Playing Card Recognition Using Deep Neural Networks: Architecture Design and Performance Analysis*

Jakub Jaromin[1]

[1]Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44100 Gliwice, Poland

Abstract

This article presents an approach to playing card recognition based on deep convolutional neural networks (CNNs). The model was trained from scratch on a publicly available dataset[1]. A range of image augmentation techniques was applied. The network architecture was designed to balance depth and computational efficiency. Experimental results confirmed the effectiveness of the proposed solution and indicated potential directions for further research in the field of image recognition. The solution has been shared on GitHub[2].

1. Introduction

Playing card recognition is a complex problem in computer vision that requires precise classification of images with subtle visual differences. This task becomes especially challenging in the presence of distortions such as variable lighting conditions or card orientation. This study aimed to design and evaluate a deep learning model capable of classifying images of playing cards. Recent strides in continual novel class discovery [3], uncertainty estimation in HDR imaging [4], second-attention for image captioning [5], and dynamic center point learning for multiple object tracking [6] present a rich toolkit of approaches that could be adapted and extended for the playing card recognition problem.

2. Related Work

Early research on image recognition using neural networks focused on simple architectures such as LeNet [7], which laid the foundation for later, more sophisticated and accurate models. A significant breakthrough came with AlexNet [8], which revolutionized image classification in the ImageNet competition through the use of Graphics Processing Units (GPUs). Further improvements were introduced by the VGG model [9], based on uniform 3x3 filters. ResNet [10] enabled the training of deep networks through the introduction of residual connections. EfficientNet [11] proposed scaling the architecture while maintaining optimal computational complexity, which allowed for better resource utilization. The model proposed in [12] was reported to use a yolo-v8 version for remote land classification, while [13] presented a lightweight pose-enhancing attention mechanism. Deep learning models are also used for novel class discovery [3] what is very important for data exploration. In the context of card recognition, some studies have combined deep learning approaches with classical image processing methods, achieving high accuracy with limited computing resources.

3. Dataset and Preprocessing

The experiment used the *Cards Image Dataset Classification*, available on Kaggle [1]. The dataset includes 53 classes representing a full deck of playing cards (with one joker class instead of two), with each class containing numerous images at a resolution of 224x224 pixels. Before training, the data was processed through preprocessing and augmentation operations. The following transformation techniques were applied:

*IVUS 2025: Information Society and University Studies 2025, May 15, Kaunas, Lithuania

jakub.jaromin2004@gmail.com (J. Jaromin)

Figure 1: Augmented images: cropping, flipping, color changes.

- random horizontal flips (p=0.5),
- random rotations (up to 15 degrees),
- brightness, contrast, saturation, and hue adjustments,
- affine translations (up to 10%) and perspective distortions (up to 0.2),
- normalization using mean and standard deviation values: [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225].

These augmentation methods increased the variety of the training data and improved the model's generalization capabilities on unseen inputs.

4. Examples of Augmented Images

Figure 1 presents examples of images after data augmentation, intended to enhance the model's ability to maintain its performance on various input variants.

5. Neural Network Architecture

The CNN model architecture was designed to balance depth and computational efficiency. Each block of the network was selected to optimize feature extraction and classification.

- **Convolutional layers (3x3)** – responsible for extracting local visual features such as edges, textures, and geometric patterns. The 3x3 filters provide a good trade-off between resolution and computational cost.
- **Batch Normalization** – stabilizes activation distributions, facilitates optimization, and improves resistance to overfitting.
- **ReLU activation functions** – introduce nonlinearity, allowing the modeling of complex relationships without the risk of vanishing gradients.
- **Max-pooling layers (2x2)** – reduce data dimensionality while retaining essential features, decreasing memory requirements.

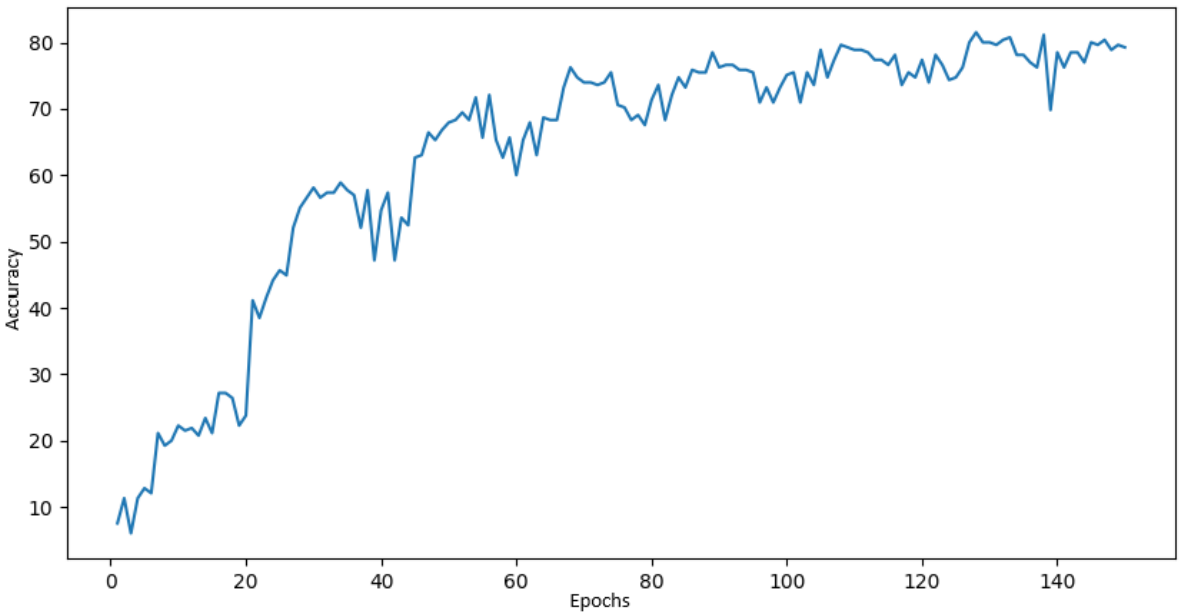

Figure 2: Learning curve showing accuracy increase over 150 epochs.

- **Dropout (p from 0.3 to 0.5 depending on the neural network block [2])** – prevents overfitting by randomly removing neurons during training.
- **Adaptive average pooling** – enables conversion of feature maps to a fixed size for fully connected layers.
- **Fully connected (dense) layers** – perform the final classification. A softmax function was applied to obtain probability distributions.

6. Experimental Setup

Experiments were conducted on a workstation equipped with:

- CPU: AMD Ryzen 9 9900X,
- GPU: NVIDIA RTX 3090,
- RAM: 64 GB DDR5.

The programming environment included Windows 11 and Python version 3.12. The model was implemented using the PyTorch library (version 2.6.0), with CUDA (version 12.6) for GPU acceleration. Code development and testing were carried out in Jupyter Notebook, integrated with Visual Studio Code.

The following libraries were used to support data processing and analysis:

- `torchvision` for image transformations and augmentations,
- `matplotlib` and `seaborn` for result visualization,
- `scikit-learn` for computing classification metrics and generating confusion matrices,
- `cv2` and `Pillow` for image processing.

Training was conducted using the `CrossEntropyLoss` loss function and the `Adam` optimizer with a `CosineAnnealingLR` learning rate scheduler. GPU acceleration reduced the training time to approximately 31 seconds per epoch.

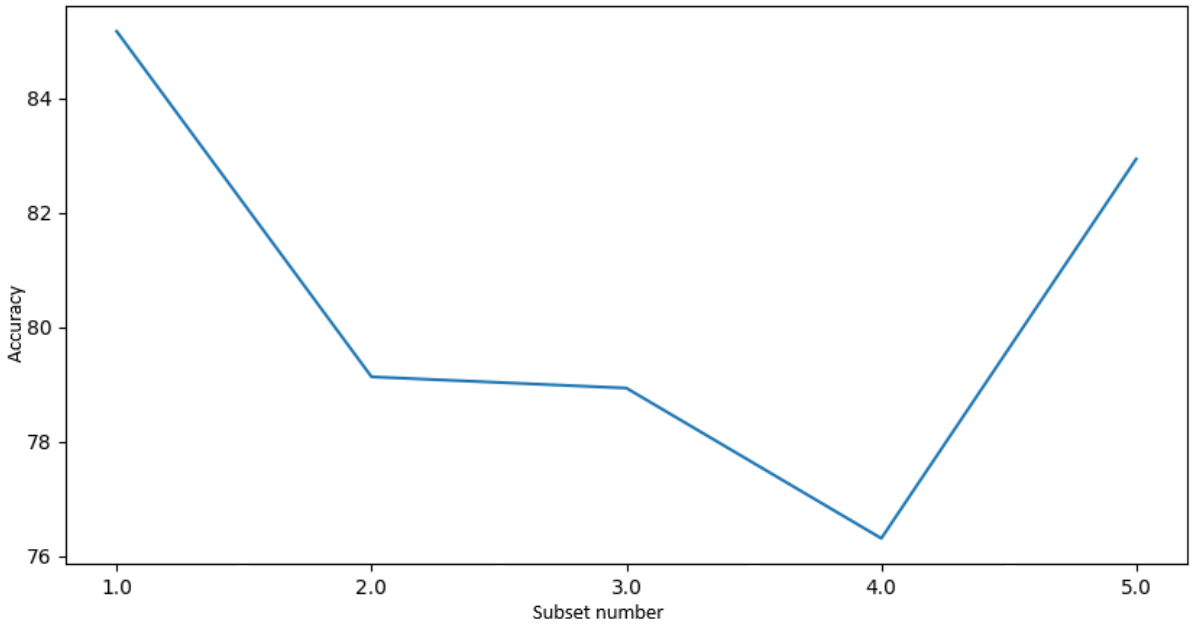

Figure 3: Testing curve showing the accuracy of individual testing subsets.

7. Experimental Results

The model was trained for 150 epochs using a training set divided into batches (batch size was 32). Model performance was evaluated on a separate test set, which was further divided into five subsets. The learning curve in Figure 2 shows an increase in accuracy and a reduction in loss over the training period. Figure 3 shows the accuracy on each subset of the test set.

8. Model Prediction Visualization

To illustrate the model's operation, example predictions are presented along with assigned probability values. Sample results are shown in Figures 4–7.

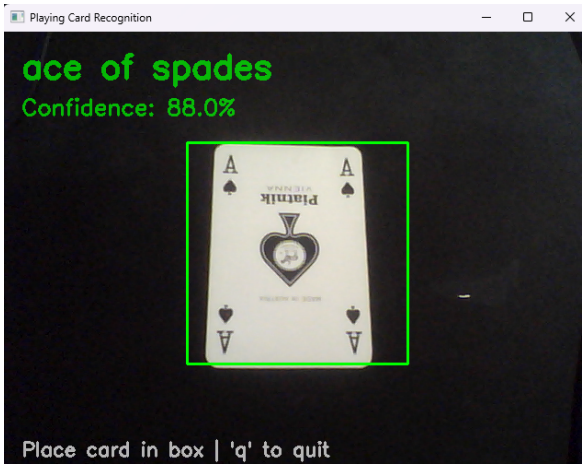

Figure 4: Prediction: Ace of Spades (88%).

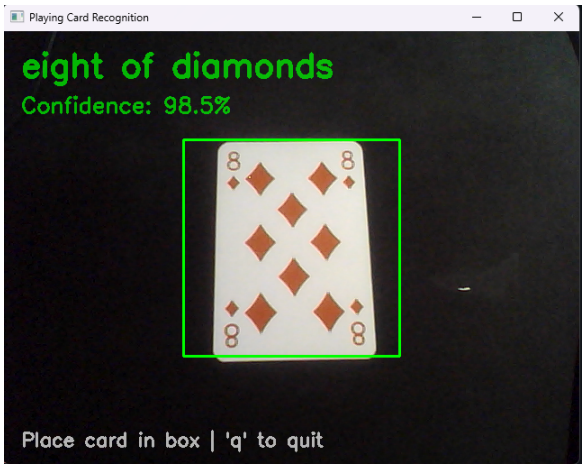

Figure 5: Prediction: 8 of Clubs (98.5%).

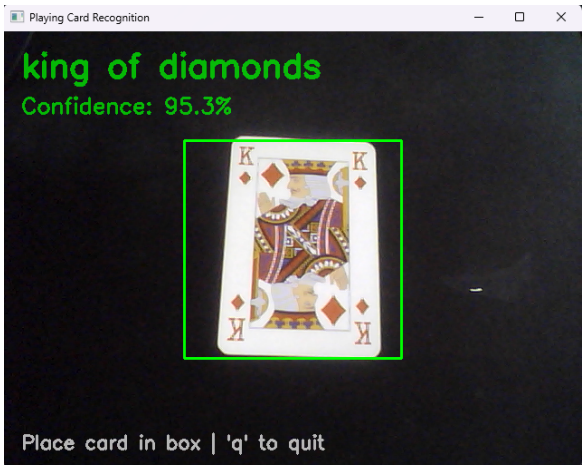

Figure 6: Prediction: King of Hearts (95.3%).

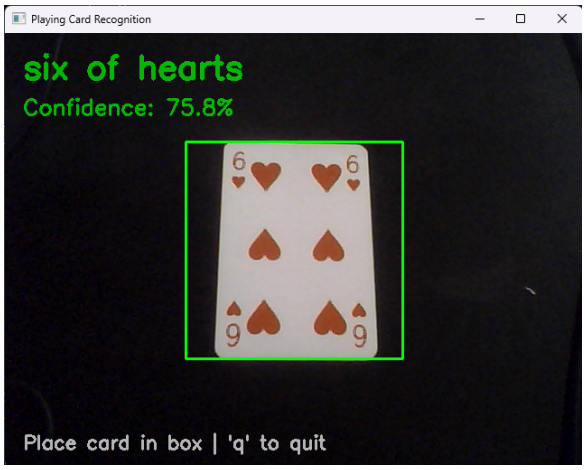

Figure 7: Prediction: 6 of Hearts (75.8%).

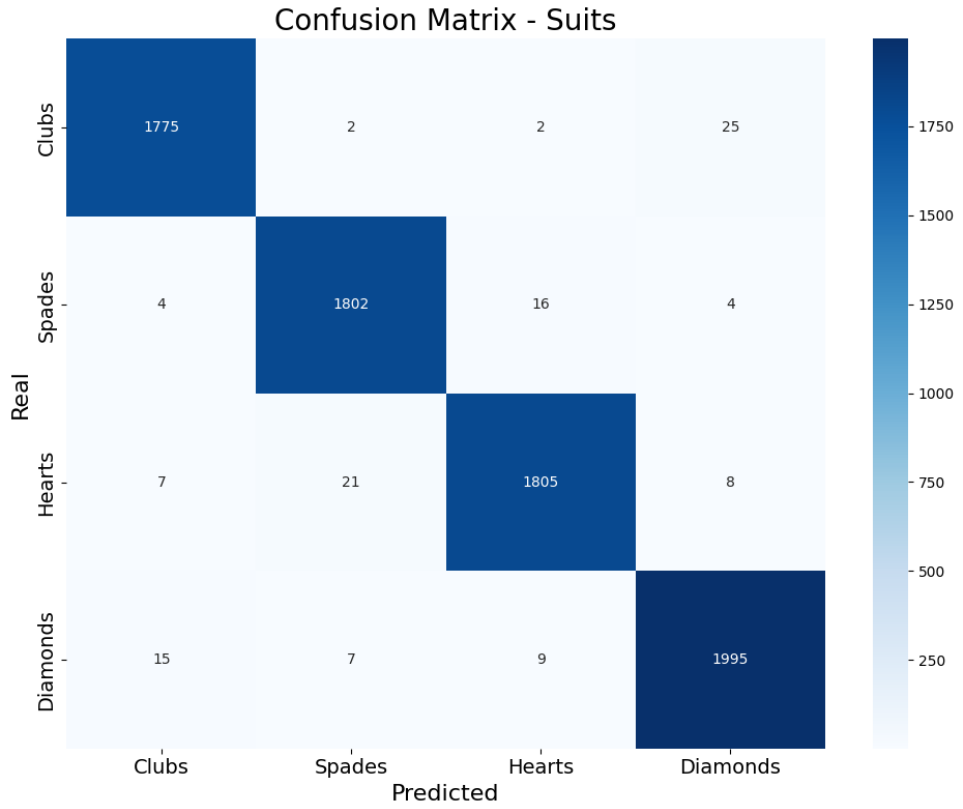

Figure 8: Confusion matrix of the test set for card suits (hearts, spades, diamonds, clubs).

9. Confusion Matrix Analysis

To evaluate the performance of the classification model by visually comparing its predictions to the actual outcomes, confusion matrices were generated (Figures 8 and 9). The results show that most misclassifications generally occurred between visually similar cards (e.g., Queen and King or Ace and Queen), although overall classification performance remained high.

10. Summary and Conclusions

The constructed neural network achieved high classification accuracy and robustness to common image distortions. This result confirms the effectiveness of using deep convolutional architectures combined with image augmentation techniques. During the project, the importance of appropriately selecting hyperparameters and employing regularization mechanisms was also demonstrated. The results of the confusion matrix analysis show that most classification errors occurred between similar cards, which opens the field for further improvements — for example, by introducing attention mechanisms or analyzing higher-resolution images.

11. Future Work

In further stages of research, the following directions are proposed:

- Optimization of the network architecture using neural architecture search (NAS),
- Implementation of attention mechanisms (e.g., SE-Blocks or transformers),
- Integration of the model into a real-time system, for example, on mobile or embedded devices,
- Extension of the dataset with additional classes or artificial examples using GAN and autoencoder models,
- Application of model quantization techniques to reduce memory footprint and increase inference speed.

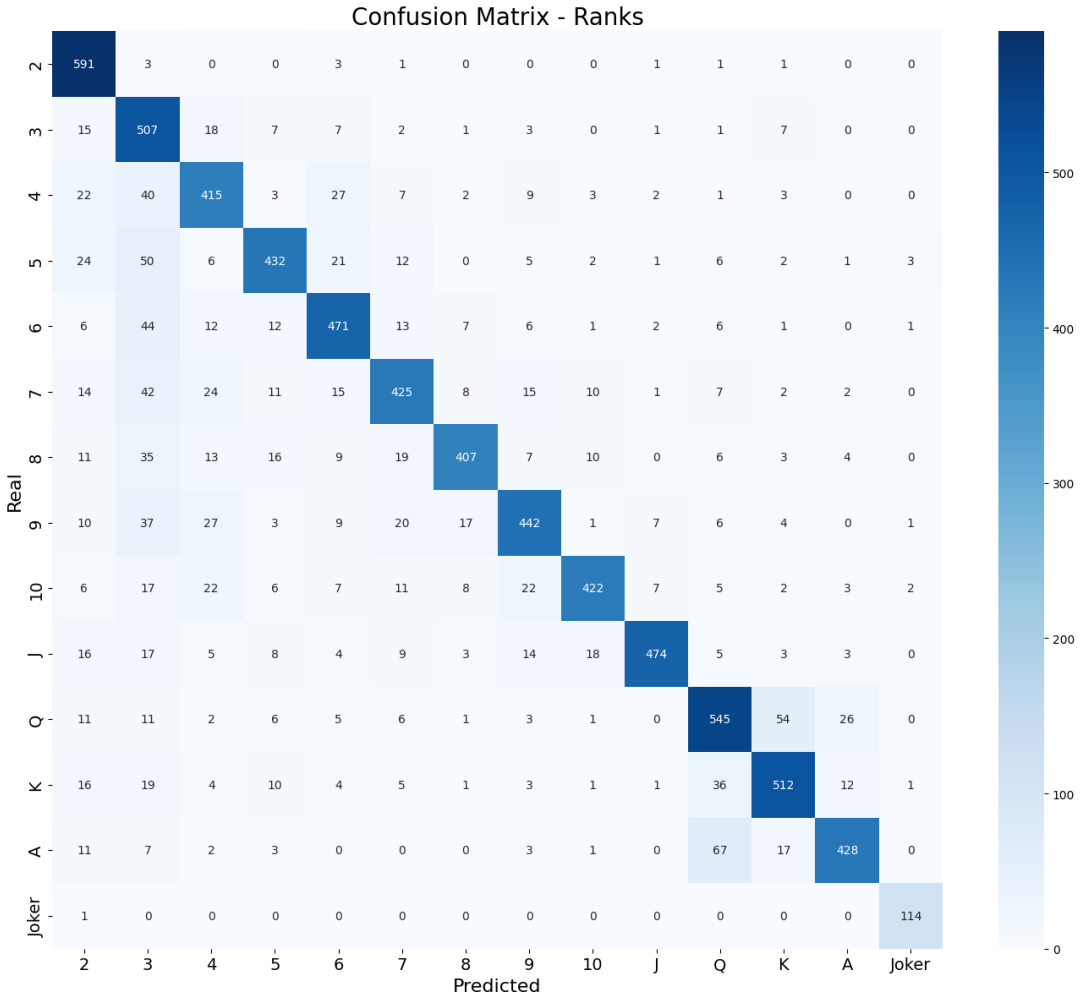

Figure 9: Confusion matrix of the test set for card values (A, 2–10, J, Q, K).

Declaration on Generative AI

During the preparation of this work, the author used Grammarly in order to: Grammar and spelling check. After using this tool, the author reviewed and edited the content as needed and takes full responsibility for the publication's content.

References

[1] Kaggle, Cards image dataset classification, 2023. Available at: https://www.kaggle.com/datasets/gpiosenka/cards-image-datasetclassification/data.

[2] J. Jaromin, Playing card recognition, 2025. Available at: https://github.com/Jaromek/playingCardRecognition.git.

[3] Q. Yan, Y. Yang, Y. Dai, X. Zhang, K. Wiltos, M. Woźniak, W. Dong, Y. Zhang, Clip-guided continual novel class discovery, Knowledge-Based Systems 310 (2025) 112920.

[4] Q. Yan, H. Wang, Y. Ma, Y. Liu, W. Dong, M. Woźniak, Y. Zhang, Uncertainty estimation in hdr imaging with bayesian neural networks, Pattern Recognition 156 (2024) 110802.

[5] X. Yang, Y. Yang, S. Ma, Z. Li, W. Dong, M. Woźniak, Samt-generator: A second-attention for image captioning based on multi-stage transformer network, Neurocomputing 593 (2024) 127823.

[6] Y. Hu, A. Niu, J. Sun, Y. Zhu, Q. Yan, W. Dong, M. Woźniak, Y. Zhang, Dynamic center point learning for multiple object tracking under severe occlusions, Knowledge-Based Systems 300 (2024) 112130.

[7] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE (1998).

[8] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems (NIPS), 2012.

[9] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556 (2014).

[10] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[11] M. Tan, Q. V. Le, Efficientnet: Rethinking model scaling for convolutional neural networks, in: Proceedings of the International Conference on Machine Learning (ICML), 2019.

[12] R. Vaghela, D. Vaishnani, P. N. Srinivasu, Y. Popat, J. Sarda, M. Woźniak, M. F. Ijaz, Land cover classification for identifying the agriculture fields using versions of yolo v8, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing (2025).

[13] Z. Tian, W. Fu, M. Woźniak, S. Liu, Pcdpose: enhancing the lightweight 2d human pose estimation model with pose-enhancing attention and context broadcasting, Pattern Analysis and Applications 28 (2025) 59.