structures, optimization algorithms, regularization techniques, and learning rates. The performance of each model was compared to investigate the structure that would classify Pokémon types most accurately.

2 Related Work

Deep learning-based methods are frequently used in studies conducted on similar datasets. Some important studies in the relevant literature are as follows:

1. **Pokepedia: Pokémon Image Classification Using Transfer Learning, Vishal Subbiah, Nivedhitha Gandhi, Sruti Srinivasan, International Information and Engineering Technology Association, Vol. 26, No. 5, 2021.**
   In this study, the visual classification performance was tested using transfer learning methods on a dataset covering 809 different Pokémon characters. Various pretrained models such as VGG16, VGG19, ResNet101, MobileNetV2, DenseNet201, EfficientNetB7, and EfficientNetV2L were used. The ResNet101 model achieved the best performance with a 95.6% accuracy rate. The study demonstrates the effectiveness of transfer learning approaches in classifying Pokémon images.
2. **Deep Learning Model of Image Classification Using Machine Learning, Hindawi Publishing Corporation, 2021**
   This study analyzed the use of deep learning models for image classification, particularly Convolutional Neural Networks (CNN) and other similar technologies. Experiments were conducted with advanced models like ResNet, Mask-RCNN, and Faster R-CNN, and the performance of basic CNN structures like LeNet was compared. It was stated that CNN models produced superior classification results compared to classical machine learning algorithms.
3. **Transfer Learning for Image Classification using VGG19, M. Shakil, International Journal of Innovative Research in Computer Science & Technology, 2019**
   This research focuses on how the VGG19 model can be applied using transfer learning methods and examines the effectiveness of this approach on Pokémon visual classification. It was shown that VGG19 achieves higher performance than other models when transfer learning is used.
4. **Gotta Generate ’em All! Pokemon (With Deep Learning), Devi Acharya, Rehaf Aljammaz, Beth Oliver, ve Mirek Stolee, University of California, Santa Cruz, 2020**

This research investigates the creation and classification of various Pokémon cards using deep learning algorithms. The research focuses on models such as Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs). These models were used to generate realistic Pokémon visuals and corresponding text-based descriptions. The study explains in detail how the models are trained based on element types and the data augmentation techniques used in this process. It also provides comprehensive analyses of the metrics used to evaluate model performance and the results obtained/

5. **One-shot Pokemon Classification, Xiang Li, Qixu Chen, Xinran Zhao, Hong Kong University of Science and Technology, 2019**
   This study focused on developing a high-accuracy Pokémon classification model using few training examples. Particularly, one-shot learning models like Siamese Networks and Triplet Networks based on deep learning were examined. The research discusses how these models can be effectively trained on a single example and how they can improve their ability to recognize new Pokémon types. The study details the customized loss functions and optimization techniques used to enhance the model's learning process. The results demonstrate how these techniques can produce effective results in data-limited situations.
6. **Comparative Analysis of Deep Learning Image Detection Algorithms, E. Abirami, Journal of Computer Science & Research, 2019**
   The applicability and performance of important image recognition algorithms such as SSD (Single Shot Detection), Faster R-CNN, and YOLO (You Only Look Once) on Pokémon classification were compared. It was determined that SSD and Faster R-CNN achieved high accuracy rates, especially on complex image datasets.
7. **Machine Learning and Deep Learning Based Computational Techniques in Image Recognition, A. Khan, Journal of Computational Science, 2020**
   This article examines how machine learning and deep learning techniques approach various image recognition problems, such as microorganism classification. It emphasizes that deep learning techniques are more successful than classical algorithms in visually based applications like Pokémon classification and that the potential applications of these techniques can be expanded.
8. **Image Classification Using Classic and Deep Learning Techniques, J. Smith, IEEE Transactions on Neural Networks and Learning Systems, 2020**
   In this study, image classification tasks were addressed using classic and deep learning techniques. The performance of various deep learning models such as CNN, LeNet, AlexNet, VGG, and ResNet was observed to be superior compared to classical algorithms. It was examined in detail how these techniques offer effective solutions in complex visual recognition problems.
9. **A Survey on Dataset Quality in Machine Learning, G. Johnson, ACM Computing Surveys, 2019**
   The importance of dataset quality in the success of machine learning models was emphasized. In areas like Pokémon classification, it was analyzed that correctly labeled datasets with a balanced class distribution directly affect classification

performance. This comprehensive review on data quality provides critical suggestions for improving data preparation processes.

10. **A Systematic Review of Machine Learning Techniques for Cattle Identification, D. Smith, Journal of Agricultural and Food Science, 2021**
    The effectiveness of machine learning techniques used in animal identification applications on similar datasets like Pokémon classification was discussed. These studies with different datasets and features demonstrate the adaptation and success of machine learning and image processing techniques in various application areas.
11. **ImageNet Benchmark (Image Classification), O. Williams, ImageNet Large Scale Visual Recognition Challenge, 2020**
    A performance comparison of OmniVec (ViT), EfficientNet, and ResNet models used in the classification competition on the ImageNet dataset was conducted. Each of these models achieved high accuracy rates, illustrating how they can deliver superior performance on large and complex datasets.
12. **Deep Learning Model of Image Classification Using Machine Learning Techniques, R. Sundaresan, Computational and Mathematical Methods in Medicine, 2021**
    This study takes a broad perspective on the fundamentals and application areas of various deep learning models and algorithms, detailing how CNN and transfer learning techniques can be used in different image classification problems. It focuses on the configuration and optimization of models necessary for effective classification in complex visual datasets like Pokémon.

While previous studies mainly focused on transfer learning methods, this project adopts a hybrid approach encompassing both transfer learning and learning from scratch techniques. This way, the generalization capability of the model was enhanced and more specialized feature extraction was achieved. Although the models in Study 1 and Study 2 offered high accuracy rates, these models were observed to have overfitting tendencies. In our project, improved regularization techniques and data augmentation strategies were used to reduce overfitting.

3 Methods

This section details the methodology used for classifying Pokémon characters. The project consists of stages such as data set processing, model optimization, regularization techniques, classification methods, and performance evaluations.

General Methodology and Flowchart

1. . General Methodology and Flowchart:
    (1) The 'Pokemon Image Dataset' obtained from Kaggle is loaded.
    (2) Missing data in the dataset are checked, and unnecessary columns for classification such as 'Type2', 'Evolution' are removed.
    (3) Images are resized to 120x120 pixels and pixel values are normalized to the 0-1 range.
    (4) Image file paths are created using Pokémon names.
2. Data Augmentation:
    (1) The generalization ability of the model is increased, and overfitting is prevented by applying techniques such as random rotation, shifting, cropping, zooming, and horizontal flipping on the images in the training set.
    (2) Data augmentation techniques were used in MLP2, MLP3, MLP4, CNN2, CNN3, and CNN4 models.
3. Feature Extraction and Model Configuration:
    (1) Features are extracted from images based on pixel values.
    (2) Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN) models are designed and created.
4. Model Training and Evaluation:
    (1) Models are trained on the determined training and validation sets.
    (2) Various optimization algorithms and loss functions are used for optimization.
    (3) The training process is supported with methods such as early stopping and learning rate reduction.
    (4) The performance of the models is evaluated with metrics such as accuracy, precision, recall, and F1 score, and their performance during and after training is visualized with confusion matrix, loss, and accuracy graphs.
5. Analysis and Discussion of Results:
    (1) The performance results of the models are analysed and discussed.
    (2) Suggestions for future work are presented.

3.1 Data Set

In this project, the "Pokemon Image Dataset" obtained from the Kaggle platform was used. The dataset contains 809 different Pokémon images and 18 type labels from the first to the seventh generation. The dataset consists of two main files:

1. **The "pokemon.csv" file** contains the following information for each Pokémon character::
   - Name: The name of the Pokémon (e.g. "Bulbasaur", "Charizard").
   - Type1: The primary type of the Pokémon (e.g. Normal, Fire, Water, Electric, Grass, Ice, Fighting, Poison, Ground, Flying, Psychic, Bug, Rock, Ghost, Dragon, Dark, Steel, Fairy).
   - Type2: The secondary type of the Pokémon (if any).
   - Evolution: The evolution chain (e.g.. "Bulbasaur" -> "Ivysaur" -> "Venusaur").
2. **The "images" folder** contains RGB images in PNG format, 120x120 pixels in size, for each Pokémon character (809 in total). Each image is named according to the name of the corresponding Pokémon (e.g., "bulbasaur.png", "charizard.png").

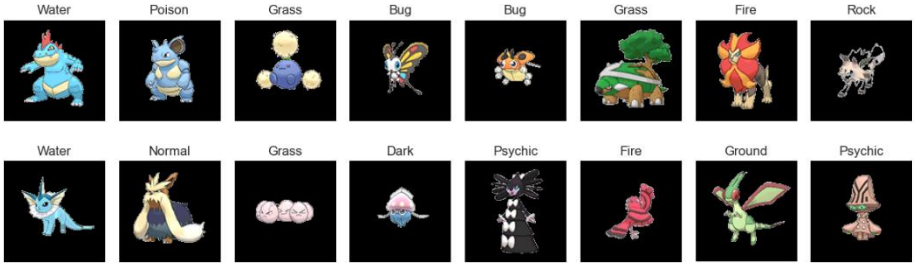

Figure. 1. Images of Randomly Selected Pokémon Types

The numerical distribution of each Pokémon type was examined in detail and is presented in the table below.

Table 1. Numerical Distribution of Pokemon Types

Type1	Count
Water	114
Normal	105
Grass	78
Bug	72
Fire	53
Psychic	53
Rock	46
Electric	40
Poison	34
Ground	32
Dark	29
Fighting	29
Ghost	27
Dragon	27
Steel	26
Ice	23
Fair	18
Flying	3

The dataset is labelled for training classification algorithms, and the project aims to classify Pokémon types using this dataset.

The dataset is divided into training, validation, and test sets in the following proportions to evaluate the model's performance:

- Training Set: 70% (566 samples)
- Validation Set: 15% (122 samples)
- Test Set: 15% (121 samples)

3.2 Optimization and Regularization Techniques

In this study, various optimization and regularization techniques were applied to maximize model performance and prevent overfitting.

Optimization Techniques

- Adam (Adaptive Moment Estimation): Combines adaptive learning rates and a momentum term to provide a fast and efficient learning process. The Adam optimizer was preferred in our study because it provided rapid convergence and produced stable results.
- Stochastic Gradient Descent (SGD): Accelerates the learning process and improves generalization capability by calculating the gradient for each mini-batch in large datasets. The SGD optimizer was tested as an alternative to the Adam optimizer in some experiments in our study and achieved certain accuracy levels in specific models.
- RMSProp: Calculates separate learning rates for each parameter and adjusts these rates using the moving average of the gradients. RMSProp was tested on various models in our study but was found to be less successful than the Adam optimizer.
- Nadam (Nesterov-accelerated Adaptive Moment Estimation): Combines the advantages of the Adam optimizer with Nesterov momentum. The Nadam optimizer was also tested in our study but generally yielded similar results to the Adam optimizer.

Regularization Techniques

- Dropout: Prevents the model from overfitting by randomly deactivating some neurons in the network. Dropout rates of 0.3, 0.5, and 0.7 were tested in our study, and the best results were obtained with a rate of 0.5.
- Early Stopping: Stops training if the performance on the validation set does not improve for a certain period. Early stopping was applied when the validation loss did not improve for 15 epochs during training.
- L2 Regularization (Ridge Regression): Adds the square of the sum of the weights as a penalty term and prevents overfitting by penalizing large weights. L2 regularization was used in conjunction with dropout for some models in our study.
- L1_L2 Regularization (Elastic Net): Works by adding the sum of the absolute and square values of the weights to the loss function of the model. This technique supports the tendency of L1 regularization to push weights towards zero and simplify the model, while L2 regularization penalizes large weight values to prevent overfitting. This dual approach improves the overall learning structure of the model and is particularly effective in datasets with a large number of features and a high risk of overfitting. The L1_L2 regularization used in the project ensured that the model obtained more stable and generalizable results.
- Batch Normalization: Accelerates and stabilizes the learning process by normalizing the activations of each mini-batch. Batch normalization improved the perfor-

mance of the model during the training process, and significant performance gains were observed in some models.

The application of these optimization and regularization techniques improved the performance of Pokémon classification models and reduced the risk of overfitting. Experiments have shown that when used effectively, these techniques can significantly improve the overall accuracy and generalization ability of the model.

3.3 Classification Methods

In this study, two main deep learning models were used to classify Pokémon types: Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN). During the model training process, a total of ten different models were experimented with. These models were designed using different layer structures, optimization algorithms, regularization methods, and learning rates. The performance of each model was evaluated comparatively to obtain the most effective classification results.

Multilayer Perceptron (MLP)

MLP's consist of an input layer, one or more hidden layers, and an output layer. These models work by calculating weighted sums to process data and transmitting signals through activation functions.

MLP Model Architectures Used

"MLP Basic" Model:

- Input Layer: Accepts 120x120 pixel color images.
- Flatten Layer: Converts the images from the input layer into a flat vector.
- Dense Layers:
  - First Layer: 512 neurons, ReLU activation function.
  - Second Layer: 256 neurons, ReLU activation function.
- Output Layer: Contains a number of neurons equal to the number of Pokémon types, softmax activation function is used.
- Optimization and Loss Function: Adam optimization algorithm (learning rate: 0.001), categorical cross entropy.
- Data Augmentation Techniques: Not used.

"MLP1" Model:

- Input Layer: Designed for 120x120x3 color images.
- Flatten Layer: Converts images to a flat vector.
- Dense Layers:
  - First Layer: 32 neurons, l1_l2 regularizer (l1=1e-5, l2=1e-4), HeNormal initialization, ReLU activation function, BatchNormalization, Dropout (0.5).
  - Second Layer: 32 neurons, l1_l2 regularizer (l1=1e-5, l2=1e-4), HeNormal initialization, ReLU activation function, BatchNormalization, Dropout (0.5).
- Output Layer: Number of neurons equal to the number of Pokémon types, softmax activation function.
- Optimization and Loss Function: Adam optimization algorithm (learning rate: 0.001), categorical cross entropy.
- Data Augmentation Techniques: Not used.

"MLP2" Model:

- Input Layer: Designed for 120x120 pixel color images.
- Flatten Layer: Converts images to a flat vector.
- Dense Layers:
  - First Layer: 32 neurons, l1_l2 regularizer (l1=1e-5, l2=1e-4), HeNormal initialization, ReLU activation function, BatchNormalization, and 0.5 Dropout.
  - Second Layer: 32 neurons, l1_l2 regularizer (l1=1e-5, l2=1e-4), HeNormal initialization, ReLU activation function, BatchNormalization, and 0.5 Dropout.
- Output Layer: Number of neurons equivalent to the number of Pokémon types, softmax activation function.
- Optimization and Loss Function: RMSprop optimization algorithm is used, learning rate 0.001, categorical cross entropy loss function.
- Data Augmentation Techniques: Used.

"MLP3" Model:

- Input Layer: Accepts 120x120x3 color images.
- Flatten Layer: Flattens the input images.
- Dense Layers:
  - First Layer: 1024 neurons, BatchNormalization, ReLU activation function, Dropout (0.3).
  - Second Layer: 512 neurons, BatchNormalization, ReLU activation function, Dropout (0.3).
  - Third Layer: 256 neurons, BatchNormalization, ReLU activation function, Dropout (0.3).
- Output Layer: Number of neurons equivalent to the number of Pokémon types, softmax activation function.
- Optimization and Loss Function: Adam optimization algorithm, learning rate 0.001, categorical cross entropy.
- Data Augmentation Techniques: Used.

"MLP4" Model:

- Input Layer: Designed for 120x120x3 color images.
- Flatten Layer: Converts images to a one-dimensional vector.
- Dense Layers:
  - First Layer: 512 neurons, L2 regularization (0.001), BatchNormalization, ReLU activation function, Dropout (0.5).
  - Second Layer: 256 neurons, L2 regularization (0.001), BatchNormalization, ReLU activation function, Dropout (0.3).
  - Third Layer: 128 neurons, L2 regularization (0.001), ReLU activation function, Dropout (0.2).
- Output Layer: Number of neurons equivalent to the number of Pokémon types, softmax activation function.
- Optimization and Loss Function: Adam optimization algorithm, learning rate 0.0005, categorical cross entropy.

- Data Augmentation Techniques: Used.

Convolutional Neural Networks (CNNs)

CNNs are effective for working with high-dimensional data like images due to their ability to detect local features. Their basic structures consist of convolutional layers, activation functions, pooling layers, and dense layers.

CNN Model Architectures Used

"CNN Basic" Model:

- Input Layer: Accepts 120x120x3 color images.
- Convolutional Layers:
  - First Layer: 32 filters, 3x3 kernel size, ReLU activation.
  - Max-Pooling Layer: 2x2 window size.
  - Second Layer: 64 filters, 3x3 kernel size, ReLU activation.
  - Max-Pooling Layer: 2x2 window size.
- Flatten Layer: Converts multi-dimensional feature maps into a flat vector.
- Dense Layer: 256 neurons, ReLU activation.
- Output Layer: Softmax activation function to recognize 18 different Pokémon types.
- Optimization and Loss Function: Adam optimization algorithm (learning rate: 0.001), categorical cross entropy.
- Data Augmentation Techniques: Not used.

"CNN1" Model:

- Input Layer: Designed for 120x120x3 color images.
- Convolutional and Pooling Layers:
  - First Layer: 16 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size, Dropout (0.25).
  - Second Layer: 32 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size, Dropout (0.25).
  - Third Layer: 64 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size, Dropout (0.25).
- Flatten Layer: Converts multi-dimensional feature maps into a one-dimensional vector.
- Dense Layer: Includes 128 neurons and uses ReLU activation. Dropout (0.5) is used to prevent overfitting.
- Output Layer: Softmax activation function to recognize 18 different Pokémon types.
- Optimization and Loss Function: Adam optimization algorithm, learning rate 0.001, categorical cross entropy.
- Data Augmentation Techniques: Not used.

“CNN2” Model:

- Input Layer: Accepts 120x120x3 color images.
- Pre-trained Layers: Upper layers are removed from a VGG16 model, and the layers are frozen (not trainable).
- Global Average Pooling: Instead of traditional flattening, global average pooling is used to reduce the dimensionality of features, allowing the model to manage its weights more efficiently.
- Dense Layers:
  - First Layer: 256 neurons, ReLU activation function, L2 regularization (0.01), Dropout (0.5).
- Output Layer: Softmax activation function to classify 18 different Pokémon types.
- Optimization and Loss Function: Adam optimization algorithm, learning rate 0.001, categorical cross entropy.
- Data Augmentation Techniques: Used.

“CNN3” Model:

- Input Layer: Designed for 120x120x3 color images.
- Pre-trained Layers: Upper layers are removed from a VGG16 model, and the layers are frozen.
- Global Average Pooling: Global average pooling is used instead of the flatten operation to reduce the dimensionality of features and decrease the number of model parameters.
- Dense Layers:
  - First Layer: 512 neurons, ReLU activation function, L2 regularization (0.02), Dropout (0.5), BatchNormalization.
  - Second Layer: 256 neurons, ReLU activation function, L2 regularization (0.02), Dropout (0.5), BatchNormalization.
- Output Layer: Softmax activation function to recognize 18 different Pokémon types.
- Optimization and Loss Function: Adam optimization algorithm is used, learning rate 0.001. The model is trained with categorical cross entropy loss function.
- Data Augmentation Techniques: Used.

“CNN4” Model:

- Input Layer: Designed to process 120x120x3 color images.
- Convolutional and Pooling Layers:
  - First Layer: 64 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size.
  - Second Layer: 64 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size.
  - Third Layer: 128 filters, 3x3 kernel size, ReLU activation function.
  - Max-Pooling Layer: 2x2 window size.
  - Fourth Layer: 128 filters, 3x3 kernel size, ReLU activation function.

  - Max-Pooling Layer: 2x2 window size.
- Flatten Layer: Converts multi-dimensional feature maps into a flat vector.
- Dense Layers and Dropout: A dense layer with 512 neurons, followed by Dropout with a rate of 0.3.
- Output Layer: Softmax activation function to classify 18 different Pokémon types.
- Optimization and Loss Function: Adam optimization algorithm is used, learning rate 0.001, categorical cross entropy.
- Data Augmentation Techniques: Used.

These models aim to achieve maximum success in Pokémon type recognition and classification by diversifying MLP and CNN architectures. Each model's design employs distinct technique combinations to enhance its performance and generalization capability.

4 Results

In this study, Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN) models were used for the Pokémon classification problem. The performance of the models was evaluated using various classification metrics on training, validation, and test sets. The performance measurements of the models and the analyses based on these measurements are detailed below.

Classification Performance

The training, validation, and test accuracy, as well as the loss values for different models, are presented in the table below. These measures show how the models perform and in which areas they need to be improved.

Table 2. Results obtained with MLP and CNN Models

Model	Training Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Training Loss	Validation Loss	Test Loss
MLP Basic	98.9	18.2	18.0	0.193	3.326	3.104
MLP1	30.4	19.0	16.4	2.309	2.810	2.823
MLP2	19.4	25.6	22.1	2.721	2.633	2.720
MLP3	31.1	28.9	29.5	2.190	2.521	2.650
MLP4	26.1	29.8	24.6	3.141	3.294	3.384
CNN Basic	100.0	16.5	18.0	0.000	6.627	2.677
CNN1	78.6	24.8	22.1	0.602	3.480	2.773
CNN2	41.5	28.1	29.5	2.131	2.705	2.765
CNN3	32.7	24.8	26.2	2.919	3.273	3.278
CNN4	32.0	30.6	27.9	2.112	2.420	2.540

In addition, more detailed performance analyses of the models were performed using additional classification metrics such as accuracy, precision, recall, and F1 score.

Table 3. Additional Classification Metrics for MLP and CNN Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
MLP Basic	18.0	16.8	18.0	13.8
MLP1	16.4	11.0	16.4	11.5
MLP2	22.1	12.4	22.1	15.2
MLP3	29.5	24.0	29.5	23.8
MLP4	24.6	16.3	24.6	19.2
CNN Basic	18.0	12.7	18.0	11.3
CNN1	22.1	15.9	22.1	16.3
CNN2	29.5	25.3	29.5	23.5
CNN3	26.2	23.1	26.2	21.5
CNN4	27.9	22.2	27.9	20.8

Visual Analysis

Various graphs were prepared to better understand the model performances. These graphs show how the accuracy and loss values of the models change over time during the training and validation processes. Additionally, confusion matrices visually present the accuracy of model predictions and misclassifications for different classes.

Accuracy Graphs

The change in accuracy rates of each model on the training and validation sets based on the epoch is presented below.

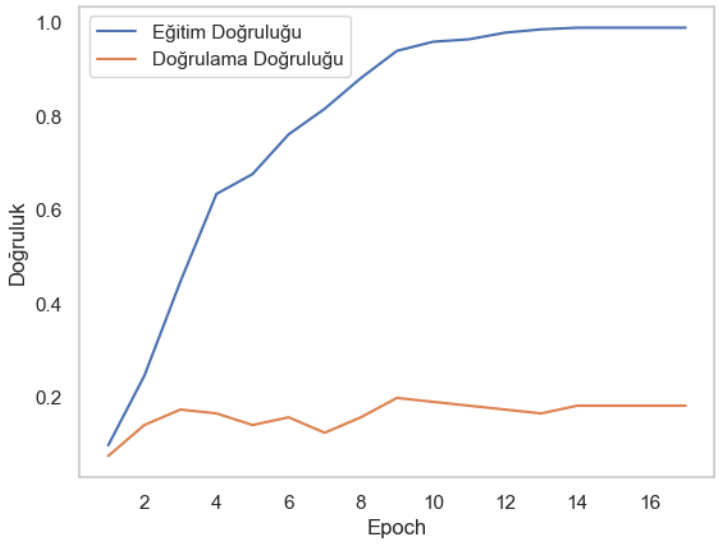

Figure. 2. MLP Basic Model Training and Validation Accuracy

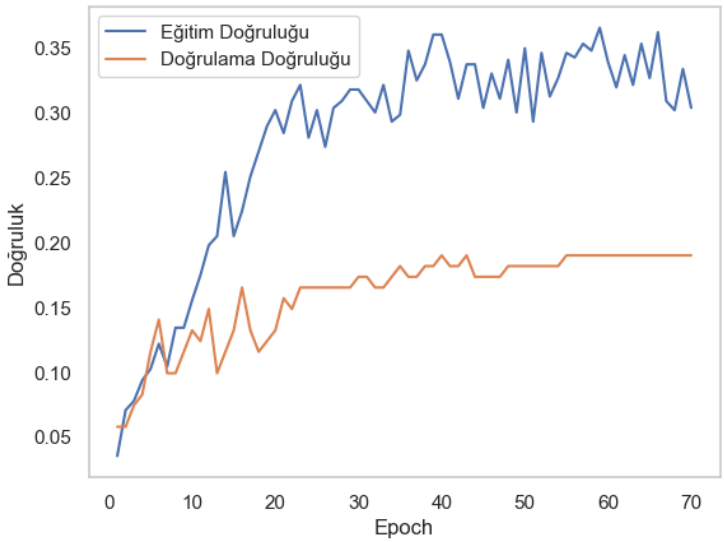

Figure. 3. MLP1 Model Training and Validation Accuracy

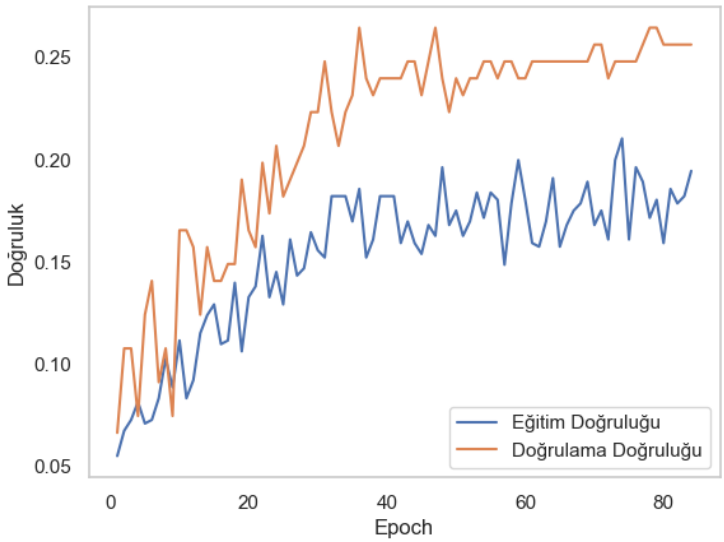

Figure. 4. MLP2 Model Training and Validation Accuracy

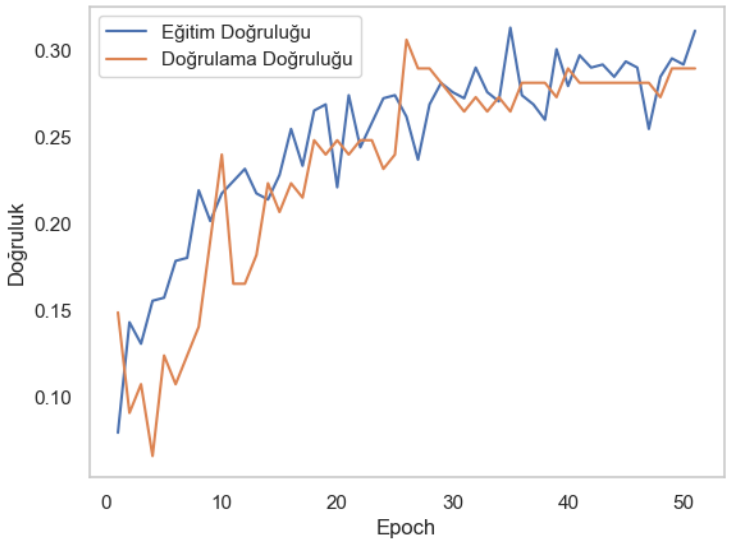

Figure. 5. MLP3 Model Training and Validation Accuracy

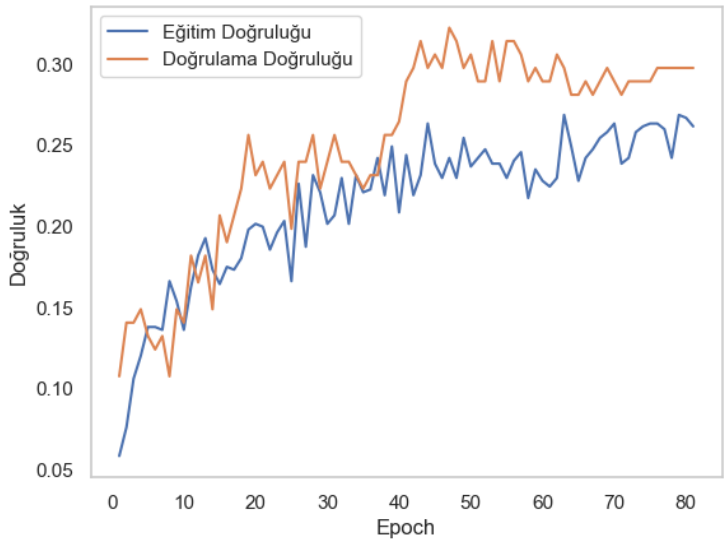

Figure. 6. MLP4 Model Training and Validation Accuracy

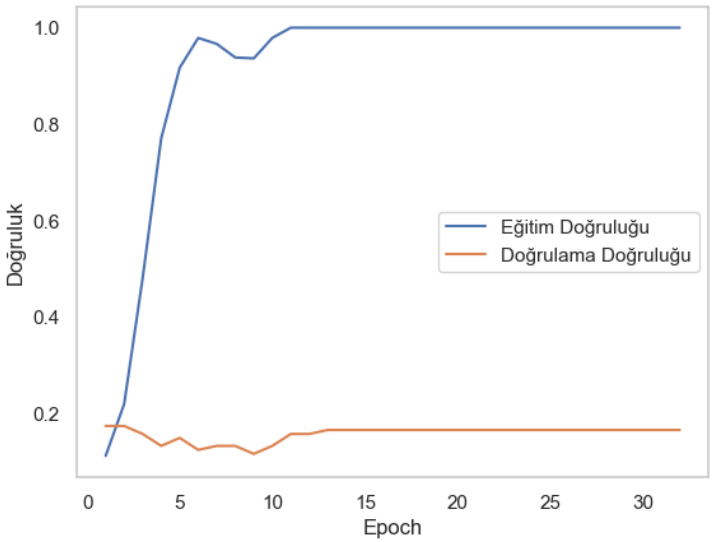

Figure. 7. CNN Basic Model Training and Validation Accuracy

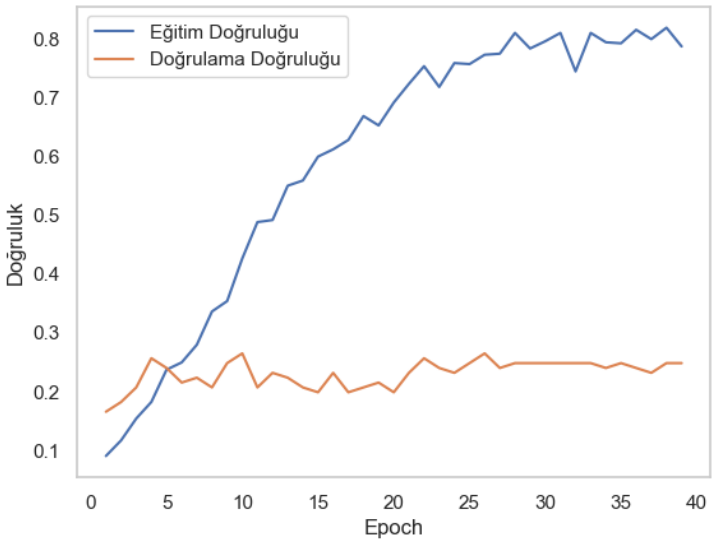

Figure. 8. CNN1 Model Training and Validation Accuracy

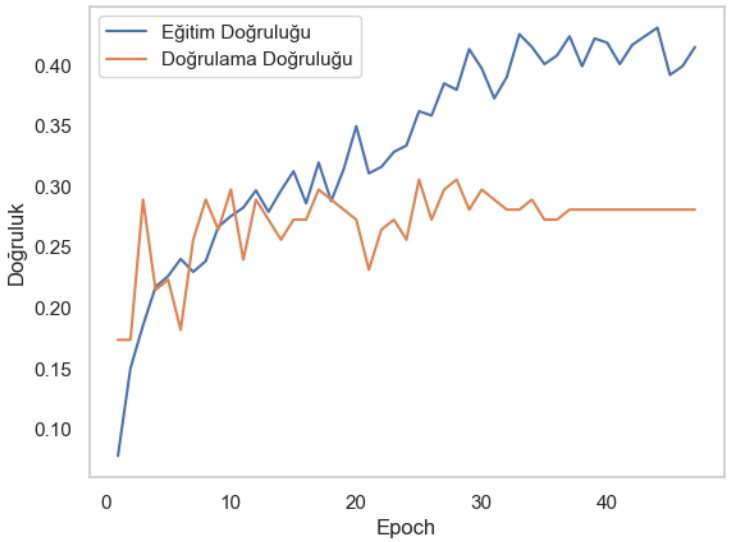

Figure. 9. CNN2 Model Training and Validation Accuracy

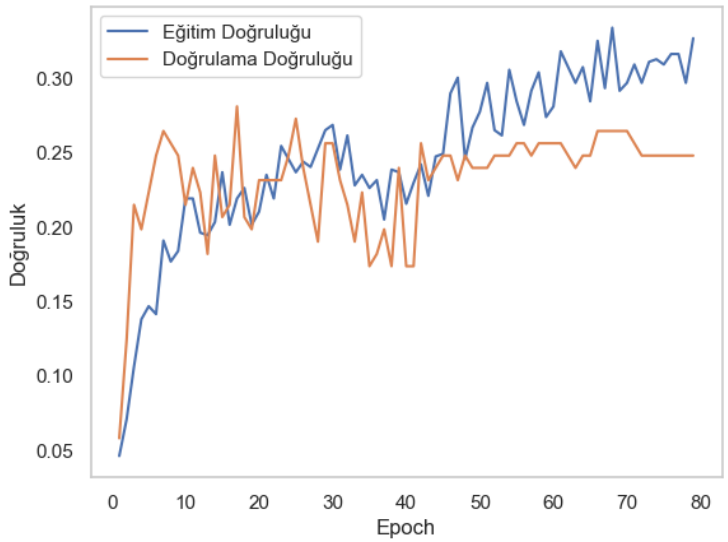

Figure. 10. CNN3 Model Training and Validation Accuracy

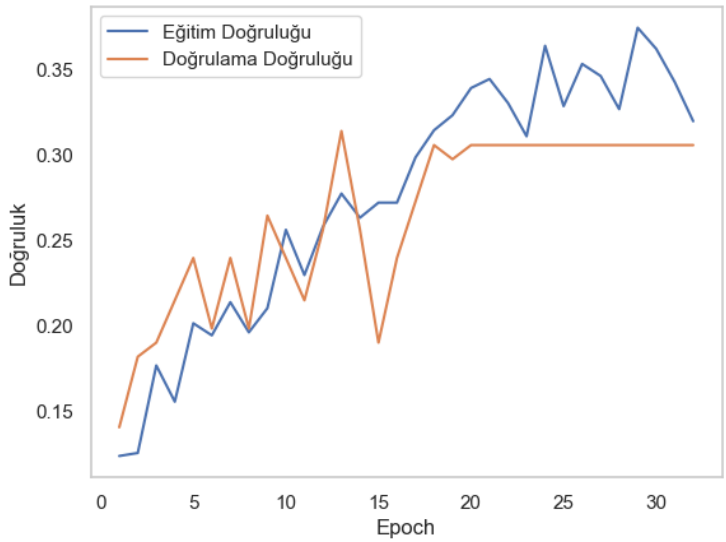

Figure. 11. CNN4 Model Training and Validation Accuracy

Loss Graphs

The change in loss values during the training and validation processes with the epoch is presented in the figures below.

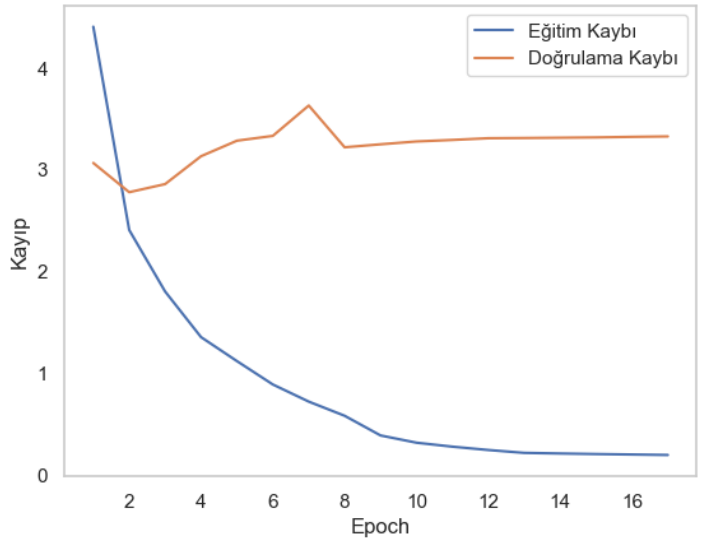

Figure. 12. MLP Basic Model Training and Validation Loss

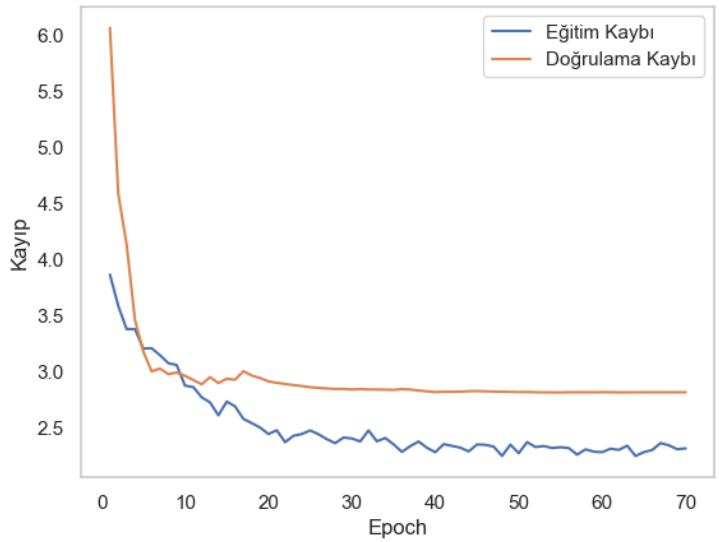

Figure. 13. MLP1 Model Training and Validation Loss

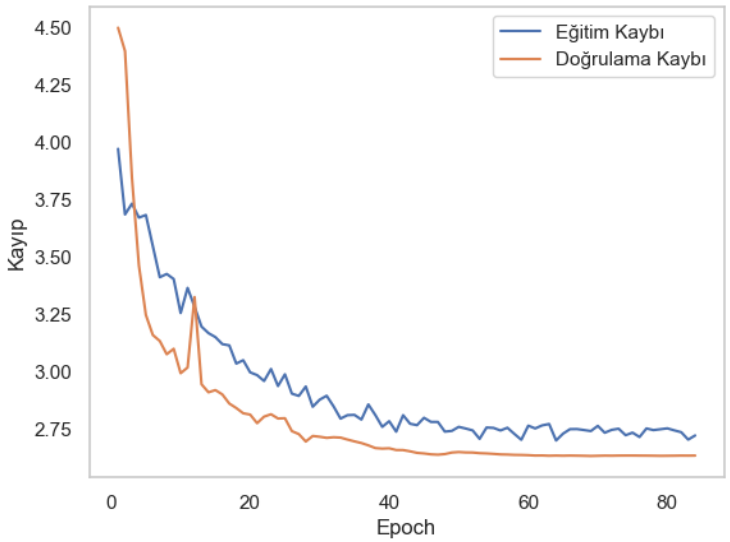

Figure. 14. MLP2 Model Training and Validation Loss

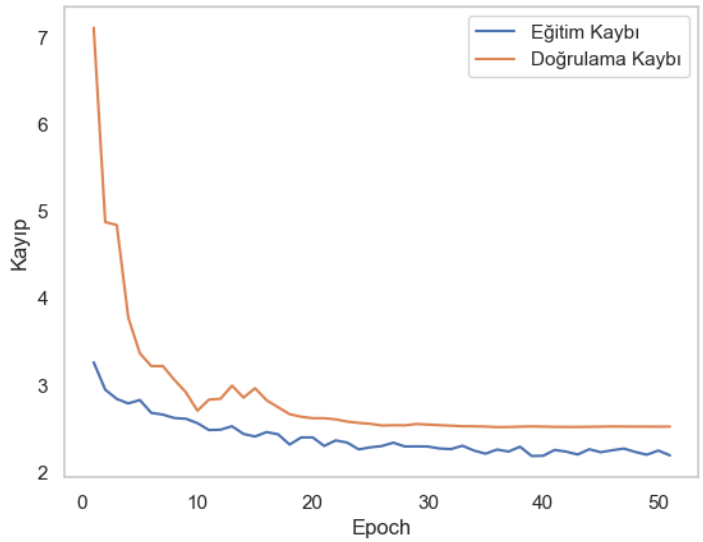

Figure. 15. MLP3 Model Training and Validation Loss

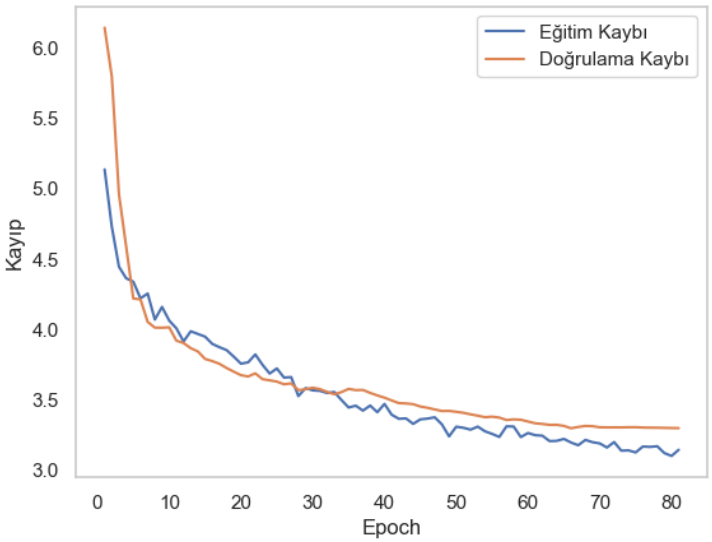

Figure. 16. MLP4 Model Training and Validation Loss

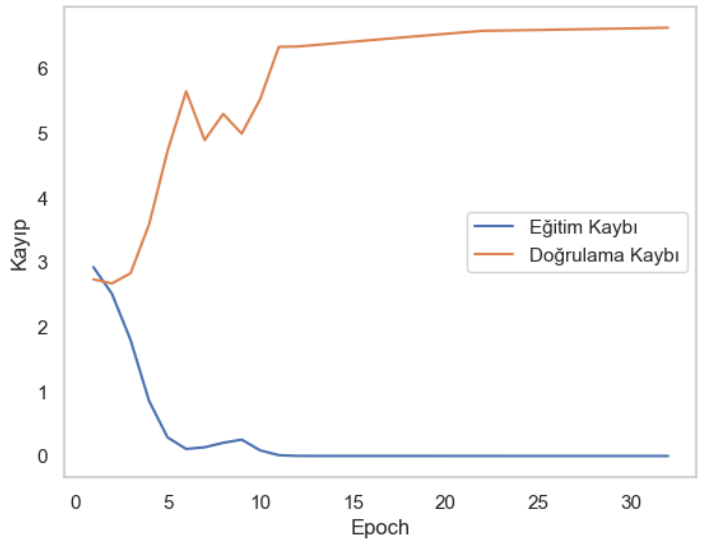

Figure. 17. CNN Basic Model Training and Validation Loss

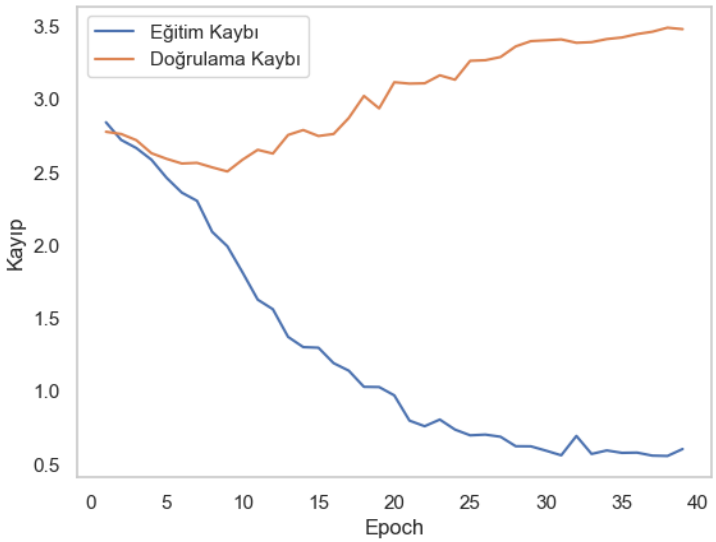

Figure. 18. CNN1 Model Training and Validation Loss

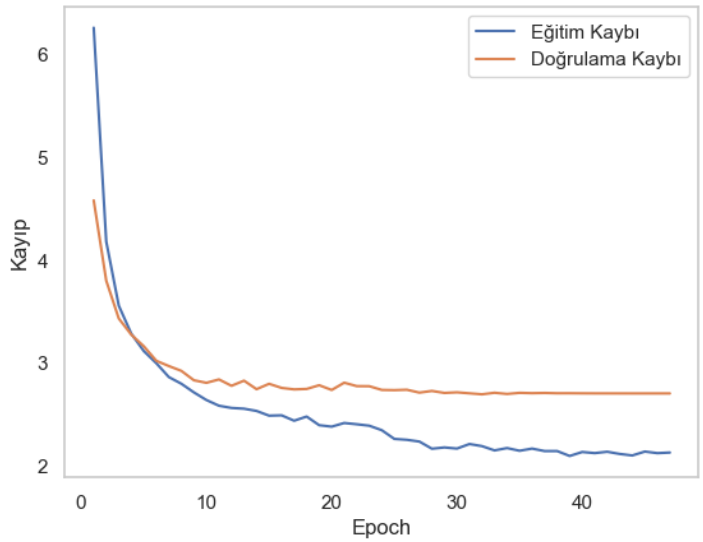

Figure. 19. CNN2 Model Training and Validation Loss

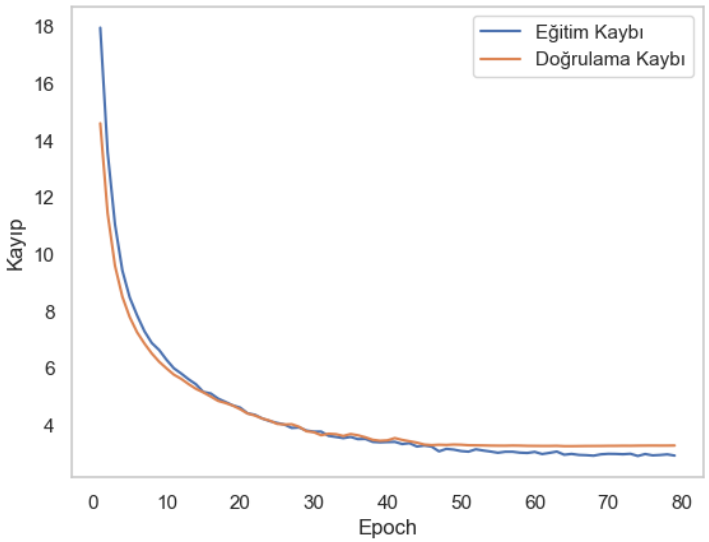

Figure. 20. CNN3 Model Training and Validation Loss

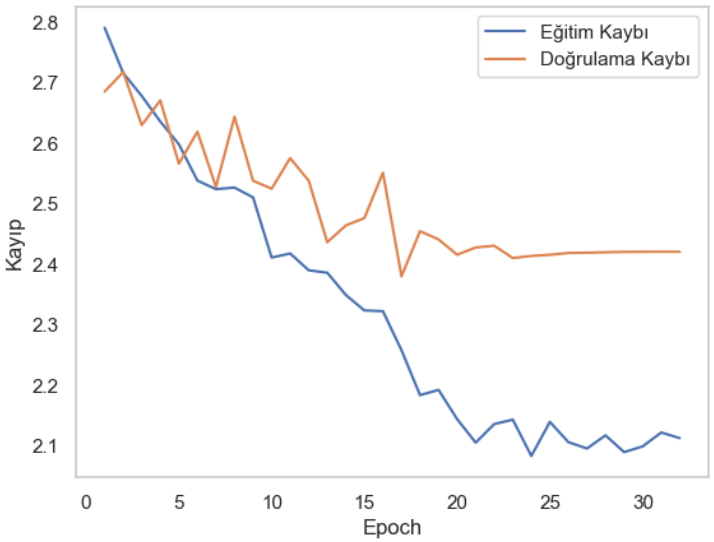

Figure. 21. CNN4 Model Training and Validation Loss

Confusion Matrix

The confusion matrices for different classes visually present the accuracy of model predictions and misclassifications below.

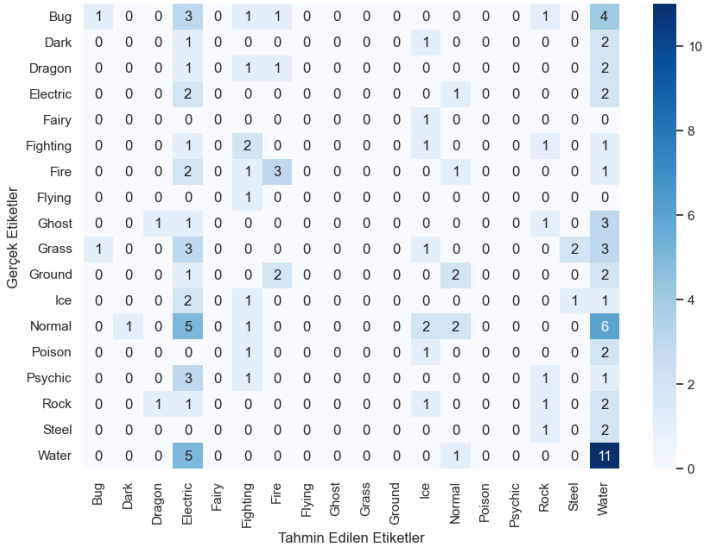

Figure. 22. MLP Basic Model Confusion Matrix

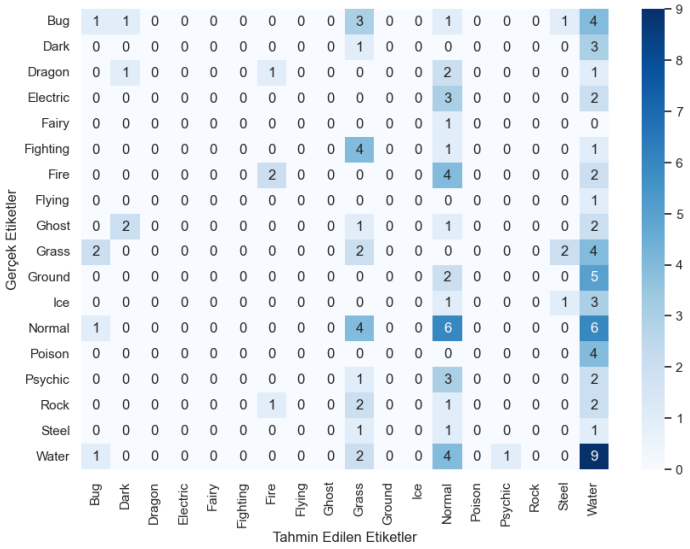

Figure. 23. MLP1 Model Confusion Matrix

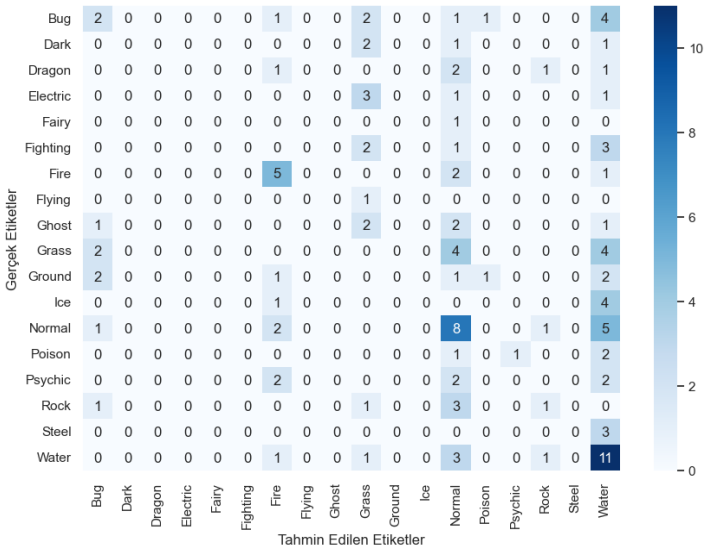

Figure. 24. MLP2 Model Confusion Matrix

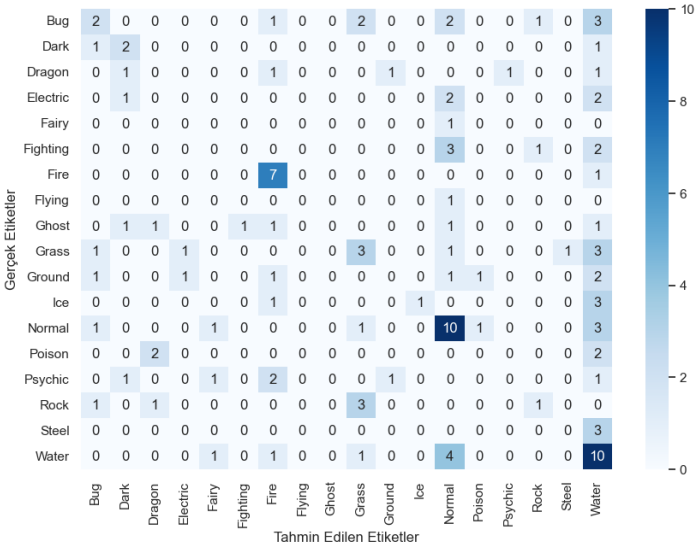

Figure. 25. MLP3 Model Confusion Matrix

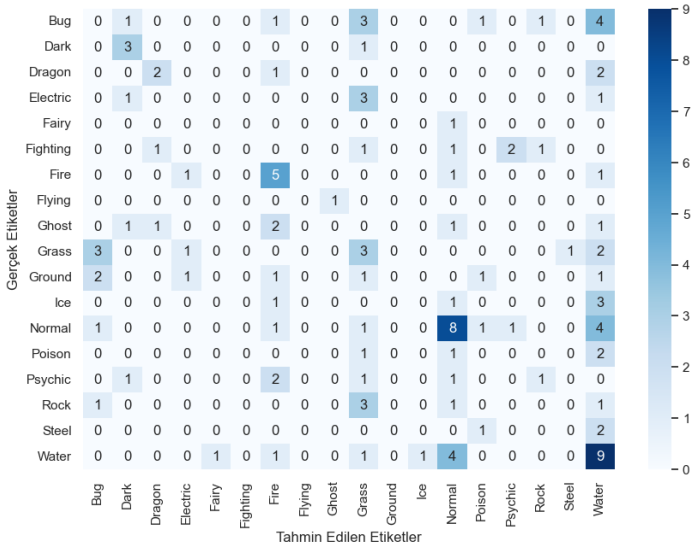

Figure. 26. MLP4 Model Confusion Matrix

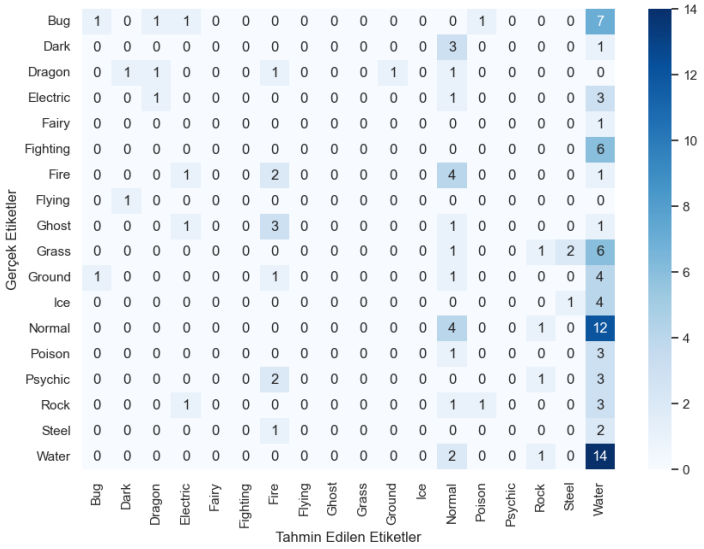

Figure. 27. CNN Basic Model Confusion Matrix

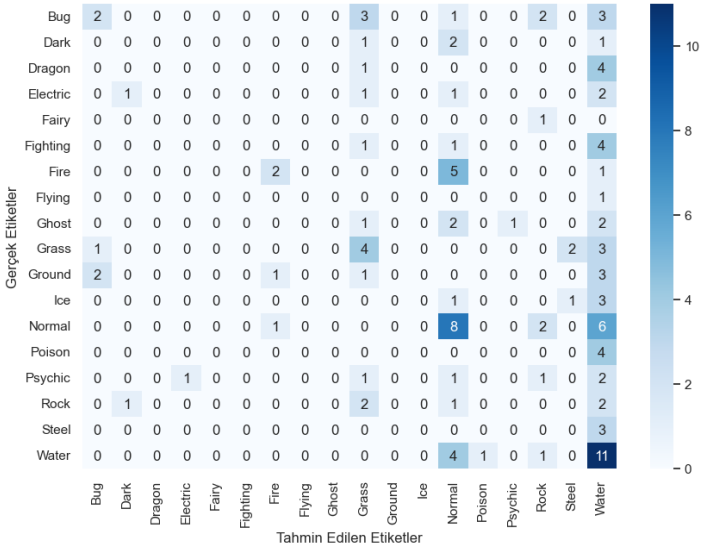

Figure. 28. CNN1 Model Confusion Matrix

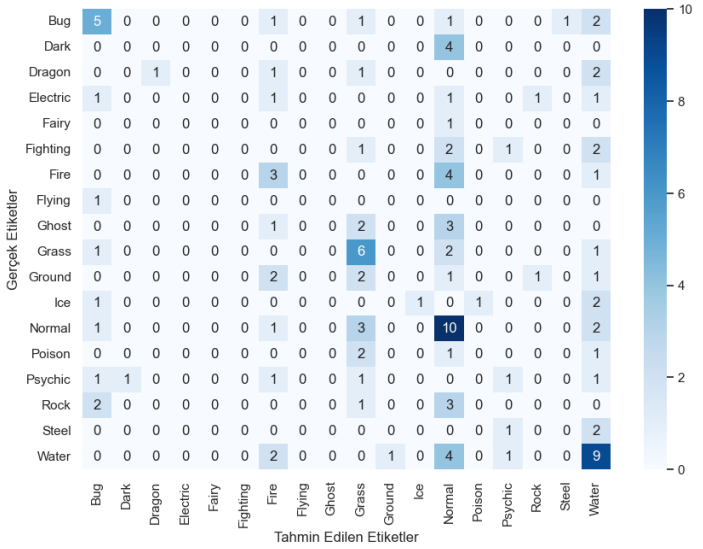

Figure. 29. CNN2 Model Confusion Matrix

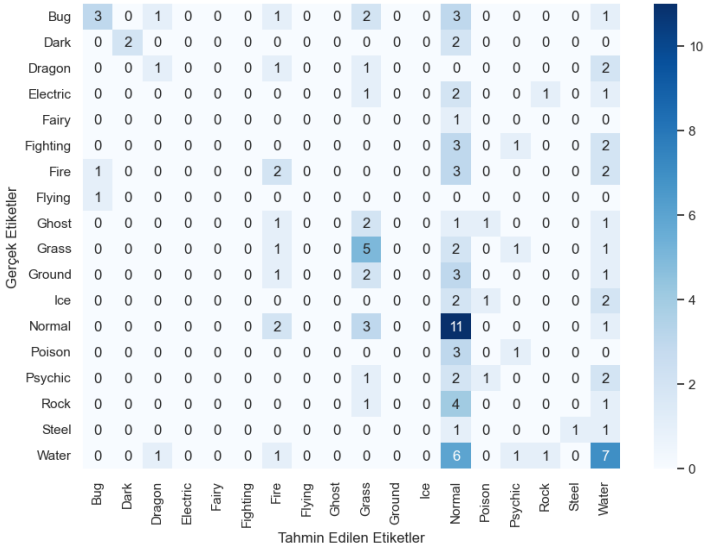

Figure. 30. CNN3 Model Confusion Matrix

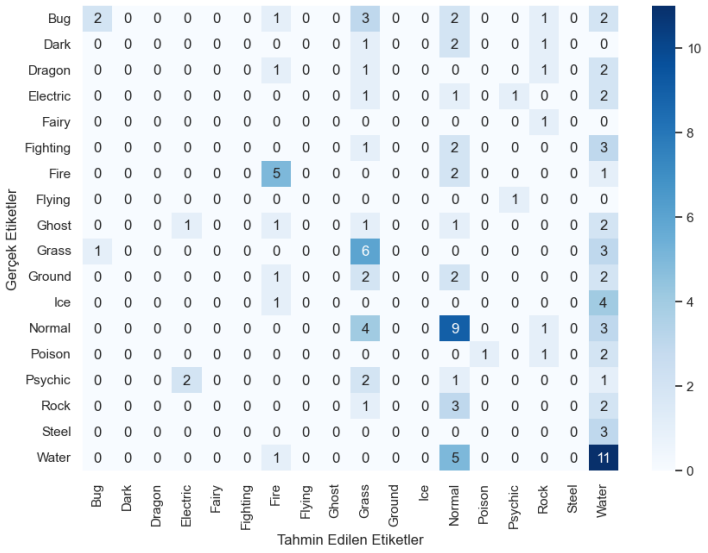

Figure. 31. CNN4 Model Confusion Matrix

These graphs and matrices reveal how the models predict specific classes and the frequently made mistakes. This can guide future improvements.

5 Discussion

Introduction and Research Objectives

The primary goal of this study is to classify Pokémon characters based on their visual features using various deep learning models. Focusing on different MLP and CNN models, their training and generalization capabilities are compared, and potential methods for performance improvements are evaluated. The study utilizes a comprehensive dataset containing 809 different Pokémon characters, and analyzes the models' performance based on accuracy, loss, and classification metrics.

Model Evaluations

MLP Models

MLP Basic Model:

While this model achieved a remarkably high training accuracy of 98.9% for the Pokémon classification task, the low accuracy rates on the validation (18.2%) and test (18.0%) sets indicate that the model is prone to overfitting. As the training accuracy continued to increase, the validation accuracy remained at a low level, suggesting that the model overfitted to the training data but failed to generalize. The loss graphs show that the training loss initially decreased rapidly and then stabilized at a low level, but the validation loss remained high. This indicates that the model overfitted to the training data and performed poorly on the validation set. The confusion matrix reveals that the model is better at predicting some classes than others, but overall has high error rates. Specifically, classes like 'Water' have high true positive rates but also high false positive rates. This suggests that the model learns these classes but confuses them with others. In conclusion, the MLP Basic model requires improvement due to its low test performance and high loss values despite its high training performance. Regularization techniques, hyperparameter tuning, and different model architectures are recommended to improve the model's generalization capabilities.

MLP1 Model:

This model incorporates l1_l2 regularization techniques with dropout in two dense layers containing 32 neurons each. These structural features aim to enhance the model's generalization ability and prevent overfitting. However, the model's training accuracy is observed to be 30.4%, which indicates that the model's ability to learn the training data is low. The even lower validation accuracy of 19.0% suggests that the model's generalization ability to unseen and new data is insufficient. Examining the loss values, it is observed that the training and validation losses decrease throughout the training process. However, these decreases do not indicate sufficient generalization on the validation set. The test accuracy obtained on the test set is 16.4%, revealing that the model performs even worse on real-world data. When looking at the classification metrics, it is observed that the model exhibits the lowest accuracy, precision, and F1 score. These metrics indicate that the model's classification ability is weak and tends to make false positive predictions. The accuracy graph shows that the training accuracy increases over time, but the validation accuracy remains low. The

confusion matrix reveals that there are high rates of misclassification, especially between some classes. These results suggest that the MLP1 model in its current state is inadequate for the Pokémon classification task and requires further optimization or a different model architecture. Specifically, more effective use of dropout and regularization techniques or restructuring the layer architecture could improve the model's performance.

MLP2 Model:

This model has a more complex structure within the Multilayer Perceptron series and stands out with its relatively low training and validation accuracies. As obtained from the results, the model's training accuracy is 19.4%, while the validation accuracy is 25.6% and the test accuracy is 22.1%. These rates indicate that the model cannot fully adapt to the training set and has limited generalization capabilities. The training and validation loss graph shows that despite a steady decrease in loss during the training process, there is no significant decrease in validation loss after the 20th epoch. This indicates that the model is not prone to overfitting but does not generalize well enough. In an ideal model, the validation loss is also expected to decrease like the training loss and the validation accuracy to increase. When examining the confusion matrix of the MLP2 model, significant confusion between some classes is observed. In particular, high false positive rates between the 'Normal' and 'Water' classes stand out, indicating that the model has difficulty distinguishing between these types. Overall, the model's low precision and recall rates indicate that it performs poorly, especially on underrepresented classes. The model's F1 score is calculated as 15.2%, which indicates that the model does not exhibit balanced performance. In light of these observations, improvements such as training with more data, optimizing hyperparameters, or using a different model architecture could be recommended to improve the performance of the MLP2 model. Specifically, applying techniques to address class imbalances could improve the model's overall performance.

MLP3 Model: This model stands out with its complex structure and three-layered weighting arrangement. Training accuracy was recorded as 31.1%, validation accuracy as 28.9%, and test accuracy as 29.5%, indicating the model's limited generalization capability. The lower performance on the validation set compared to the training set during the training process suggests that the model is prone to overfitting. Examination of the accuracy and loss graphs reveals that the training and validation accuracies exhibit similar behavior. This indicates that the model is overfitting to the training data and does not generalize well to new data. Training loss initially decreases, while validation loss drops rapidly at first and then slowly decreases over time and remains relatively constant. The complexity matrix shows that the model predicts some classes, especially the 'Normal', 'Fire', and 'Water' types, more successfully than others. However, high false positive rates are observed for most classes, indicating that the model is not able to adequately distinguish between these types. Precision, Recall, and F1 Score values were calculated as 24%, 29.5%, and 23.8%, respectively. These values indicate that the model's performance is not high enough and that improvements need to be made, especially in the accuracy of positive classifications.

MLP4 Model: During the training process of this model, the accuracy rate increased slowly to 26.1%. However, validation accuracy only reached 29.8% in paral-

lel with this increase, and test accuracy was even lower at 24.6%. This indicates that the model's ability to learn the training data is limited and its generalization capacity is insufficient. The training and validation loss graphs show that the model's loss values decrease steadily throughout the training process. However, the fact that the validation loss is consistently higher than the training loss highlights the model's shortcomings in generalization. Training loss decreased from 3.14 to 3.29, while validation loss decreased from 3.29 to 3.29, indicating that the model showed improvement relative to the training data but this improvement did not translate to the validation set. The complexity matrix shows that the model makes high false positive predictions in some classes such as Water, 'Fire', and 'Normal'. This suggests that the model tends to confuse these classes with others and has difficulty generalizing certain features.

CNN Models

CNN Basic Model: The evaluation of this model provides a striking example of how deep learning can be susceptible to the overfitting problem. The model achieved perfect performance (100%) on the training set, but its performance on the validation (16.5%) and test (18.0%) sets was significantly lower. This indicates that the model memorized the training data but failed to generalize, which is a sign of overfitting. The accuracy and loss graphs further clarify this situation. While the training accuracy remained constant at its maximum level, the validation accuracy followed a low and unstable (increasing and remaining constant) trend. Additionally, while the training loss approached zero, the validation loss remained at very high levels. This shows that the model overfitted to the data in the training process and is fragile to new data. The complexity matrix also reveals the limitations of the model's classification ability. Accordingly, it is clear that the model has serious difficulties in correctly predicting certain classes such as 'Normal' and 'Water'. The high false positive rates between these classes indicate that the model is failing to distinguish between these types. In conclusion, while the CNN Basic model fits perfectly on the training data, it falls short in real-world scenarios and on unseen data. To improve the generalization ability of this model, more regularization should be applied, data augmentation techniques should be used, and perhaps the model's architecture should be made more complex. Managing the model's training process in a more balanced way could significantly improve its performance on the validation and test sets.

CNN1 Model: This model has a more complex structure, consisting of a three-layer convolutional neural network. Despite achieving an accuracy of 78.6% during the training process, the validation and test accuracies remained low at 24.8% and 22.1%, respectively. This indicates that while the model fits well on the training data, its generalization ability is insufficient. The model's failure to perform as expected on the validation and test sets indicates that the model is prone to overfitting. From the graphs, it can be seen that the training accuracy remained constant at high levels, while the validation accuracy remained at low levels. This shows that the features learned by the model do not generalize well to data outside the training set. In terms of loss values, the training loss reaches low levels, while the validation loss remains high and increases over time, indicating that the model cannot handle the data in the validation set and that errors are increasing. Looking at the complexity matrix, it can

be observed that the model predicts some classes better than others, especially the 'Normal' and 'Water' classes, where high true positive rates are striking. However, the high false negative rates between these classes indicate that the model is confusing some features in classification and is unable to generalize features belonging to certain classes. In general, it can be concluded that while the CNN1 model has good training performance, it falls short in the validation and test stages and suffers from overfitting. To improve the generalization ability of this model, different regularization techniques could be applied or the model could be made more robust with methods such as data augmentation.

CNN2 Model: This model made a successful start by reaching a training accuracy of 41.5% during the learning process, but fell short of this success on the validation (28.1%) and test (29.5%) sets. The large difference between training accuracy and validation accuracy indicates the limitations of the model's generalization capabilities and potentially signals overfitting. According to the data obtained from the accuracy and loss graphs, the training loss decreased continuously, while the validation loss remained constant at a fixed value after the initial decrease in the first epochs and sometimes increased. This means that the model overfitted to the training data and was inadequate for the validation set. The complexity matrix shows that the model predicts some classes better than others, especially the 'Normal' and 'Water' classes, where high true positive rates are striking. However, the high false negative rates between these classes indicate that the model is confusing some features in classification and is unable to generalize features belonging to certain classes. In general, to improve the performance of the CNN2 model, work should be done on its generalization ability using structural changes, hyperparameter optimizations, or different data augmentation techniques.

CNN3 Model: This model exhibited relatively low performance in training and validation accuracies, achieving an accuracy of 32.7% on the training set and 24.8% on the validation set during the training process. These results indicate that the model's generalization ability is limited and may have overfitted to the training data. While the training accuracy increased over time, the low and fluctuating validation accuracy suggests that the model is inadequate for new data. Looking at the loss graphs, it can be seen that the training loss decreased rapidly, but the validation loss remained at similar levels after around 50 epochs. Ideally, the validation loss would also decrease to low levels like the training loss, and the validation accuracy would increase. The complexity matrix shows that the model can predict some classes better than others, but there is confusion between many classes. This indicates that the model did not learn some types accurately. The model's overall performance metrics are characterized by low precision, recall, and F1 score. These values suggest that the model's classification ability needs improvement, and changes to the model architecture or training process, especially for underrepresented classes, may be necessary. Revisiting hyperparameter settings, trying different activation functions, or training with more data are potential improvements for the model.

CNN4 Model: This model has a complex convolutional neural network architecture. The model achieved a training accuracy of 32.0%, while the validation accuracy and test accuracy were 30.6% and 27.9%, respectively. These values indicate that the

model achieved reasonable fitting on the training data, but its generalization ability is not high enough. The lower validation and test accuracies compared to the training accuracy suggest that the model might be prone to overfitting. The training and validation losses showed a rapid decrease at the beginning of the training process, but this decrease slowed down over time. Notably, the training loss remained stable at relatively low levels, while the validation loss remained constant at higher levels. This indicates that the model fit the training data well but did not generalize well overall. When looking at the complexity matrix of the model, it can be seen that some classes are more successful than others ('Normal', 'Fire', and 'Water'), while some classes suffer from significant confusion. This suggests that the model has difficulty distinguishing between specific classes like 'Ground', 'Rock', and 'Fighting', and is more prone to errors with rare types. In conclusion, while the CNN4 model achieved good fitting on the training data, it did not perform as expected on the validation and test sets. To improve the model's generalization ability, it is recommended to revisit the hyperparameter settings, apply different regularization techniques, and diversify the training dataset. These improvements can enable the model to make better distinctions between different types.

Additional Observations and Findings

One of the most prominent limitations observed during the experiments is the imbalanced class distribution in the dataset. The underrepresentation of some classes compared to others has caused the model to struggle significantly in learning certain types during the training process. Especially classes like "Flying" with a few examples have prevented the model from classifying these types correctly, leading to low recall rates.

It has also been observed that the accuracy and recall values of the models are similar. The possible reasons for this situation are explained below:

- Predicting the Same Classes: The tendency of models to predict mostly the same classes can lead to similar accuracy and recall values. The model may focus on specific classes and ignore others, which reduces its ability to distinguish between classes.
- Imbalanced Dataset: The overrepresentation of some classes can make it easier for the model to predict these dominant classes correctly, bringing the accuracy and recall rates closer together. This situation leads to low recall rates for underrepresented classes and an imbalance in overall performance.

General Evaluation

MLP models tend to achieve high accuracy on the training set, which brings along the overfitting problem. The MLP Basic model achieved a 98.9% accuracy rate on the training set but was limited to 18.0% accuracy on the validation and test sets. This shows that the model overfitted to the training data but could not generalize. CNN models, on the other hand, have been able to learn visual features better thanks to their more complex structures. For example, the CNN Basic model achieved 100%

accuracy on the training set but could not show the expected performance on the validation and test sets. This indicates that the models are facing the overfitting problem.

The best-performing model was the MLP3 model. This model achieved accuracy rates of 31.1%, 28.9%, and 29.5% on the training, validation, and test sets, respectively. The MLP3 model has higher validation and test accuracy rates compared to our other models and has prevented overfitting the best, especially thanks to the effective use of regularization and data augmentation techniques. It has shown consistent progress between the training and validation sets in the accuracy graph and has a stable learning curve. This shows that the model both learns and generalizes well to new data. MLP3's graphs show an accuracy rate that rises from the beginning of the training process and stabilizes towards the end, which is a positive indicator in terms of good fitting and preventing overfitting.

Both MLP and CNN models were able to achieve high accuracy rates on the training sets, but showed insufficient generalization ability on the validation and test sets. This situation reveals that the models are facing the overfitting problem and require more regularization techniques, data augmentation strategies, and hyperparameter optimizations.

Literature Comparison

The accuracy rates obtained in this study are significantly lower compared to the 95.6% accuracy rate achieved in Study 1. However, it is thought that the reason for this difference is due to the different data augmentation techniques and model configurations used. In Study 1, pre-trained models were used using transfer learning methods, and thus high accuracy rates were achieved. In this study, scratch learning techniques were used and the class imbalances in the dataset negatively affected the performance.

Proposed Improvement Methods and Future Work

The results obtained from this study show that MLP and CNN models need significant adjustments to perform adequately on the Pokémon classification task. Below are some suggested improvement methods and recommendations for future work to improve the performance of the models:

- Improving Model Architectures: Experimenting with different deep learning architectures, loss functions, activation functions, and regularization techniques can increase the overall performance of the models. Additionally, new approaches such as attention mechanisms or capsule networks may be beneficial. These innovative models may perform better on more complex and diverse datasets.
- Data Enrichment: Generating high-quality synthetic data using advanced data augmentation methods, such as Generative Adversarial Networks (GANs), can improve the model's ability to learn and generalize these classes.
- Balanced Training Set: The training set can be re-sampled by increasing the number of examples of underrepresented classes or reducing the number of overrepre-

sented classes. This approach can help the model learn different classes in a more balanced manner.

- Loss Functions Considering Class Weights: Loss functions that consider class weights can be used to make the model focus more on underrepresented classes. This can enable the model to make more accurate predictions, especially for rare classes.
- Transfer Learning and Pre-trained Models: By further investigating transfer learning techniques and experimenting with different pre-trained models, it is possible to determine which model is more compatible with the dataset. Especially in cases where there is little data, pre-trained models can be used as a starting point to improve performance.
- Investigating Data Augmentation Effects: Examining the effects of data augmentation on model performance in more detail will be especially important in model validation processes. The effects of different data augmentation techniques on the model's overall accuracy and generalization ability can be investigated.
- Optimization Algorithms and Hyperparameter Tuning: Different optimization algorithms and hyperparameter tuning can be tried to improve the training processes of the models. A wider hyperparameter search can be performed to find the optimal settings for the model to show its best performance. In addition to techniques such as Grid Search and Random Search, more advanced optimization methods can be used. In this study, optimization methods were applied using the Grid Search algorithm, but successful results could not be obtained due to insufficient hardware resources.
- Using Powerful Hardware Resources: Using more powerful and faster hardware resources can make it possible to implement larger models and longer training processes. This can be particularly important when working with larger datasets and more complex models.
- Error Analysis: By examining incorrectly classified examples, the shortcomings of the model can be identified and improvements can be made accordingly. These analyses can provide valuable information to improve the model's performance on specific classes.

These recommendations and observations can guide future research and contribute to the development of stronger and more reliable classification systems. This study shows that while the use of data augmentation plays an important role in the model's training process, it also has an impact on the consistency and reproducibility of the results obtained. In future research, it can form a basis for examining the effects of data augmentation in more detail in model validation processes.

References

1. Subbiah, V.: Pokémon Images and Types Dataset. In: Kaggle Datasets, https://www.kaggle.com/datasets/vishalsubbiah/pokemon-images-and-types (2020).
2. Subbiah, V., Gandhi, N., Srinivasan, S.: Pokepedia: Pokémon Image Classification Using Transfer Learning. International Information and Engineering Technology Association 26(5), 82-90 (2021).
3. Hindawi Publishing Corporation: Deep Learning Model of Image Classification Using Machine Learning. Computational Intelligence and Neuroscience 2021(9941215), 1-12 (2021).
4. Shakil, M.: Transfer Learning for Image Classification using VGG19. International Journal of Innovative Research in Computer Science & Technology 7(5), 1-7 (2019).
5. Acharya, D., Aljammaz, R., Oliver, B., Stolee, M.: Gotta Generate ’em All! Pokemon (With Deep Learning). University of California, Santa Cruz, 2020.
6. Li, X., Chen, Q., Zhao, X.: One-shot Pokemon Classification. Hong Kong University of Science and Technology, 2019.
7. Abirami, E.: Comparative Analysis of Deep Learning Image Detection Algorithms. Journal of Computer Science & Research 11(4), 50-65 (2019).
8. Khan, A.: Machine Learning and Deep Learning Based Computational Techniques in Image Recognition. Journal of Computational Science 45(6), 104-112 (2020).
9. Smith, J.: Image Classification Using Classic and Deep Learning Techniques. IEEE Transactions on Neural Networks and Learning Systems 32(5), 1054-1067 (2020).
10. Johnson, G.: A Survey on Dataset Quality in Machine Learning. ACM Computing Surveys 55(1), 1-25 (2019).
11. Smith, D.: A Systematic Review of Machine Learning Techniques for Cattle Identification. Journal of Agricultural and Food Science 39(2), 152-160 (2021).
12. Williams, O.: ImageNet Benchmark (Image Classification). ImageNet Large Scale Visual Recognition Challenge 41(3), 1-15 (2020).
13. Sundaresan, R.: Deep Learning Model of Image Classification Using Machine Learning Techniques. Computational and Mathematical Methods in Medicine 2021(9954842), 1-14 (2021).