Image Recognition Model Based on a Vector of Uncorrelated Features

Andriy Fesenko [1], Volodymyr Druzhynin [1], Nataliia Tsopa [2] and Vladyslav Synhaivskyi [1]

[1] *Taras Shevchenko National University of Kyiv, 60 Volodymyrska Street, Kyiv, 01033, Ukraine*
[2] *National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 37, Prospect Beresteiskyi, Kyiv, 03056, Ukraine*

Abstract

This article explores the task of image recognition based on multiple unrelated features, using the example of identifying circulating coins through convolutional neural networks. The paper outlines the conventional method of image recognition which employs a standard convolutional neural network with a single output, where each image corresponds to a distinct class. The language used is clear, objective, and value-neutral, and technical term abbreviations are explained upon first use. The structure is clear and logical, with causal connections between statements. The text is free from grammatical errors, spelling mistakes, and punctuation errors, and adheres to American English spelling and grammar conventions. An analysis of the results and methodology suggests that the present architecture is not optimal for datasets with a large number of classes. To enhance recognition accuracy, we advocate for a convolutional neural network architecture with multiple outputs. This method entails branching the network structure into several branches at a particular stage. By employing this neural network type, each image corresponds to a list of various independent characteristics, rather than a solitary composite category. This division creates multiple sub-tasks for recognizing images, each designated to a distinct branch of the neural network. A comparative analysis of the traditional neural network and a network with multiple outputs was conducted in the study. The identified objective of the study was to assess the pros and cons of each approach and impartially scrutinize the rationale for observed result disparities.
The results indicate that implementing a less conventional architecture of a convolutional neural network with multiple outputs can surmount the hindrances connected with a sizable quantity of composite classes and the indistinctness of ascertaining active coins by several traits. This suggested technique broadens the opportunities for utilizing neural networks in image recognition assignments having numerous classes and a scant quantity of training data.

Keywords [1]

Convolutional neural network, neural network with multiple outputs, image recognition, machine learning, artificial intelligence.

1. Introduction

Today, the convolutional neural network (CNN) is widely considered the most effective method for image recognition. The aim of this study is to create a CNN that can identify images based on a range of unrelated characteristics, as exemplified by its successful identification of circulating coins from various European countries. Additionally, alternative techniques were examined to enhance the precision of image recognition. To achieve this goal, we carefully scrutinized the traditional methodology for image recognition and made modifications to the neural network model that accounted for the unique characteristics of the input data.

Most features of a coin, such as its denomination, currency unit, country, year of issue, and mint markings, can be determined from the images of its front and back. However, measuring the diameter and weight of the coin necessitates the use of additional tools. Because capturing photographs of the

Information Technology and Implementation (IT&I-2023), November 20-21, 2023, Kyiv, Ukraine
EMAIL: volodymirdruzhynin68@gmail.com (A. 1); zhurakovskiybyu@tk.kpi.ua (A. 2), tzapanv@gmail.com (A. 3); vladssss19@gmai.com (A. 4)
ORCID: 0000-0002-5340-6237 (A. 1); 0000-0003-3990-5205 (A. 2); 0000-0002-3215-0711 (A. 3)

coin's obverse and reverse is effortless and requires minimal equipment, users can recognize coins most conveniently by using these photos to identify their critical characteristics.

To effectively train a CNN, having a large dataset of labeled images that correspond to specific categories is crucial. Although specialized websites on the internet offer readily available datasets for most image recognition tasks, this research necessitates the manual labeling of images from freely available online sources and personal photographs since there is no available premade dataset.

2. Analysis of Recent Research

When addressing image recognition tasks, it is customary to employ a classic approach - a standard convolutional neural network initially developed by Yann LeCun in 1995 [1]. Such a network accepts an image as its input and generates a vector of numbers that demonstrates the neural network's degree of confidence in associating the image with each designated category. In this approach, images within the training dataset are assigned to distinct classes based on their distinctive and defining features. For identification of circulating coins, it is necessary to assign a unique attribute to each coin image comprising its denomination, currency type, and country. Consequently, a separate "1 coin, Ukraine" category would be created for each image. For European nations, this classification scheme would potentially result in hundreds of such categories. However, due to the dataset's limited number of images, each class would end up with a small number of images.

A conventional convolutional neural network includes sequential convolution layers with the ReLU activation function and utilizes filters, as shown in Fig. 1 [2]. As the image is processed, these filters generate feature maps for each filter, which are then sub-sampled by pooling layers to decrease their spatial dimensions [3]. Image classification is performed at the end of this network utilizing one or more fully connected neural layers.

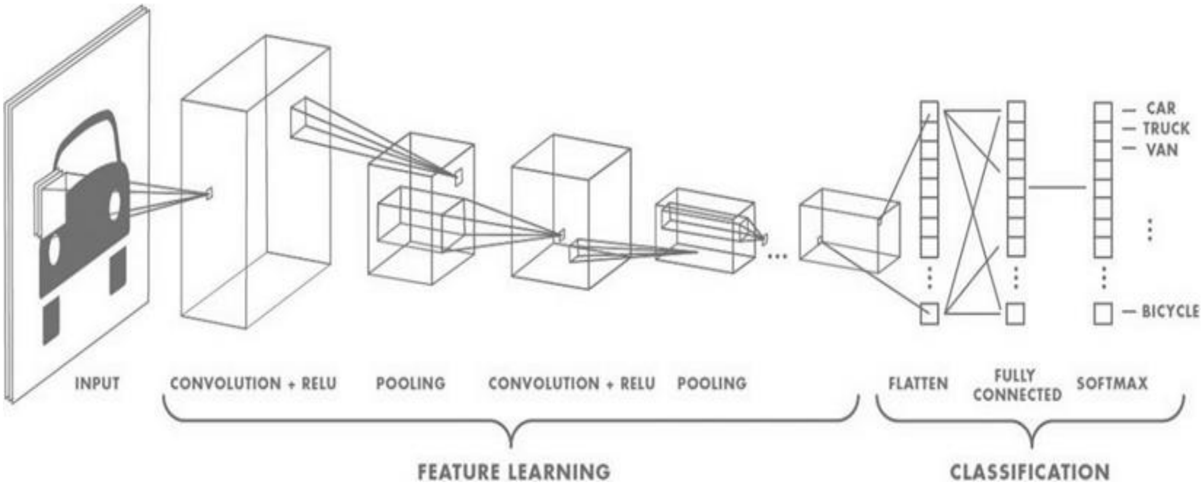


Figure 1: Typical Convolutional Neural Network.

For numerous image recognition tasks, a pre-existing neural network architecture is frequently utilized, with architectures that have won the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [4-9] often being selected. This challenge assesses algorithms for image recognition and object detection. The VGG16 [10-14] and ResNet50 [15-19] networks are the most commonly employed, both trained on the ImageNet dataset. The study utilizes Stevo Bozinovski's [20] transfer learning technique to transfer knowledge from the extensive ImageNet dataset to a smaller one. However, due to the specificity of the input data and limited neural network training resources, this approach does not yield notably improved outcomes. As a result, a tailored neural network design is imperative.

3. Main Section

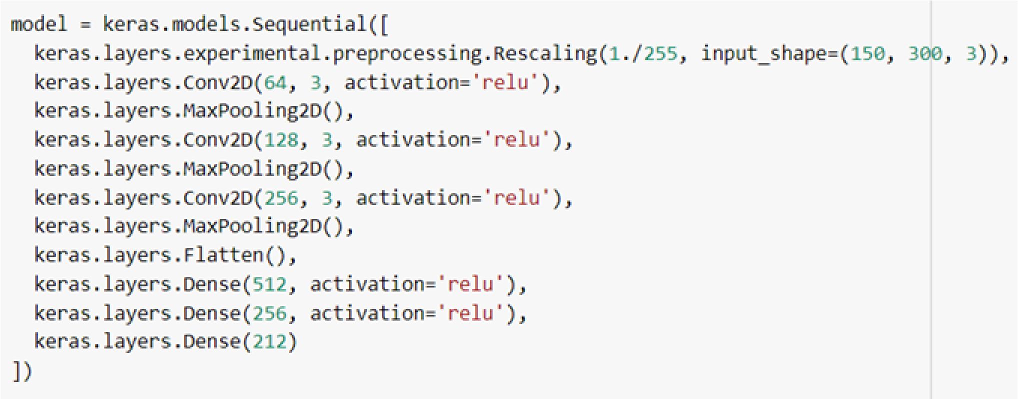
For this study, a dataset was compiled that included over 40,000 photographs of current coins from 36 countries, including Ukraine, Poland, the United Kingdom, and other European Union nations. The dataset features two square images (150x150 pixels) of each coin, showcasing the obverse and reverse sides against either a white or light gray backdrop. All photographs were standardized at 300 pixels wide and 150 pixels high. The dataset was split into two parts. 80% of the total amount was utilized as training data for the neural network. The other 20% of the dataset was used as test data to evaluate the network's performance. Within the training data, 80% was solely reserved for training, while 20% was employed for validation during each epoch of neural network training. This method enabled swift overfitting detection. Table 1 displays the distribution of images in the dataset across various countries.

Table 1
Distribution of coin images by European countries.

№	Country	Number of images	№	Country	Number of images
1	Austria	1678	19	Monaco	1236
2	Belgium	2122	20	Netherlands	2066
3	Bulgaria	414	21	Germany	6942
4	Vatican	1050	22	Norway	490
5	United Kingdom	3252	23	Poland	2166
6	Greece	1550	24	Portugal	1356
7	Denmark	976	25	Romania	572
8	Estonia	470	26	San Marino	1454
9	Ireland	1436	27	Slovakia	650
10	Iceland	558	28	Slovenia	674
11	Spain	2336	29	Hungary	1296
12	Italy	2324	30	Ukraine	2186
13	Cyprus	702	31	Finland	2280
14	Latvia	472	32	France	5280
15	Lithuania	438	33	Croatia	1452
16	Luxembourg	1942	34	Czech Republic	1380
17	Malta	926	35	Switzerland	3506
18	Moldova	686	36	Sweden	1054

The dataset comprised 353 image classes categorized based on the format "country of origin - currency unit - denomination - coin type." The pictures of coins, following a 1:1 aspect ratio, displaying the obverse or reverse side of the coin, were photographed against a white, light gray, light blue, or light yellow background. Depending on the technical specifications of the camera used, as well as the intensity and color of the lighting, the background hue shifted correspondingly. To enlarge the dataset and ensure precise image recognition even under suboptimal conditions, data augmentation was employed. The neural network was developed using the Google Colab environment, which provides ample computational power for creating, training, testing, and deploying neural networks. Additionally, users are able to work with TensorFlow, the open-source machine learning library [21], and its high-level API, Keras [22], through Google Colab notebooks. Although TensorFlow supports multiple programming languages, Python was chosen for its widespread use in developing neural networks. Given the atypical rectangular format of the input data (300x150 pixels) and narrow standardization in image presentation (both sides of the coin displayed on a white or light gray background), and limited resources available, a customized neural network was chosen for analysis. See Figure 2 for the code outlining this network.

This neural network consists of three pairs of convolutional layers, each consisting of 64, 128, and 256 filters that are 3x3 in size, which alternate with max pooling layers. The max pooling layers feature a. Filters of dimensions 2x2 recognize and transfer the highest value to the subsequent layer within the neural network. The network ends with two totally connected layers equipped with the ReLU activation function, with a capacity of 512 and 256 neurons, respectively. Additionally, the neural network includes a single fully connected layer composed of 212 neurons for the purpose of classification. Figure 3 illustrates the resulting graph of the network. The network's performance results, obtained after training for 10 epochs, are displayed in Figure 4.



```
model = keras.models.Sequential([
  keras.layers.experimental.preprocessing.Rescaling(1./255, input_shape=(150, 300, 3)),
  keras.layers.Conv2D(64, 3, activation='relu'),
  keras.layers.MaxPooling2D(),
  keras.layers.Conv2D(128, 3, activation='relu'),
  keras.layers.MaxPooling2D(),
  keras.layers.Conv2D(256, 3, activation='relu'),
  keras.layers.MaxPooling2D(),
  keras.layers.Flatten(),
  keras.layers.Dense(512, activation='relu'),
  keras.layers.Dense(256, activation='relu'),
  keras.layers.Dense(212)
])
```

Figure 2: Neural Network Definition

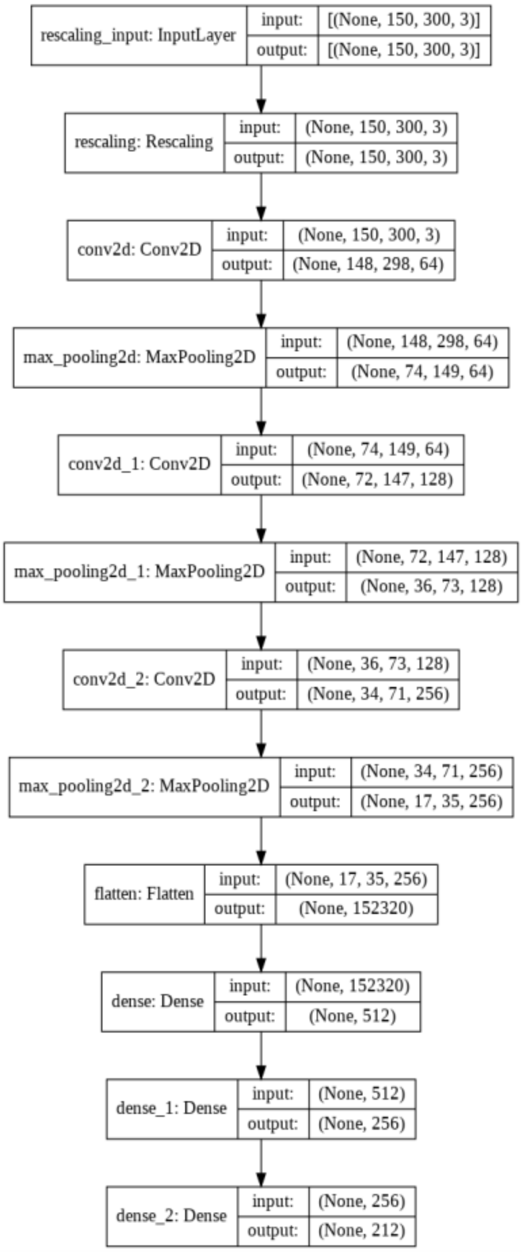
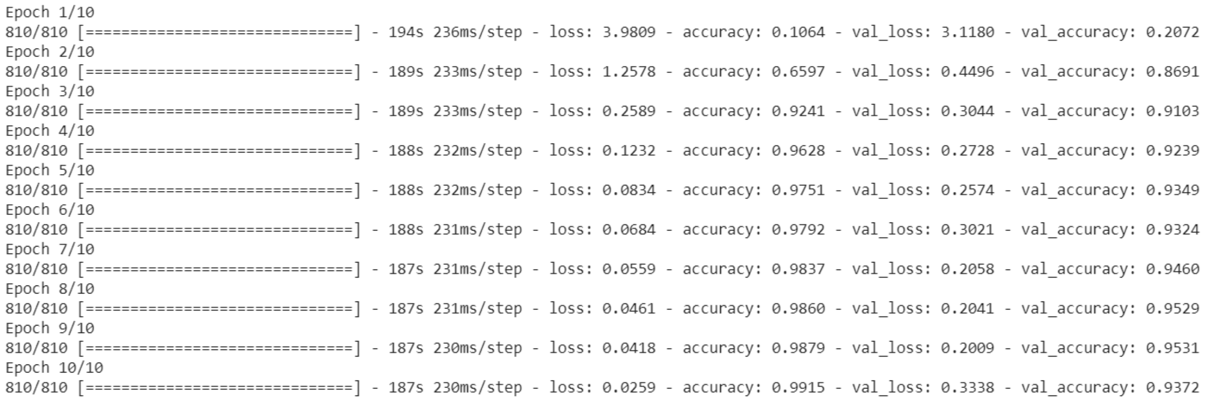

rescaling_input: InputLayer | input: [(None, 150, 300, 3)] | output: [(None, 150, 300, 3)]
rescaling: Rescaling | input: (None, 150, 300, 3) | output: (None, 150, 300, 3)
conv2d: Conv2D | input: (None, 150, 300, 3) | output: (None, 148, 298, 64)
max_pooling2d: MaxPooling2D | input: (None, 148, 298, 64) | output: (None, 74, 149, 64)
conv2d_1: Conv2D | input: (None, 74, 149, 64) | output: (None, 72, 147, 128)
max_pooling2d_1: MaxPooling2D | input: (None, 72, 147, 128) | output: (None, 36, 73, 128)
conv2d_2: Conv2D | input: (None, 36, 73, 128) | output: (None, 34, 71, 256)
max_pooling2d_2: MaxPooling2D | input: (None, 34, 71, 256) | output: (None, 17, 35, 256)
flatten: Flatten | input: (None, 17, 35, 256) | output: (None, 152320)
dense: Dense | input: (None, 152320) | output: (None, 512)
dense_1: Dense | input: (None, 512) | output: (None, 256)
dense_2: Dense | input: (None, 256) | output: (None, 212)

Figure 3: Neural Network Graph

For further clarity, accuracy and loss function graphs for both training and validation data are also shown in Figure 5.



```
Epoch 1/10
810/810 [==============================] - 194s 236ms/step - loss: 3.9809 - accuracy: 0.1064 - val_loss: 3.1180 - val_accuracy: 0.2072
Epoch 2/10
810/810 [==============================] - 189s 233ms/step - loss: 1.2578 - accuracy: 0.6597 - val_loss: 0.4496 - val_accuracy: 0.8691
Epoch 3/10
810/810 [==============================] - 189s 233ms/step - loss: 0.2589 - accuracy: 0.9241 - val_loss: 0.3044 - val_accuracy: 0.9103
Epoch 4/10
810/810 [==============================] - 188s 232ms/step - loss: 0.1232 - accuracy: 0.9628 - val_loss: 0.2728 - val_accuracy: 0.9239
Epoch 5/10
810/810 [==============================] - 188s 232ms/step - loss: 0.0834 - accuracy: 0.9751 - val_loss: 0.2574 - val_accuracy: 0.9349
Epoch 6/10
810/810 [==============================] - 188s 231ms/step - loss: 0.0684 - accuracy: 0.9792 - val_loss: 0.3021 - val_accuracy: 0.9324
Epoch 7/10
810/810 [==============================] - 187s 231ms/step - loss: 0.0559 - accuracy: 0.9837 - val_loss: 0.2058 - val_accuracy: 0.9460
Epoch 8/10
810/810 [==============================] - 187s 231ms/step - loss: 0.0461 - accuracy: 0.9860 - val_loss: 0.2041 - val_accuracy: 0.9529
Epoch 9/10
810/810 [==============================] - 187s 230ms/step - loss: 0.0418 - accuracy: 0.9879 - val_loss: 0.2009 - val_accuracy: 0.9531
Epoch 10/10
810/810 [==============================] - 187s 230ms/step - loss: 0.0259 - accuracy: 0.9915 - val_loss: 0.3338 - val_accuracy: 0.9372
```

Figure 4: Neural Network Training Results

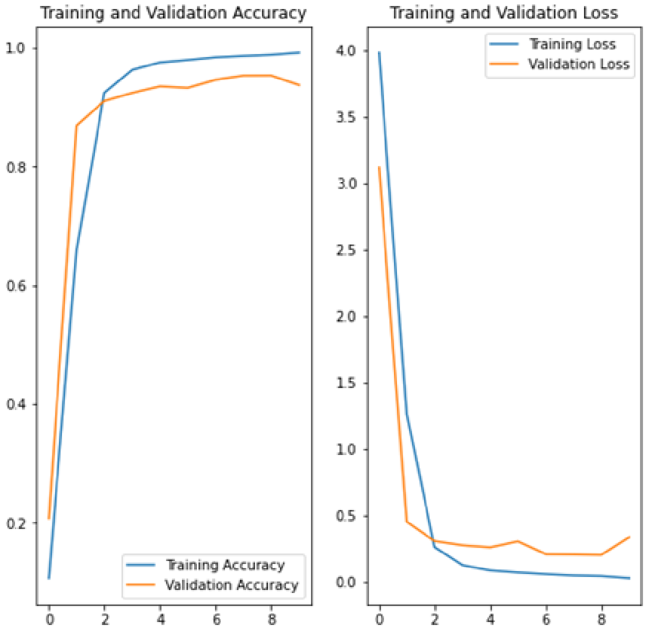


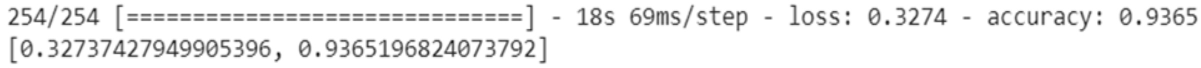
Figure 5: Graph of Recognition Accuracy and Loss Function

Based on the validation data graphs, it is apparent that increasing the number of epochs leads to overfitting. This happens when the network memorizes the training data, resulting in improved recognition accuracy at the cost of maintaining almost the same accuracy on validation data. The recognition accuracy on test data was 0.9365 (Figure 6).

Numerous experiments were performed on the neural network in an attempt to boost its complexity by adding more layers and neurons throughout the project. Regrettably, these adjustments were determined to have negligible impact on the recognition accuracy of test data, which persevered within the range of 93-94%. On the other hand, there was a significant surge in both computational resources demand and training time. To improve recognition accuracy, the possibilities include enlarging the image dataset through image augmentation or using an alternative neural network architecture. A probable solution, considering the specificity of the data, would be to adopt a convolutional neural network with multiple outputs.

4. Overview of the neural network with multiple outputs

When examining why traditional convolutional neural networks present relatively low accuracy rates, an unusual aspect of the data comes to light. The current definition of traits of circulating coins, specifically "denomination, currency unit, country," only allows for limited image categorization, resulting in a substantial number of classes (212, specifically). Within each category, there is a paucity of images, and having a restricted number of images across numerous categories leads to a decline in training precision.



```
254/254 [==============================] - 18s 69ms/step - loss: 0.3274 - accuracy: 0.9365
[0.32737427949905396, 0.9365196824073792]
```

Figure 6: Testing results of the neural network

After further examination, it is clear that each coin classification comprises three distinct characteristics: denomination, currency unit, and country of origin. None of these traits alone can provide enough information for definitive identification of a specific coin. Focusing exclusively on denomination, for example, can result in significantly fewer classifications for the same number of training images. Expanding the dataset to include more images within each category enhances recognition accuracy for that particular feature. The accuracy of identifying coins based solely on their currency or country of origin is similarly increased. Additionally, augmenting the dataset by incorporating images of modern circulating coins from another country to the combined feature may require adding approximately ten new classes. However, the list of country classes would only require one new class, while zero to two new classes may be added for currency unit and denomination classes. When working with a dataset that includes many composite classes, using a convolutional neural network with multiple outputs [23] can prove to be beneficial. This architecture receives an input image and generates several correspondence vectors, each corresponding to a different classification. To achieve this, the neural network branches at a specific point, resulting in the creation of multiple branches. This branching can occur at any stage of the neural network (refer to Figure 7).

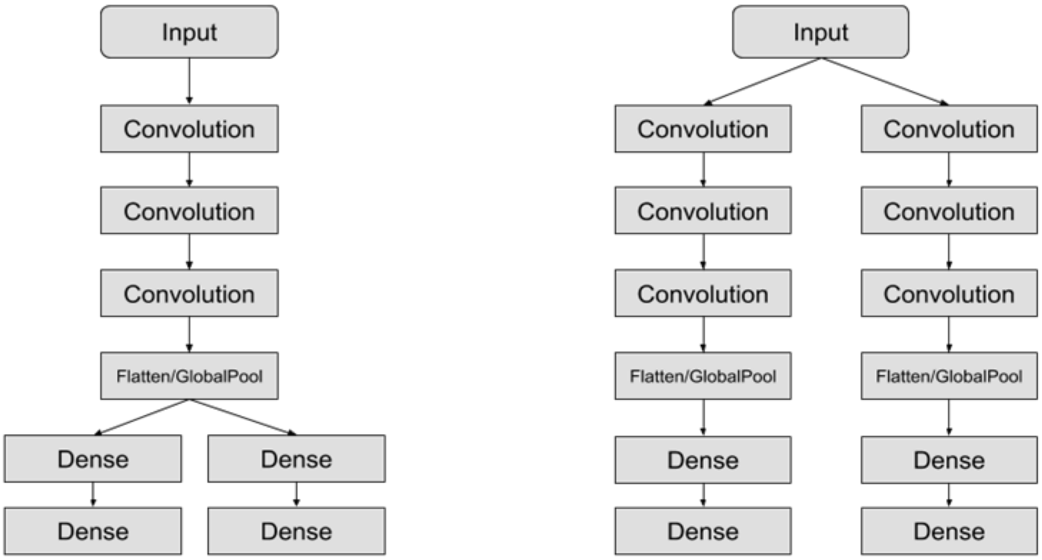
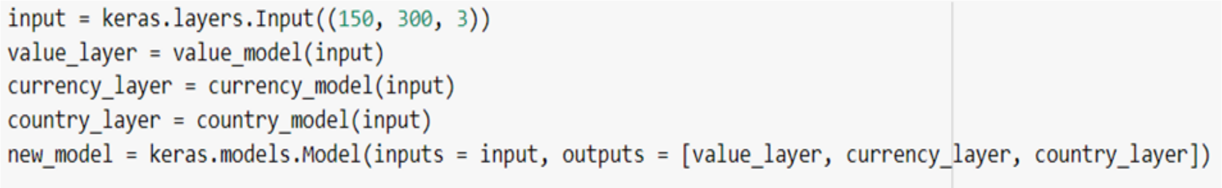


Figure 7: Examples of neural networks with multiple outputs

This network architecture is commonly used to collect data from different formats that demand distinct processing methods, incorporating outcomes from classification and regression analysis [24]. Moreover, it can handle images containing multiple unrelated characteristics. When utilizing a convolutional neural network with multiple outputs for coin recognition, a total of three correspondence vectors can be acquired, with each vector corresponding to a specific denomination, currency unit, and country. Through the implementation of this approach, the recognition accuracy for each feature is significantly improved in comparison to a regular network featuring a solitary output that utilizes compound characteristics as classes, according to source [12].

Another advantage of this approach is the ability to fine-tune the neural network with greater accuracy and conserve resources during training by incorporating pre-trained models as branch layers after branching. This is because disparate feature models may require different amounts of training epochs. In this study, we implemented a neural network that consists of three branches at the beginning, each of which contains a single convolutional model layer. The layer order does not affect the overall neural network, but it enables individual branch training and weight usage in the general network. Figure 8 showcases the code for the general neural network, while Figure 9 depicts its graphical representation.

The value_model, currency_model, and country_model function as grouped layers and identify the denomination, currency type, and country, respectively, using a structure similar to that of a conventional convolutional neural network model (refer to Figure 2). Technical term abbreviations are explained upon first use. The language is clear, concise, and objective, avoiding biased or figurative language and maintaining formal register. The text adheres to common academic structure and citation styles, and is free of grammatical and spelling errors. However, they differ by having fewer neurons in the final classification layer: eight for the denomination and currency models and twenty-six for the country model.



```
input = keras.layers.Input((150, 300, 3))
value_layer = value_model(input)
currency_layer = currency_model(input)
country_layer = country_model(input)
new_model = keras.models.Model(inputs = input, outputs = [value_layer, currency_layer, country_layer])
```

Figure 8: Definition of a neural network with three outputs

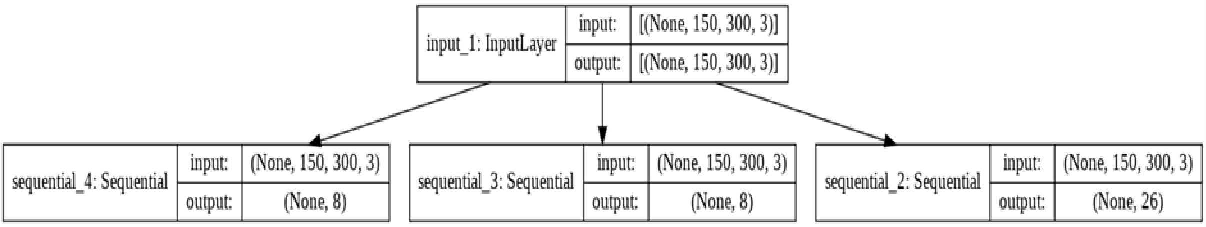


Figure 9: Graph of the neural network with three outputs

The models were trained separately, with consistent distribution of data into training and test sets. However, each model had a varying distribution of data for training and validation. Figures 10, 12, and 14 illustrate accuracy graphs and loss functions for training and validation data of denomination, currency, and country recognition models, respectively. The denominations were recognized with an accuracy of 0.9941, the currency unit with an accuracy of 0.9989, and the country with an accuracy of 0.9641, as shown in Figures 11, 13, and 15. The graphs show when each model begins to overfit, allowing determination of the necessary number of training epochs for individual models. Analyzing the test data reveals high accuracy in denomination and currency recognition, roughly 99%, while country recognition accuracy is lower, approximately 96-97%. This lower precision may be attributed to the larger number of input data categories and characteristics. Specifically, the presence of 2 euro commemorative coins in the dataset complicates identifying their country of origin within the European Union. Separating denomination, currency, and country recognition leads to higher accuracy than traditional networks. In addition, accurately determining denomination and currency is highly likely. The final accuracy, calculated by multiplying branch results, is 0.9574, surpassing traditional convolutional neural network accuracy by 0.0209. This improvement shows promise. Improving flexibility in configuring and training individual models within a three-output convolutional neural network can lead to enhanced results.

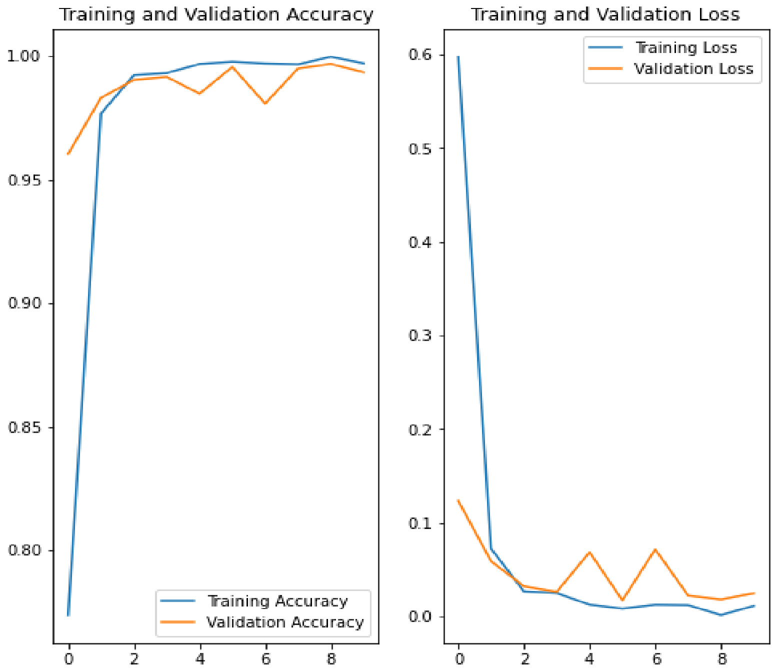
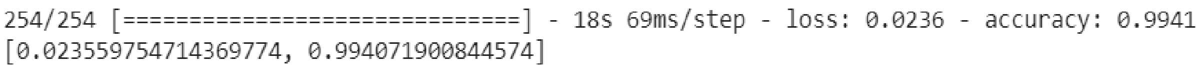


Figure 10: Training graphs for the denomination recognition model



```
254/254 [==============================] - 18s 69ms/step - loss: 0.0236 - accuracy: 0.9941
[0.023559754714369774, 0.994071900844574]
```

Figure 11: Testing results of the denomination recognition model

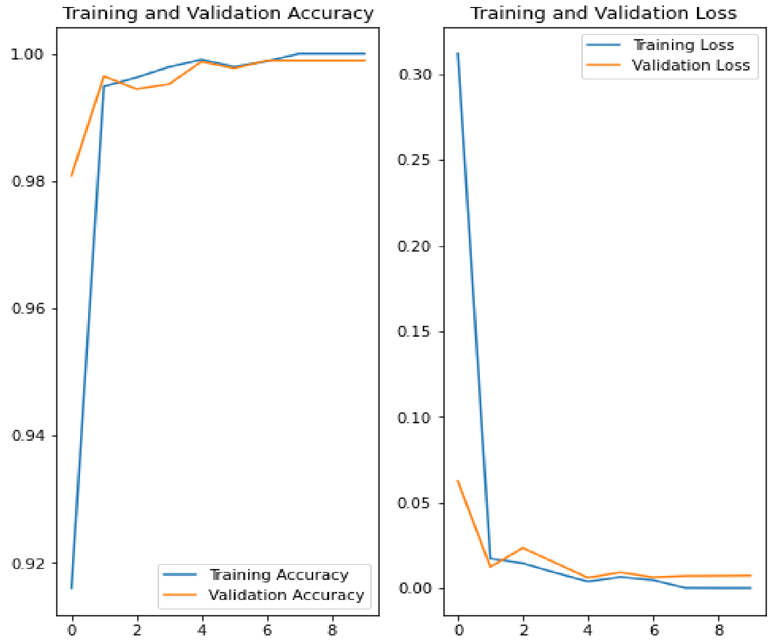
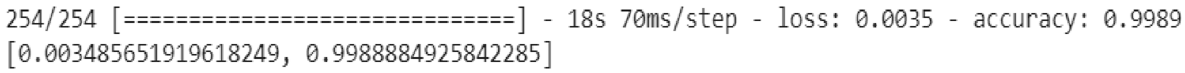
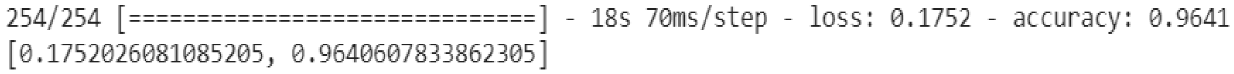


Figure 12: Training graphs of the currency recognition model



```
254/254 [==============================] - 18s 70ms/step - loss: 0.0035 - accuracy: 0.9989
[0.003485651919618249, 0.9988884925842285]
```

Figure 13: Testing results of the currency recognition model



```
254/254 [==============================] - 18s 70ms/step - loss: 0.1752 - accuracy: 0.9641
[0.1752026081085205, 0.9640607833862305]
```

Figure 15: Testing results of the country recognition model

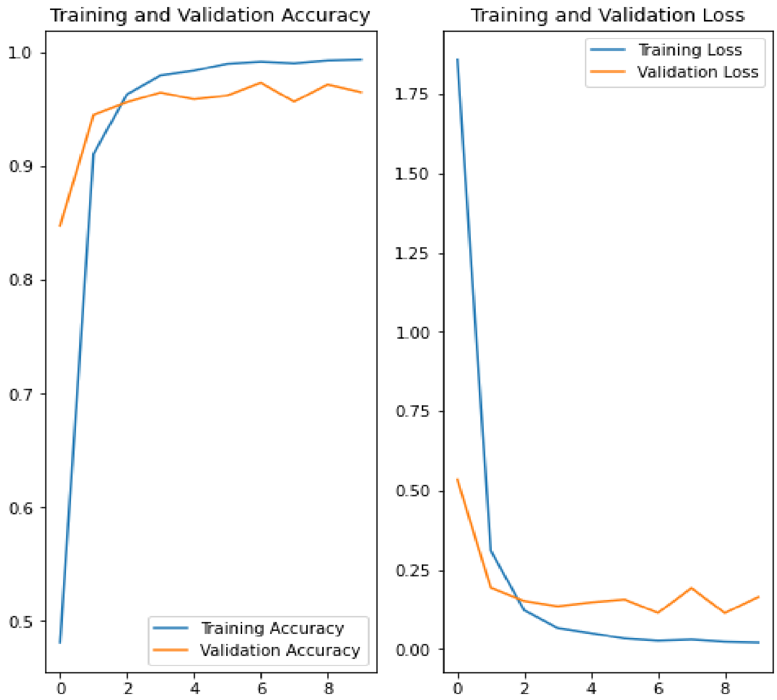


Figure 14: Training graphs of the country recognition model

5. Conclusions

The recognition of coins through images presents a challenging task due to its reliance on distinguishing unrelated characteristics. However, this allows for the implementation of various approaches in classifying images and constructing convolutional neural networks. Our study proposes a traditional approach where each image belongs to a single composite class, identifying its denomination, currency, and country. The neural network yields one output. Additionally, a method was presented that utilizes a neural network with various outputs. Each image feature is classified separately, and the neural network has three outputs, each corresponding to one of them.

Due to the large number of small-volume image classes in the conventional approach, the recognition result was suboptimal at 93-94%. Nevertheless, branching the neural network into three separate branches for each characteristic resulted in significant improvement, with a success rate of 99% for denomination and currency, and 95-96% for the country. A noteworthy advantage of this approach is the option to flexibly adjust and train separate models for each characteristic and incorporate them as layers of separate branches of the neural network. One drawback of this approach is the significant increase in training time and resource use due to the multiple-fold increase in the number of neural network parameters. Nonetheless, training the embedded models separately can decrease the amount of resources utilized concurrently during network training.

Thus, the proposed approach to recognizing circulating coins differs from classical methods using convolutional neural networks. Not only is a narrow classification into one complex class considered, but also the separation of each class into three separate characteristics. This approach solves the problem of complex classes and expands the possibilities of using limited training data, providing more accurate recognition. A branched architecture of a convolutional neural network with multiple outputs is proposed, which takes into account each characteristic separately, corrects the limitations of classical models, and allows to work effectively with small and complex datasets.

6. References

[1] LeCun Y. Convolutional Networks for Images, Speech, and Time-Series / Y. LeCun, Y. Benigo. – 1995. – 14 p.

[2] Mishra, M. Convolutional Neural Networks, Explained|by Mayank Mishra|Towards Data Science. Towards Data Science. 2020. Available online: https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.

[3] Yampolskyi Leonid Stefanovych. Neurotechnologies and neurocomputer systems: a textbook / L.S. Yampolsky, O.I. Lisovychenko, V.V. Oliynyk - K.: Dorado-Druk, 2016. 576 p.

[4] ImageNet [Electronic resource] URL: https://image-net.org/challenges/LSVRC/index.php.

[5] Long-tailed visual recognition with deep models: A methodological survey and evaluation. / Yu Fu, Liuyu Xiang, Guiguang Ding and other // Neurocomputing Volume 509, 14 October 2022, Pages 290-309. https://doi.org/10.1016/j.neucom.2022.08.031

[6] Application of a convolutional neural network with multiple outputs for recognizing circulating coins / E.Y. Vaivala, N.V. Tsiopa, V.S. Shmidke // Collection of scientific papers of the Military Institute of Taras Shevchenko National University of Kyiv - K. : Military Institute of Taras Shevchenko National University of Kyiv, 2021 - № 71. - P. 49-58.

[7] lga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015.

[8] One evolutionary algorithm deceives humans and ten convolutional neural networks trained on ImageNet at image recognition / Ali Osman Topal, Raluca Chitic, Franck Leprévost // Applied Soft Computing, Volume 143, August 2023, 110397. https://doi.org/10.1016/j.asoc.2023.110397

[9] Knowledge driven weights estimation for large-scale few-shot image recognition / Jingjing Chen, Linhai Zhuo, Zhipeng Wei and other // Pattern Recognition, Volume 142, October 2023, 109668. https://doi.org/10.1016/j.epsr.2023.109241

[10] Simonyan K. Very Deep Convolutional Networks for Large-Scale Image Recognition / K. Simonyan, A. Zisserman., 2015. – 14 p. https://arxiv.org/abs/1409.1556

[11] High accuracy keyway angle identification using VGG16-based learning method / Soma Sarker, Sree Nirmillo Biswash Tushar, Heping Chen // Journal of Manufacturing Processes, Volume 98, 28 July 2023, Pages 223-233 https://doi.org/10.1016/j.jmapro.2023.04.019

[12] MANet: A two-stage deep learning method for classification of COVID-19 from Chest X-ray images / Yujia Xu, Hak-Keung Lam, Guangyu Jia // Neurocomputing, Volume 443, 5 July 2021, Pages 96-105 https://doi.org/10.1016/j.neucom.2021.03.034

[13] Multi-level residual network VGGNet for fish species classification / Eko Prasetyo, Nanik Suciati, Chastine Fatichah // Journal of King Saud University - Computer and Information Sciences, Volume 34, Issue 8, Part A, September 2022, Pages 5286-5295. https://doi.org/10.1016/j.jksuci.2021.05.015

[14] Theckedath, D., Sedamkar, R.R. / Detecting Affect States Using VGG16, ResNet50 and SE-ResNet50 Networks. // SN COMPUT. SCI. 1, 79 (2020). https://doi.org/10.1007/s42979-020-0114-9

[15] Deep Residual Learning for Image Recognition / K.He, X. Zhang, S. Ren, J. Sun., 2015. – 12 p.

[16] Wen, L., Li, X. & Gao, L. A transfer convolutional neural network for fault diagnosis based on ResNet-50. Neural Comput & Applic 32, 6111–6124 (2020). https://doi.org/10.1007/s00521-019-04097-w

[17] Deep learning model for defect analysis in industry using casting images / Rupesh Gupta, Vatsala Anand, Sheifali Gupta, Deepika Koundal // Expert Systems with Applications Volume 232, 1 December 2023, 120758. https://doi.org/10.1016/j.eswa.2023.120758

[18] Mascarenhas, Sheldon and Mukul l Agarwal. “A comparison between VGG16, VGG19 and ResNet50 architecture frameworks for Image Classification.” 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON) 1 (2021): 96-99.

[19] Siva Satya Sreedhar, P. and N. Nandhagopal. “Classification Similarity Network Model for Image Fusion Using Resnet50 and GoogLeNet.” Intelligent Automation & Soft Computing, vol.31 (2022), no.3, Pages 1331-1344.

[20] Bozinovski S. Reminder of the First Paper on Transfer Learning in Neural Networks / Stevo Bozinovski., 2020. – 12 p.

[21] Module: tf | TensorFlow v2.14.0 URL: https://www.tensorflow.org/api_docs/python/tf?hl=en.

[22] Keras API reference URL: https://keras.io/api/.

[23] D. Xu, Y. Shi, I. W. Tsang, Y. -S. Ong, C. Gong and X. Shen, "Survey on Multi-Output Learning," in IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 7, pp. 2409-2429, July 2020, doi: 10.1109/TNNLS.2019.2945133.

[24] A survey on multi-output regression / H.Borchani, G. Varando, C. Bielza, P. Larranaga. // WIREs Data Mining Knowl Discov 2015, 5:216–233. doi: 10.1002/widm.1157