Pokémon Classification with Deep Learning Applications

Hacettepe University, Institute of Informatics, Turkiye

VBM 689 - Applications of Deep Learning

Mert Caliskan

June 2nd, 2024

Introduction

Project Objective:

- To classify Pokémon characters based on their visual features.

Dataset Used:

- The "Pokemon Image Dataset" from Kaggle, containing 809 different Pokémon characters.

Methods Used:

- Multilayer Perceptrons (MLP)
- Convolutional Neural Networks (CNN)

Related Work

Highlighted Work:

- **Article:** Pokepedia: Pokémon Image Classification Using Transfer Learning
- **Authors:** Vishal Subbiah, Nivedhitha Gandhi, Sruti Srinivasan
- **Publication:** International Information and Engineering Technology Association, Vol. 26, No. 5, 2021.
- **Models Used:** VGG16, VGG19, ResNet101, MobileNetV2, DenseNet201, EfficientNetB7, EfficientNetV2L
- **Highest Performance:** 95.6% accuracy with the ResNet101 model

Methodology
Methodology and Flow Diagram (1/2)

1. Data Loading and Preprocessing

- Dataset: Pokémon characters are loaded using the 'Pokemon Image Dataset' from Kaggle.
- Preprocessing: Missing data is checked, unnecessary columns ('Type2', 'Evolution') are removed.
- Image Processing: Images are resized to 120x120 pixels and pixel values are normalized to the range of 0-1.
- File Paths: Image file paths are created using Pokémon names.

2. Data Augmentation

- Techniques: Random rotation, translation, shearing, zooming, and horizontal flipping are applied to the images in the training set.
- Model Application: Data augmentation techniques were used in MLP2, MLP3, MLP4, CNN2, CNN3, and CNN4 models.

3. Feature Extraction and Model Configuration

- Feature Extraction: Features are extracted from the images based on pixel values.
- Models: Multilayer Perceptrons (MLP) and Convolutional Neural Networks (CNN) models are designed and created.

Methodology
Methodology and Flow Diagram (2/2)

4. Model Training and Evaluation

- Training: Models are trained on the specified training and validation sets.
- Optimization: Models are optimized using various optimization algorithms and loss functions.
- Supporting Techniques: The training process is supported by methods such as early stopping and learning rate decay.
- Performance Evaluation: Models' performance is evaluated using metrics such as accuracy, precision, recall, and F1 score. Training and post-training performances are visualized using confusion matrices, loss and accuracy graphs.

5. Analysis and Discussion of Results

- Performance Analysis: Models' performance results are analyzed in detail.
- Recommendations: Recommendations are provided for future work.

Methodology
Dataset

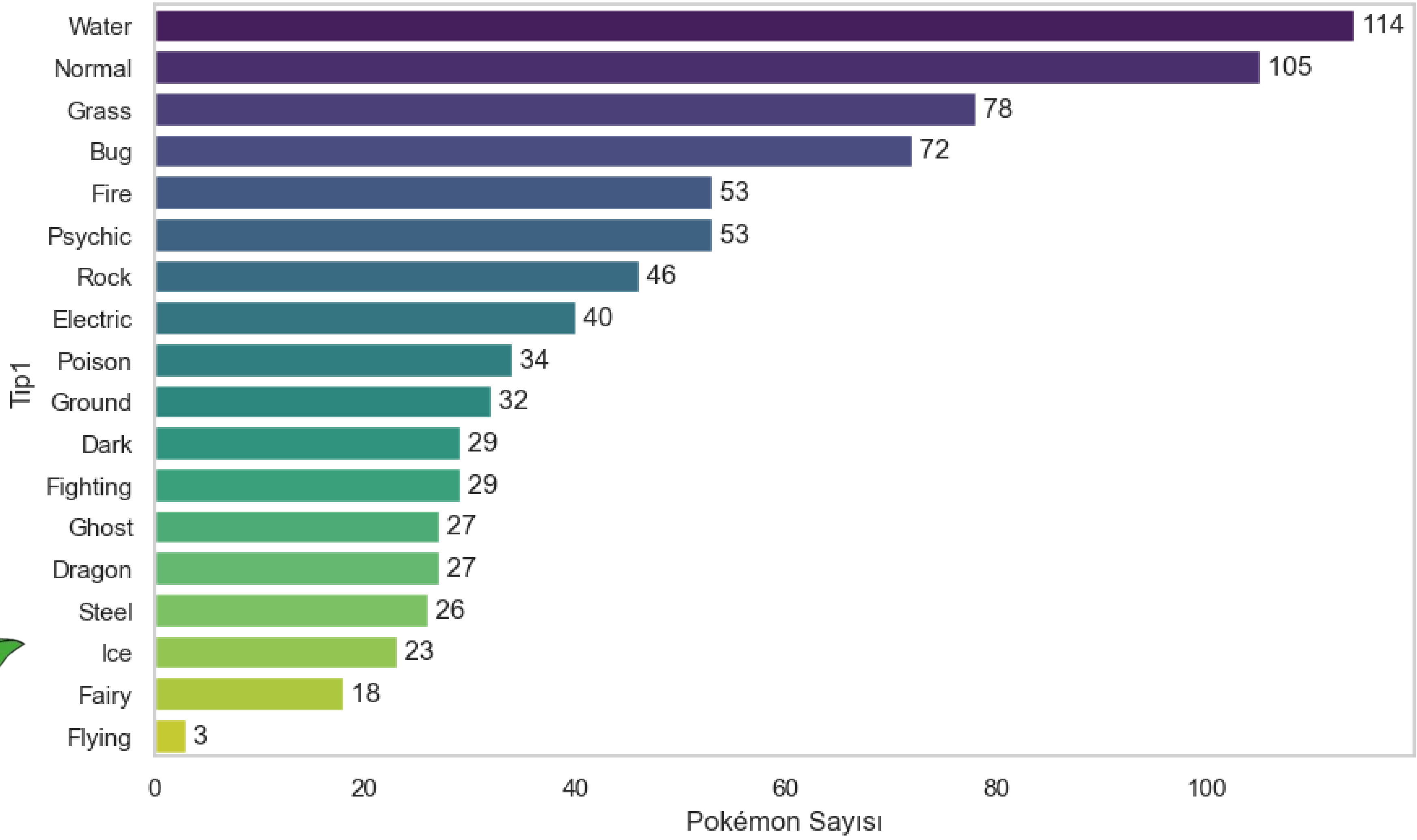
Dataset Source: "Pokemon Image Dataset" - kaggle
Dataset Scope: Consists of two main files

1. "images" folder,
   - 120x120 pixel RGB format images of 809 Pokémon characters.
2. "pokemon.csv" file,
   - Name: Pokémon's name (e.g. "Bulbasaur", "Charizard").
   - Type1: Pokémon's primary type (e.g. Normal, Fire, Water… - 18 types).
   - Type2: Pokémon's secondary type (if any).
   - Evolution: Evolution chain (e.g. "Bulbasaur" -> "Ivysaur" -> "Venusaur").

Training/ Validation/ Test Set Distribution:

- Training Set: 70%
- Validation Set: 15%
- Test Set: 15%

Methodology
Dataset



Methodology
Optimization and Regularization

Optimization Techniques Considered:

- Adam Optimizer: Provides fast and efficient learning with adaptive learning rates and momentum term. Mostly preferred in our models.
- SGD (Stochastic Gradient Descent): Accelerates the learning process by calculating gradients for each mini-batch.
- RMSProp: Calculates separate learning rates for each parameter and adjusts these rates using the moving average of gradients.
- Nadam: Combines the advantages of Adam optimizer with Nesterov momentum

Methodology
Optimization and Regularization

Regularization Techniques Considered:

- Dropout: Prevents overfitting by randomly disabling some neurons in the network.
- Early Stopping: Stops training if the validation loss does not improve for a certain period.
- L2 Regularization (Ridge Regression): Prevents overfitting by adding the square sum of the weights as a penalty term.
- L1_L2 Regularization (Elastic Net): Simplifies the model and prevents overfitting by adding the sum of the absolute and square values of the weights.
- Batch Normalization: Accelerates and stabilizes the learning process by normalizing the activations of each mini-batch.

Methodology
Classification Methods

Model Name	Input Layer	Dense Layers and Features	Output Layer	Optimisation and Loss Function	Data Augmentation
MLP Basic	120x120x3 colour image	1. Layer: 512 neurons, ReLU 2. Layer: 256 neurons, ReLU	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	No
MLP1	120x120x3 colour image	1. Layer: 32 neurons, l1_l2 regulariser, HeNormal, ReLU, BatchNormalisation, 0.5 Dropout 2. Layer: 32 neurons, l1_l2 regulariser, HeNormal, ReLU, BatchNormalisation, 0.5 Dropout	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	No
MLP2	120x120x3 colour image	1. Layer: 32 neurons, l1_l2 regulariser, HeNormal, ReLU, BatchNormalisation, 0.5 Dropout 2. Layer: 32 neurons, l1_l2 regulariser, HeNormal, ReLU, BatchNormalisation, 0.5 Dropout	Softmax, 18 Pokémon types	RMSProp (0.001 learning rate), categorical cross entropy	Yes
MLP3	120x120x3 colour image	1. Layer: 1024 neurons, BatchNormalisation, ReLU, 0.3 Dropout 2. Layer: 1024 neurons, BatchNormalisation, ReLU, 0.3 Dropout 3. Layer: 256 neurons, BatchNormalisation, ReLU, 0.3 Dropout	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	Yes
MLP4	120x120x3 colour image	1. Layer: 512 neurons, L2 regulariser, BatchNormalisation, ReLU, 0.5 Dropout 2. Layer: 256 neurons, L2 regulariser, BatchNormalisation, ReLU, 0.5 Dropout 3. Layer: 128 neurons, L2 regulariser, BatchNormalisation, ReLU, 0.5 Dropout	Softmax, 18 Pokémon types	Adam (0.0005 learning rate), categorical cross entropy	Yes
CNN Basic	120x120x3 colour image	1. Layer: 32 filters, 3x3, ReLU, Max-Pooling 2x2 2. Layer: 64 filters, 3x3, ReLU, Max-Pooling 2x2	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	No
CNN1	120x120x3 colour image	1. Layer: 16 filters, 3x3, ReLU, Max-Pooling 2x2, 0.25 Dropout 2. Layer: 32 filters, 3x3, ReLU, Max-Pooling 2x2, 0.25 Dropout 3. Layer: 64 filters, 3x3, ReLU, Max-Pooling 2x2, 0.25 Dropout	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	No
CNN2	120x120x3 colour image	1. Pre-Trained Layers: VGG16, top layers removed, layers untrained 2. Global Average Pooling 3. Dense Layer: 256 neurons, ReLU, L2 regulator (0.01), 0.5 Dropout	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	Yes
CNN3	120x120x3 colour image	1. Pre-Trained Layers: VGG16, top layers removed, layers untrained 2. Global Average Pooling 3. Layer: 512 neurons, ReLU, L2 regulariser (0.02), 0.5 Dropout, BatchNormalisation 4. Layer: 256 neurons, ReLU, L2 regulariser (0.02), 0.5 Dropout, BatchNormalisation	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	Yes
CNN4	120x120x3 colour image	1. Layer: 64 filters, 3x3, ReLU, Max-Pooling 2x2 2. Layer: 64 filters, 3x3, ReLU, Max-Pooling 2x2 3. Layer: 128 filters, 3x3, ReLU, Max-Pooling 2x2 4. Layer: 128 filters, 3x3, ReLU, Max-Pooling 2x2	Softmax, 18 Pokémon types	Adam (0.001 learning rate), categorical cross entropy	Yes

Results and Discussion
Classification Performance

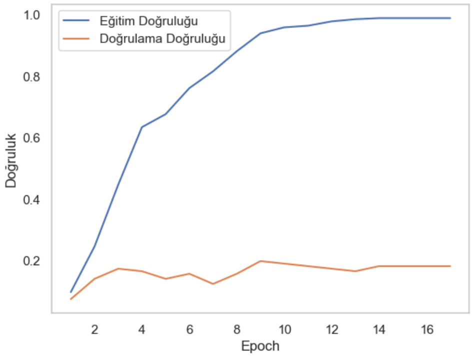
Model	Training Accuracy (%)	Validation Accuracy (%)	Test Accuracy (%)	Training Loss	Validation Loss	Test Loss
MLP Basic	98.9	18.2	18.0	0.193	3.326	3.104
MLP1	30.4	19.0	16.4	2.309	2.810	2.823
MLP2	19.4	25.6	22.1	2.721	2.633	2.720
MLP3	31.1	28.9	29.5	2.190	2.521	2.650
MLP4	26.1	29.8	24.6	3.141	3.294	3.384
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

Results and Discussion
Classification Performance

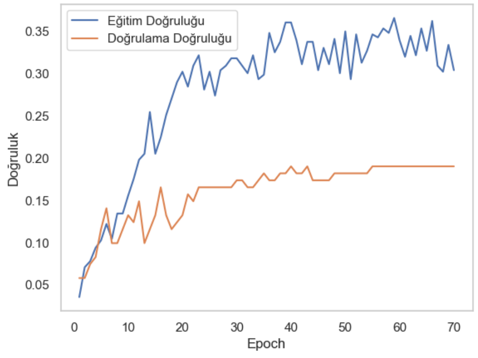
Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
MLP Basic	18.0	16.8	18.0	13.8
MLP1	16.4	11.0	16.4	11.5
MLP2	22.1	12.4	22.1	15.2
MLP3	29.5	24.0	29.5	23.8
MLP4	24.6	16.3	24.6	19.2
CNN Basic	18.0	12.7	18.0	11.3
CNN1	22.1	15.9	22.1	16.3
CNN2	29.5	25.3	29.5	23.5
CNN3	26.2	23.1	26.2	21.5
CNN4	27.9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

Results and Discussion
Visual Analysis

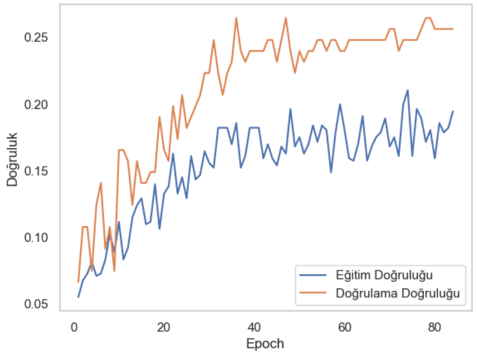
- Accuracy Graphs



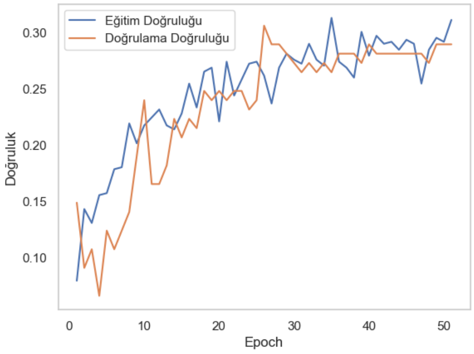
MLP Basic Model Training and Validation Accuracy



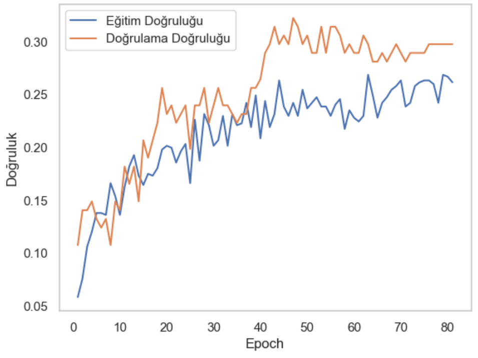
MLP1 Model Training and Validation Accuracy



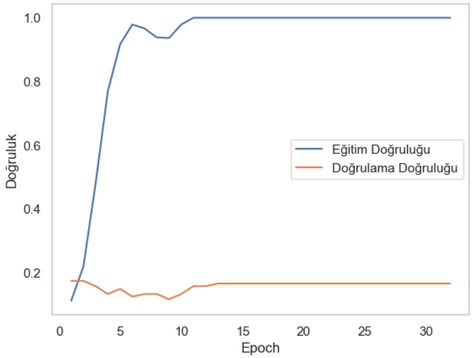
MLP2 Model Training and Validation Accuracy



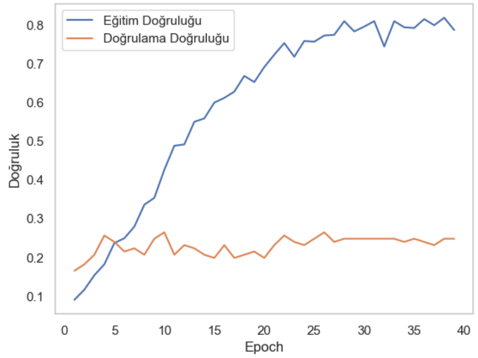
MLP3 Model Training and Validation Accuracy



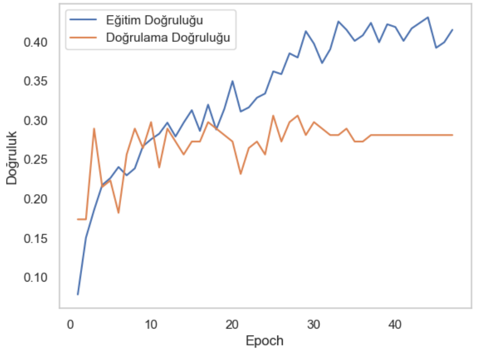
MLP4 Model Training and Validation Accuracy



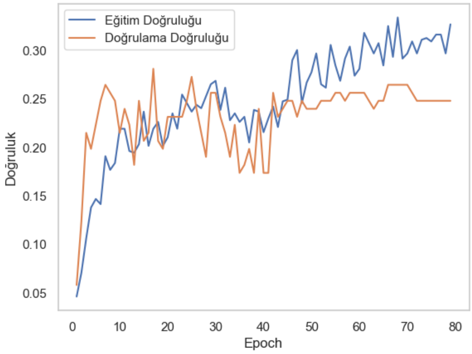
CNN Basic Model Training and Validation Accuracy



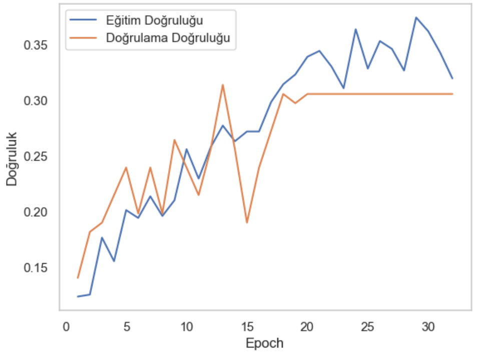
CNN1 Model Training and Validation Accuracy



CNN2 Model Training and Validation Accuracy



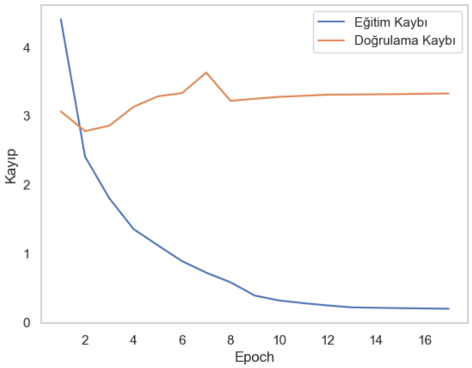
CNN3 Model Training and Validation Accuracy



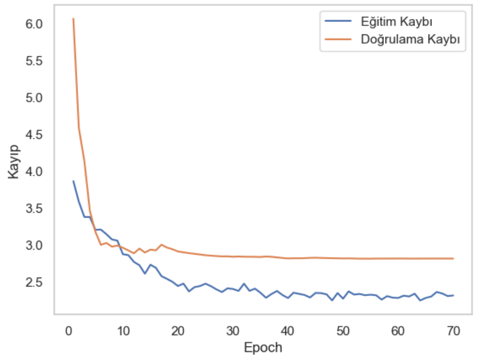
CNN4 Model Training and Validation Accuracy

Results and Discussion Visual Analysis

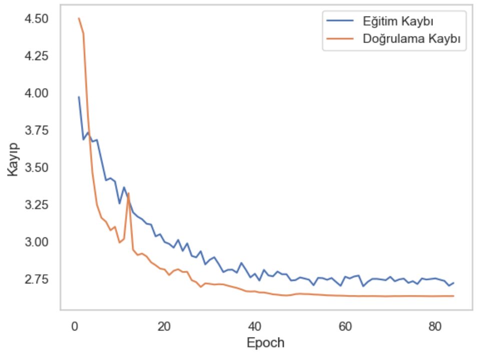
- Loss Graphs



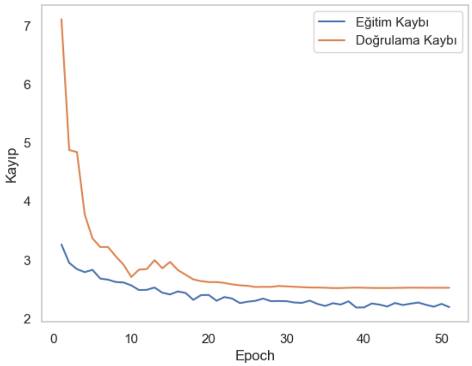
MLP Basic Model Training and Validation Loss



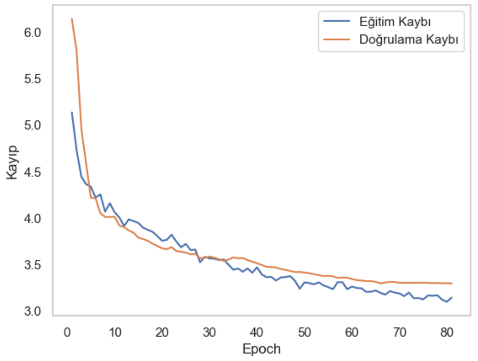
MLP1 Model Training and Validation Loss



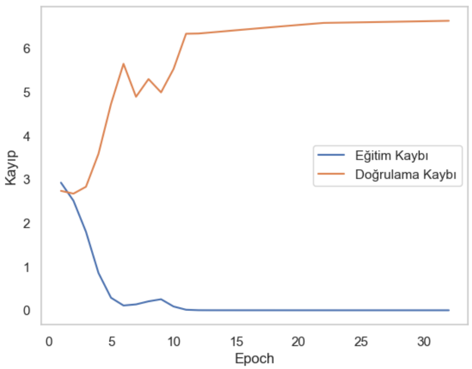
MLP2 Model Training and Validation Loss



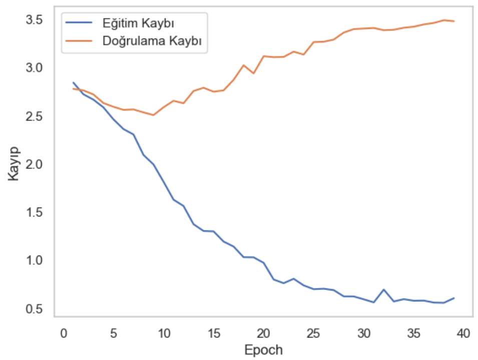
MLP3 Model Training and Validation Loss



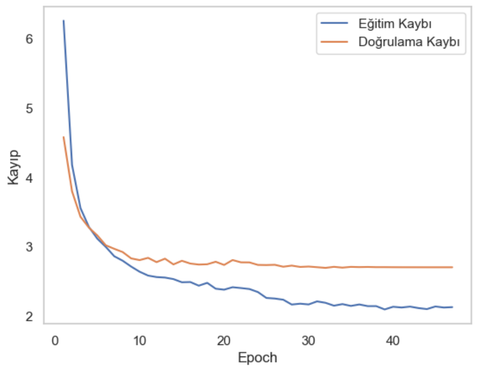
MLP4 Model Training and Validation Loss



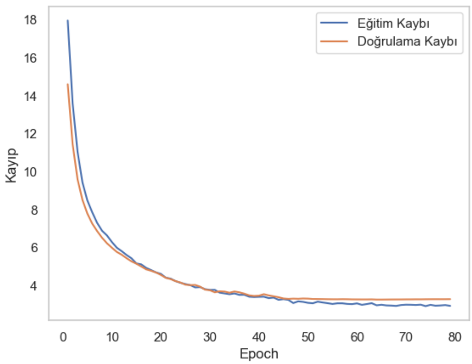
CNN Basic Model Training and Validation Loss



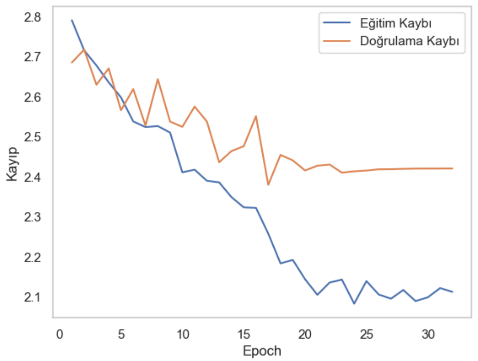
CNN1 Model Training and Validation Loss



CNN2 Model Training and Validation Loss



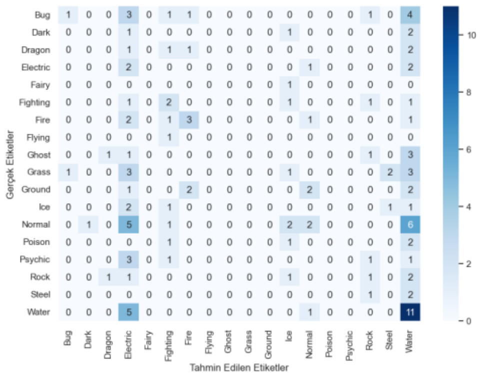
CNN3 Model Training and Validation Loss



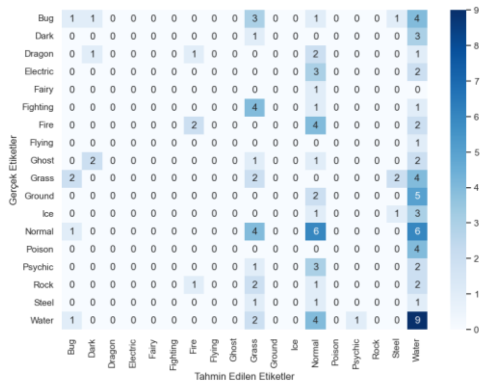
CNN4 Model Training and Validation Loss

Results and Discussion
Visual Analysis

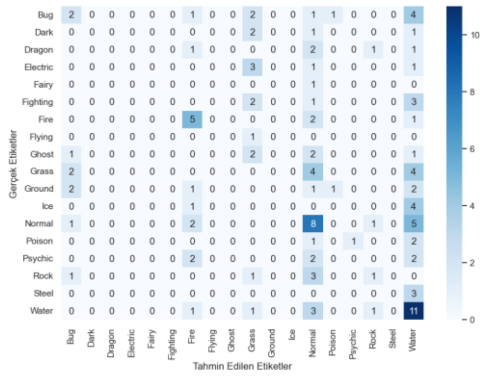
- Confusion Matrices



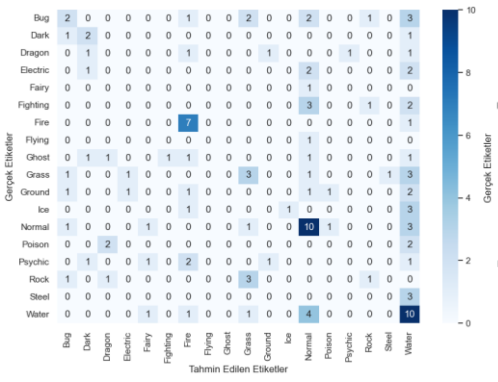
MLP Basic Model Confusion Matrix



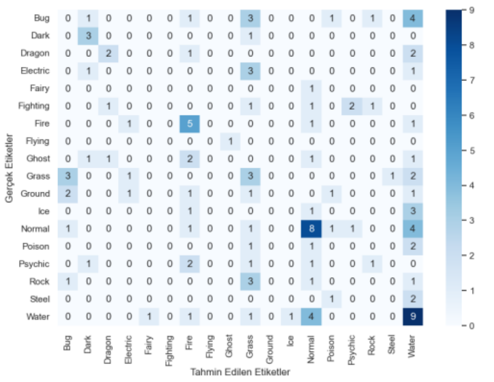
MLP1 Model Confusion Matrix



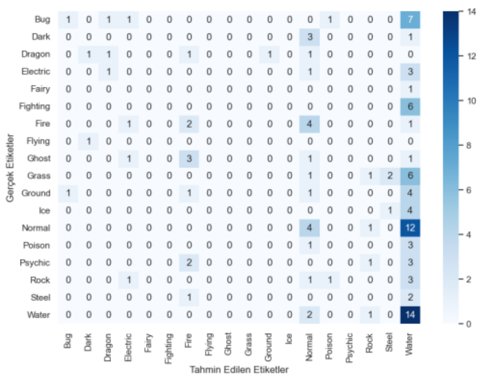
MLP2 Model Confusion Matrix



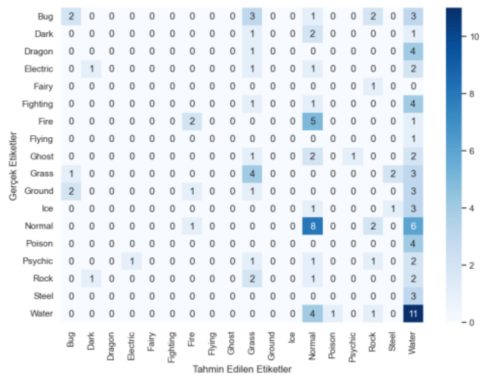
MLP3 Model Confusion Matrix



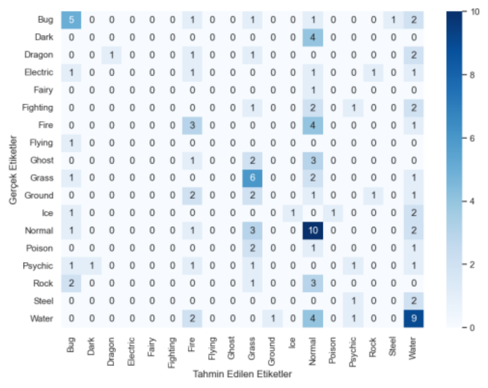
MLP4 Model Confusion Matrix



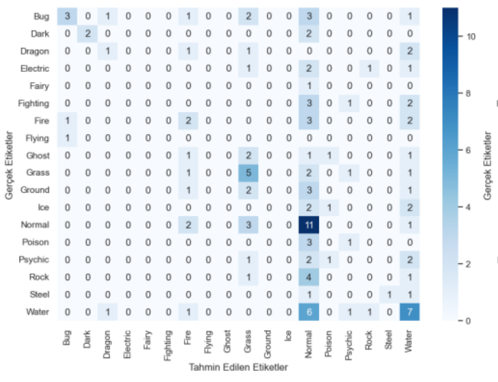
CNN Basic Model Confusion Matrix



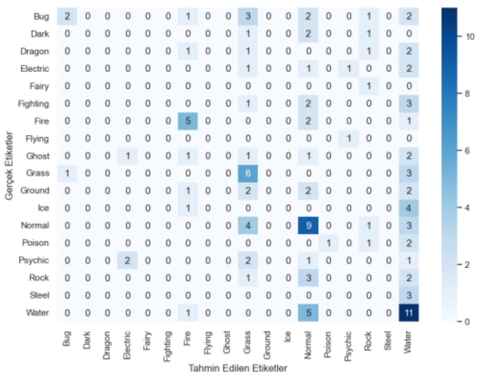
CNN1 Model Confusion Matrix



CNN2 Model Confusion Matrix



CNN3 Model Confusion Matrix



CNN4 Model Confusion Matrix

Discussion
Additional Observations and Findings

Imbalanced Class Distribution

- Some classes are less represented than others.
- Classes with few examples, like "Flying," result in low recall rates.

Similar Accuracy and Recall Values

- Models often predict the same classes.
- The imbalanced dataset brings accuracy and recall rates closer together.

Discussion
Overall Evaluation

General Model Performance

- MLP models showed high accuracy on the training set but low performance on the validation and test sets.
- CNN models learned visual features better due to their complex structures but struggled with generalization.

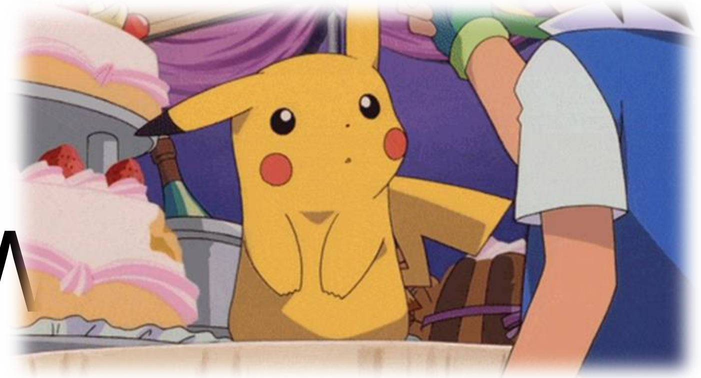
Best and Worst Performance

- Best performing model: MLP3, with more stable performance on the training and validation sets.
- Worst performing model: MLP Basic, high accuracy on the training set but low accuracy on the validation and test sets.

Dataset Imbalance

- The imbalanced class distribution in the dataset made it difficult for models to learn certain classes and led to low generalization ability.

Discussion
Recommendations and Future W



Improving Model Architectures

- Explore new learning architectures, activation functions, and regularization techniques.

Data Augmentation

- Utilize advanced data augmentation methods and Generative Adversarial Networks (GANs).

Balanced Training Set

- Resample the training set and consider class weights.

Optimization and Hyperparameter Tuning

- Implement different optimization algorithms and hyperparameter search techniques.

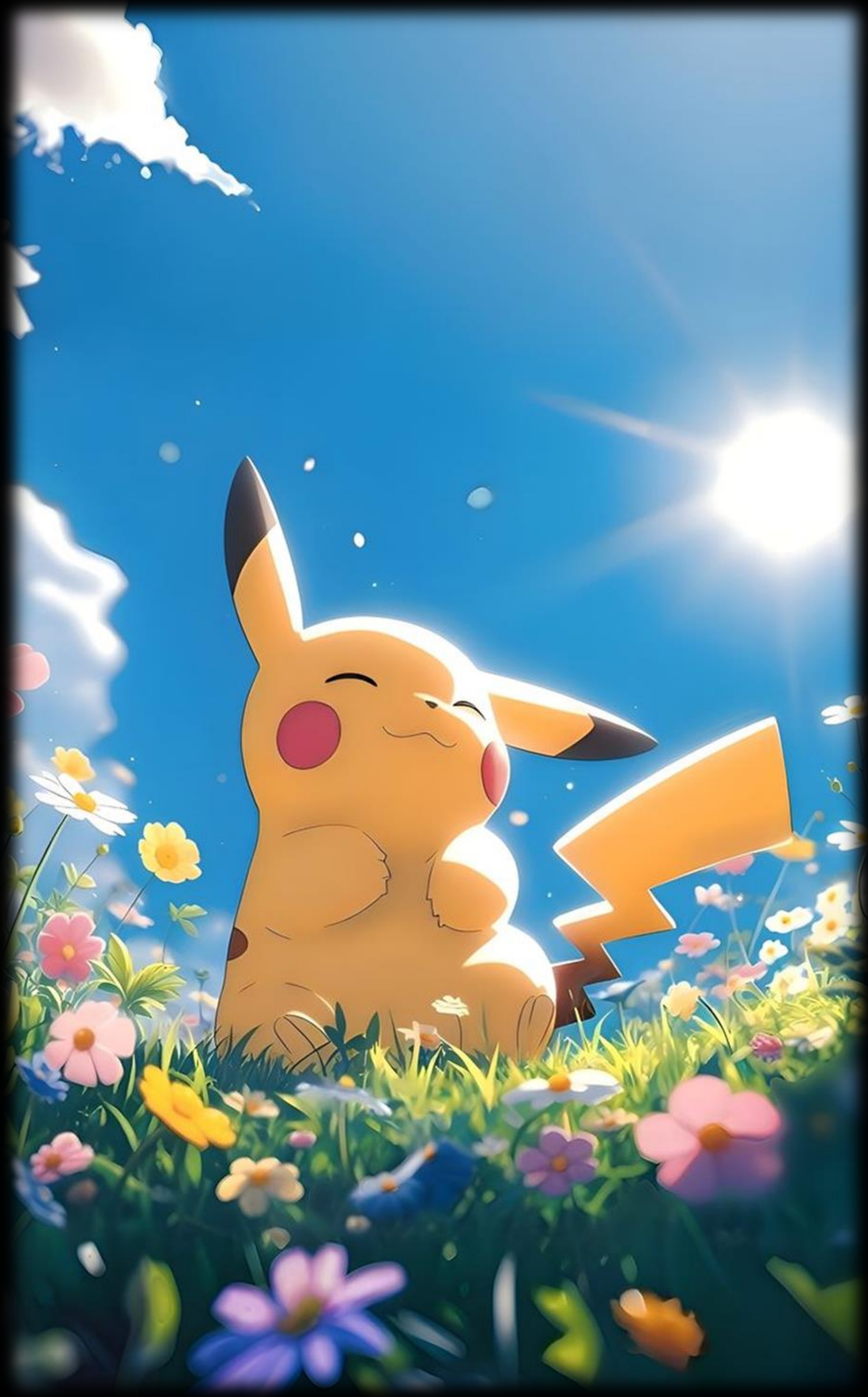
Powerful Hardware Resources

- Employ more powerful hardware resources.

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).

Thank you for your attention!



Contact Information:

Hacettepe University, Institute of Informatics, Turkiye
Mert Caliskan, mertcaliskan@hacettepe.edu.tr