The Effectiveness of Deep Learning Algorithms in Sign Road Recognition Problem

Mohammad Ali Sheikh Soltani[1], Marat Nurtas[1,2], Bazargul Matkerim[3] and Nurgul Nalgozhina[1]

[1]*International Information Technology University, Manas St. 34/1, Almaty, 050040, Kazakhstan*
[2]*Institute of Ionosphere, Gardening community IONOSPHERE 117, Almaty, 050020, Kazakhstan*
[3]*Al-Farabi Kazakh National University, al-Farabi Avenue 71, Almaty, 050040, Kazakhstan*

Abstract

This research investigates road sign recognition using deep learning methods, comparing them to traditional approaches and emphasizing the potential of simplified convolutional neural networks. Evaluations were conducted on a diverse dataset, encompassing images from different cameras and varying weather conditions. The study aimed to assess the effectiveness of various neural network architectures in road sign recognition. Results demonstrate that, in specific scenarios, simplified convolutional neural networks surpass conventional methods, providing more efficient solutions. This study highlights the practicality of such networks, particularly when addressing diverse data sources and conditions. These results can be applied to solving the problem of recognizing road signs in practice in systems related to the movement of vehicles. The article also provides a comparative analysis of some of the main methods used to classify objects with a created convolutional neural network, which allows researchers to choose algorithms more suitable for their purposes in the future. Future research could explore the optimization and fine-tuning of these architectures for enhanced practical performance.

Keywords

Convolutional Neural Network, VGG16, Residual Neural Network, MobileNet, Traffic-sign recognition

1. Introduction

Some time ago, cars with autopilots seemed like a fantasy and technology of the distant future, but now it is a promising development direction to become a reality. Management without constant human involvement using special tools such as LIDAR [1] and neural networks [2] has already been tested, and many manufacturers have released their models with autopilots.

Under the concept of autopilot, groups of technologies are combined to help remove part of the load from a person when driving a vehicle. The autopilot helps maintain the course while driving. It monitors the specified safety characteristics of the movement route and complies with the selected operating limits. At the same time, the person remains in the driver's seat and controls the movement because he is responsible for everything that happens on the road. Based on this approach, it can be noted that several levels of cars with autopilot will differ in their degree of independence according to the Society of Automotive Engineers classification [3].

Currently, there is no widespread Level 5 autopilot. However, because many car manufacturers are developing a full-fledged autopilot, we should expect that many cars with similar systems will soon appear, and they will become commonplace.

Recognizing road signs is essential for driver-assistance systems in modern vehicles. Acknowledging the distance between objects on the road, road signs, and markings is crucial for safe driving. If specialized sensors are sufficient to recognize the distance, recognizing road signs on the go requires more complex algorithms and machine learning [4] methods.

DTESI 2023: Proceedings of the 8th International Conference on Digital Technologies in Education, Science and Industry, December 06–07, 2023, Almaty, Kazakhstan

sheikhsoltanimohammadali@gmail.com (M. Soltani); m.nurtas@iitu.edu.kz (M. Nurtas); n.nalgozhina@iitu.edu.kz (N. Nalgozhina); bazargul.matkerim@gmail.com (B. Matkerim)

0000-0003-4351-0185 (M. Nurtas); 0000-0002-5336-687X (B. Matkerim); 0000-0003-0254-8670 (N. Nalgozhina)

In addition, the introduction of autonomous cars in urban environments is fraught with several difficulties. Navigation in densely populated areas requires high accuracy and adaptability because the system must quickly respond to sudden changes in traffic flow, pedestrian crossings, and unpredictable road conditions. Ensuring seamless integration with existing transport infrastructure while ensuring passenger safety remains critical for researchers and engineers.

In addition, the issue of cybersecurity is increasing in discussions on autonomous vehicles. As interconnected systems are integrated and vast amounts of confidential data are exchanged, vulnerability to cyber threats has become a key factor. Ensuring the security of a car's software against hacking or malicious intrusion is becoming a priority for developers.

Deep learning [5] plays a significant role in the creation of autopilots. They allow the processing of a considerable amount of data and make decisions based on the data regarding further actions on the road.

Traditional methods [6] for recognizing road signs involve manual extraction of signs and classification, which takes time and requires considerable human effort. In recent years, deep learning algorithms [7] have shown great promise in computer vision [8], including in recognizing road signs. In this article, we propose a comparison of various methods for recognizing road signs using deep learning algorithms. In the field of application of neural networks, in addition to accuracy [9], recognition speed also plays a unique role, because the system must immediately make decisions regarding recognized road signs and correct driving.

This study used deep learning algorithms to compare various methods for recognizing road signs in images. This study was conducted on a publicly available dataset that considered images of road signs under different weather conditions, allowing the results to be closer to practical tasks. In addition, the results of this study will be helpful when choosing different architectures [10] depending on the image recognition sphere.

2. Dataset selection and data pre-processing

A dataset [11] taken from open sources was used in this study. The dataset used in this study was a set of more than 50,000 images of German road signs.

Figure 1: Examples of images from the German Traffic Sign Recognition Benchmark dataset [12]

In addition to this dataset, three additional datasets were analyzed. The following disadvantages were identified during the analysis:

1. LISA Traffic Sign Dataset [13]: Some images may contain noise, and the dataset size is smaller than others.
2. BelgiumTS [14]: Some classes may be underrepresented, leading to retraining problems.
3. Russian Traffic Sign Dataset [15]: Some images may be poor quality.

The German Traffic Sign Recognition Benchmark is an optimal dataset based on the above factors. Its advantages include the availability of a large number of high-quality images. In addition, the images were made in various lighting and weather conditions, allowing the model trained on these data to be closer to actual practical tasks. This dataset was chosen because of a large number of images in different weather conditions, which helps to implement created models to different datasets and achieve similar results. This dataset contains 43 different classes of road signs:

1. Speed limit (20km/h);
2. Speed limit (30km/h);
3. Speed limit (50km/h);
4. Speed limit (60km/h);
5. Speed limit (70km/h);
6. Speed limit (80km/h);
7. End of speed limit (80km/h);
8. Speed limit (100km/h);
9. Speed limit (120km/h);
10. No passing;
11. No passing veh over 3.5 tons;
12. Right-of-way at intersection;
13. Priority road;
14. Yield;
15. Stop;
16. No vehicles;
17. Veh > 3.5 tons prohibited;
18. No entry;
19. General caution;
20. Dangerous curve left;
21. Dangerous curve right;
22. Double curve;
23. Bumpy road;
24. Slippery road;
25. Road narrows on the right;
26. Road work;
27. Traffic signals;
28. Pedestrians;
29. Children crossing;
30. Bicycles crossing;
31. Beware of ice/snow;
32. Wild animals crossing;
33. End speed + passing limits;
34. Turn right ahead;
35. Turn left ahead;
36. Ahead only;
37. Go straight or right;

38. Go straight or left;
39. Keep right;
40. Keep left;
41. Roundabout mandatory;
42. End of no passing;
43. End no passing veh > 3.5 tons.

These classes were universal. The results obtained after training the deep learning model on this dataset can be used not only in Germany but also in other countries around the world. Although this dataset is considered the best for use in the field, it also has some drawbacks. For example, in this dataset, data were unevenly distributed across classes.

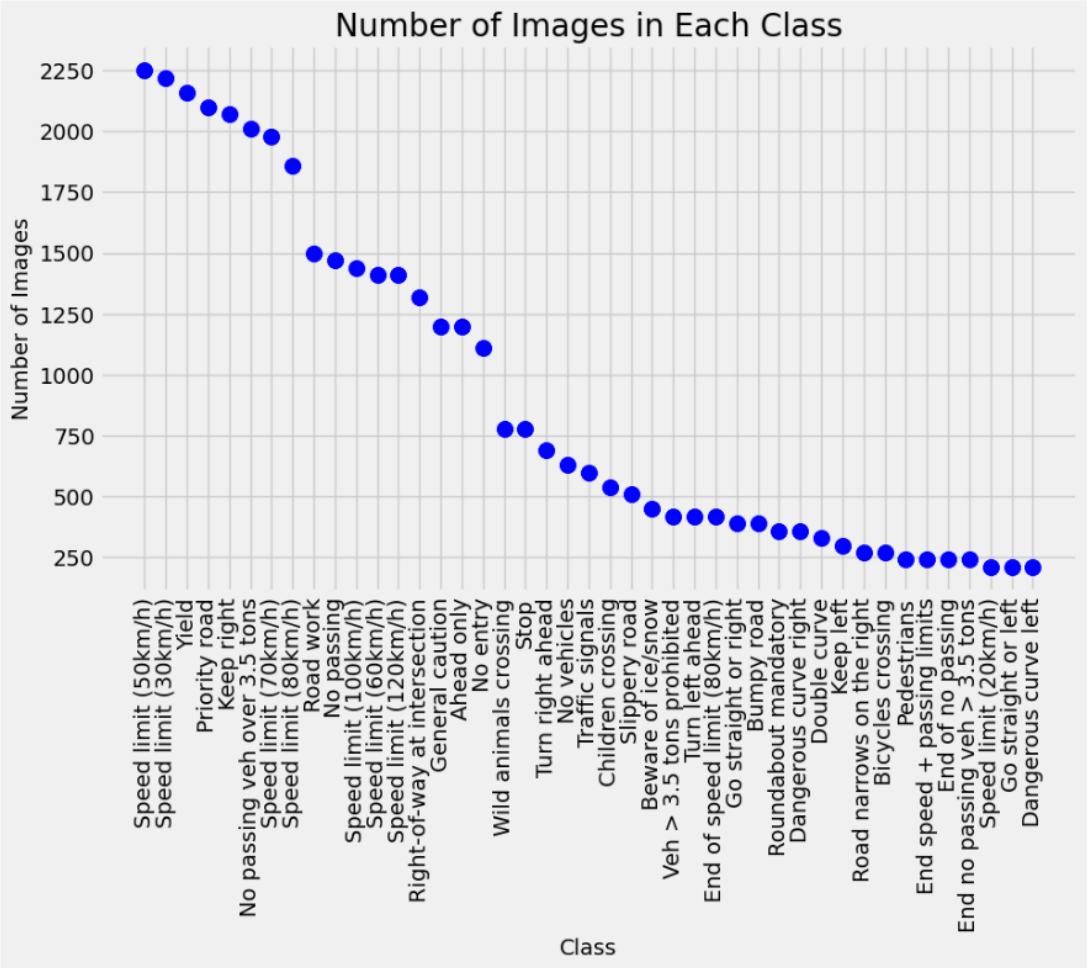

Figure 2: Plotted distribution of images by sign class in our version of German Traffic Sign Recognition Benchmark

3. Algorithms and architecture of neural networks

The next step after collecting [16], processing [17], and analyzing the dataset is to choose the optimal machine-learning model for our task. Because we faced the analytical task of comparing various machine-learning methods, we chose the most common methods widely used to solve such problems. We used such libraries as tensorflow [18] and Keras [19] to create models and sklearn [20] to import different metrics.

The first method is a convolutional neural network [21]. Convolutional neural networks are essential tools for computer vision tasks. Such networks allow the classification of [22] images

and find various objects. A CNN consists of several layers. The more layers, the more powerful the architecture and the better the neural network training. The main elements of a convolutional as follows:

- convolutional layer [23];
- pooling;
- batch normalization;
- fully connected layer.

The neural network performs operations on each layer using the submitted image. On the convolutional and pulling layers, the neural network tries to retain important data for itself, such as the edges of objects. The normalization layer accelerates the model learning and improves the generalization ability of the algorithm. The fully connected layer, in turn, is responsible for classification and decision-making based on the extracted features. Each neural network had its own architecture and configuration. There is an input layer where the input signal is fed, an output layer from which the result of the neural network is removed, and hidden layers between them. During training, the neural network learns to recognize small details such as the faces of some objects. Subsequently, from such faces, more complex patterns are formed, from which, after several iterations, the object as a whole is recognized.

The architecture of the neural network affects all further studies of the algorithm. When setting each task, we must select an architecture that corresponds to the task and allows us to achieve the most optimal results in terms of the time taken and the accuracy of the solution. In the learning process, the loss function is continuously optimized, but different architectures may require different amounts of time for the learning process. In addition, in the process of training each model, we must ensure that there is no retraining. To avoid overfitting, there are different regularization methods, such as L1 and L1 regularization [24], and dropouts.

We have developed a convolutional neural network to solve the problem of recognition and classification of road signs. In this convolutional neural network, an architecture of several layers was used, each of which performs certain operations with input data:

1. Conv2D Layer: The first Conv2D layer with filters=16 and kernel_size = (10.10) applied 16 convolutional filters of size 10 ×10 to the input images. It uses the rectified linear unit (ReLU) activation function to introduce nonlinearity.
2. MaxPool2D Layer: The next MaxPool2D layer with the pool_size = (2, 2) parameter performs the maximum pooling operation with a 2 × 2 window, thereby reducing the dimension of the feature space.
3. BatchNormalization Layer (batch normalization layer): After each convolutional layer, a batch normalization layer is applied, normalizing the input data by centering and scaling, contributing to the stability and acceleration of learning.
4. Dropout Layer (Regularization Layer): A dropout layer with a parameter rate of 0.25 is used to randomly reset the input units with a given probability, which helps to reduce the retraining of the model.
5. Flatten Layer (Alignment Layer): Aligning multidimensional data into a one-dimensional vector and preparing them for feeding to fully connected layers.
6. Dense Layers: Fully connected Dense layers with the activation='relu' parameter perform transformation operations and apply the ReLU activation function to the received data.
7. Softmax Layer: The last Dense layer with the activation='softmax' parameter applies the Softmax activation function, which converts the output values of neurons into probabilities of belonging to each of the 43 classes.

Each layer in this neural network architecture plays a vital role in processing and extracting image features, which allows the network to effectively solve the tasks of classifying road signs.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 41, 41, 16)        4816

 conv2d_5 (Conv2D)           (None, 32, 32, 32)        51232

 max_pooling2d_2 (MaxPoolin  (None, 16, 16, 32)        0
 g2D)

 batch_normalization_3 (Bat  (None, 16, 16, 32)        128
 chNormalization)

 conv2d_6 (Conv2D)           (None, 12, 12, 64)        51264

 conv2d_7 (Conv2D)           (None, 8, 8, 64)          102464

 max_pooling2d_3 (MaxPoolin  (None, 4, 4, 64)          0
 g2D)

 batch_normalization_4 (Bat  (None, 4, 4, 64)          256
 chNormalization)

 dropout_2 (Dropout)         (None, 4, 4, 64)          0

 flatten_1 (Flatten)         (None, 1024)              0

 dense_2 (Dense)             (None, 512)               524800

 batch_normalization_5 (Bat  (None, 512)               2048
 chNormalization)

 dropout_3 (Dropout)         (None, 512)               0

 dense_3 (Dense)             (None, 43)                22059

=================================================================
Total params: 759067 (2.90 MB)
Trainable params: 757851 (2.89 MB)
Non-trainable params: 1216 (4.75 KB)
_________________________________________________________________
```

Figure 3: Architecture of the developed convolutional neural network

The second algorithm was a neural network with the MobileNet [25] architecture. The convolutional part of the network consists of one ordinary convolutional layer with a 3 × 3 convolution at the beginning and 13 blocks with a gradually increasing number of filters and a decreasing spatial dimension of the tensor. The two hyperparameters of the MobileNet architecture are the width and depth multipliers.

- The width multiplier is responsible for the number of channels in each layer.
- The resolution multiplier is responsible for the spatial dimensions of the input tensors.

Both parameters allowed the size of the network to be varied. By reducing the width and resolution multipliers, we reduced the recognition accuracy, but at the same time, we increased the speed and reduced the memory consumed.

The MobileNet architecture model consists of the following layers:

1. Conv2D Layer: Conv2D with parameters filters=32 and kernel_size = (3,3). This layer performs a convolution operation with filters to extract the features from the input data.

2. BatchNormalization Layer (batch normalization layer): After the convolutional layer, a batch normalization layer is applied, which normalizes the data on the batch, providing stability and acceleration of learning.
3. ReLU Layer: The ReLU activation layer is applied after normalization to introduce nonlinearity, allowing the model to learn complex dependencies in the data.
4. DepthwiseConv2D Layer (Deep Convolution Layer): The DepthwiseConv2D layer with the parameters kernel_size = (3,3) and strides=1 performs deep convolution with a core size of 3 × 3 and a step of 1. It works separately for each input channel and is an effective tool for extracting local patterns.
5. Dense Layer (fully connected layer): At the end of the model, fully connected dense layers were added with the Softmax activation function to perform classification into 43 classes.
6. The model is compiled with the loss function 'categorical_cross entropy,' the optimizer 'adam,' and the metric 'accuracy' for monitoring the learning process.

This architecture is a deep convolutional neural network capable of extracting complex spatial features of images and classifying them into 43 classes using the categorical cross-entropy loss function and the Adam optimizer.

The third algorithm we chose to recognize road signs was the VGG-19 architecture [26]. The VGG is a convolutional neural network with a depth of 19 layers. It was built and trained by K. Simonyan and A. Zisserman at Oxford University in 2014. The VGG-19 network was trained using more than one million images from the ImageNet database. She was trained on color images that measured 224 × 224 pixels. For our task, we used a neural network with the following architecture:

1. VGG19 Layer (VGG-19 Layer): A pre-trained VGG-19 model loaded with weights trained on the ImageNet dataset was used. This layer allows us to extract more complex image features owing to a deep architecture with multiple convolutional and pooling layers.
2. BatchNormalization Layer: The BatchNormalization layer is used to normalize batch data, which helps accelerate learning and increase the stability of the model.
3. Flatten Layer: The Flatten layer converts multidimensional data into a one-dimensional vector, preparing them for feeding to fully connected layers.
4. Dense Layers: Fully connected dense layers with sigmoid and softmax activation functions perform linear affine transformation and application of corresponding nonlinearities to the obtained data. Sigmoid activation introduces nonlinearity, and Softmax activation in the final layer predicts the probabilities of belonging to each of the 43 classes.

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 1, 1, 512)         20024384

 batch_normalization_3 (Bat  (None, 1, 1, 512)         2048
 chNormalization)

 flatten_1 (Flatten)         (None, 512)               0

 dense_2 (Dense)             (None, 512)               262656

 dense_3 (Dense)             (None, 43)                22059

=================================================================
Total params: 20311147 (77.48 MB)
Trainable params: 20310123 (77.48 MB)
Non-trainable params: 1024 (4.00 KB)
_________________________________________________________________
```

Figure 4: Architecture of the developed VGG19 neural network

4. Results

The results of using different architectures on the same dataset revealed the optimal ways to solve this problem. The recognition of road signs and all systems related to the autopilot of vehicles are specific tasks of machine learning in general, and computer vision in particular. In addition to their high accuracy, these algorithms should show acceptable time indicators. They cannot always be run on robust systems and should not be so demanding in terms of performance.

The first algorithm was run for 15 epochs, and the calculation time was 18 min 17 s. At the end of the second epoch, a 97% accuracy was obtained; however, the accuracy remained at 86%. After completing the training, the model showed an accuracy of 99% for the test data 4% for val_accurasy 94%. The model showed good accuracy of 79.7% for the test dataset.

Figure 5: Sample of predictions made by convolutional neural network

The algorithm with the VGG19 architecture was trained for four epochs and its performance was improved from 30% accuracy and 8% val_acuracy to 98% accuracy and 87% val_acuracy, respectively. However, despite these high indicators, the algorithm performed worse on the test dataset, with an accuracy of 39%. Although the algorithm was trained for only four epochs, the training time was only 53 min.

The MobileNet architecture was trained for 30 epochs over 36 min. The results of the first epoch were 25% accuracy and 5% val_accuracy, but by the last epoch, they improved to 98% accuracy and 78% val_acuracy, which is lower than that of algorithms with other architectures. The test data showed that the algorithm was similar to VGG19 with an accuracy of 52%.

5. Conclusion

The study of different architectures using the same data allowed us to identify the most effective systems for recognizing road signs. One of the most important factors was the short training time

of the first model. Despite the shortest time, it showed good results not only on the training data but also on the test dataset.

The VGG19 and MobileNet architectures showed good results during training but poor results when used on a test dataset. One of the factors that influences this is retraining. Such models may be more demanding in terms of the amount of required data. Because of the large number of parameters, they are prone to retraining when there is a lack of data. Retraining is also possible owing to the poor generalization ability of the model. This is primarily influenced by parameters such as the learning rate or choice of the optimization algorithm.

In this paper, we show the importance of choosing the right machine learning algorithms not only for character recognition but also for computer vision in general. Also in this article we conduct a comparative analysis of the created convolutional neural network with the most commonly used neural network architectures. This will allow the introduction of neural networks, with a more suitable architecture, for the specifics of the task into autopilot systems.

In future work, we plan to study other systems suitable for recognizing images of road signs and to check the impact of various types of data on the performance of the proven algorithms.

6. References

[1] Y. Li, A. Wei Yu, T. Meng, B. Caine, J. Ngiam, D. Peng, J. Shen, Y. Lu, D. Zhou, Q. Le, A. Yuille, M. Tan. (2022), DeepFusion: Lidar-Camera Deep Fusion for Multi-Modal 3D Object Detection" IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 17161-17170, doi: 10.1109/CVPR52688.2022.01667.

[2] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities (1982). doi: 10.1073/pnas.79.8.2554.

[3] Society of Automotive Engineers, Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles J3016_202104, 2021. URL: https://www.sae.org/standards/content/j3016_202104/.

[4] Ethem A. (2020). Introduction to Machine Learning (Fourth ed.).

[5] Arnold L., Rebecchi S., Chevallier S., Paugam-Moisy H. (2011). An Introduction to Deep Learning. Statistical Foundations of Data Science.

[6] J. Stallkamp, M. Schlipsing, J. Salmen, C. Igel. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition, Neural Networks, V.32, pp. 323-332, doi: 10.1016/j.neunet.2012.02.016.

[7] Kang N., Introducing Deep Learning and Neural Networks, 2017, https://towardsdatascience.com/introducing-deep-learning-and-neural-networks-deep-learning-for-rookies-1-bd68f9cf5883.

[8] Mihajlovic I., Everything You Ever Wanted To Know About Computer Vision, 2019, https://towardsdatascience.com/everything-you-ever-wanted-to-know-about-computer-vision-heres-a-look-why-it-s-so-awesome-e8a58dfb641e.

[9] Mishra A, Metrics to Evaluate your Machine Learning Algorithm, 2018, https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234.

[10] Culurciello E., Neural Network Architectures, 2017, https://towardsdatascience.com/neural-network-architectures-156e5bad51ba.

[11] Kaggle Team, GTSRB - German Traffic Sign Recognition Benchmark, 2018. URL: https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign.

[12] V7labs core team, GTSRB, 2023. URL: https://www.v7labs.com/open-datasets/gtsrb.

[13] Kaggle Team, LISA Traffic Light Dataset, 2015, https://www.kaggle.com/datasets/mbornoe/lisa-traffic-light-dataset.

[14] Kaggle Team, BelgiumTS Dataset, 2017, https://www.kaggle.com/datasets/mahadevkonar/belgiumts-dataset.

[15] Konushin A., Shakhuro V., Russian traffic sign images dataset (RTSD), https://graphics.cs.msu.ru/projects/traffic-sign-recognition.html.
[16] Yuji Roh, Geon Heo, Steven Euijong Whang, Senior Member. (2018). A Survey on Data Collection for Machine Learning A Big Data - AI Integration Perspective, doi: 10.48550/arXiv.1811.03402.
[17] Khalid K. Al-jabery, Tayo Obafemi-Ajayi, Gayla R. Olbricht, Donald C. Wunsch II. (2020). Computational Learning Approaches to Data Analytics in Biomedical Applications, Academic Press, doi:10.1016/B978-0-12-814482-4.00002-4.
[18] Sumit Saha, A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way, 2018, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.
[19] TensorFlow core team, TensorFlow, 2023. URL: https://www.tensorflow.org/.
[20] Keras core team, Introducing Keras Core: Keras for TensorFlow, JAX, and PyTorch, 2023. URL: https://keras.io/keras_core/announcement/.
[21] Scikit Learn Core Team, scikit-learn Machine Learning in Python, 2023. URL: https://scikit-learn.org/stable/index.html.
[22] FutureAnalytica core team, Classification in Data Science, 2022, https://medium.com/@futureanalytica/classification-in-data-science-7e7e9e6e8c39.
[23] Mayank Mishra, Convolutional Neural Networks, Explained, 2020, https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939.
[24] Anuja Nagpal, L1 and L2 Regularization Methods, 2017, https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c.
[25] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. (2017). Mobile Nets: Efficient Convolutional Neural Networks for Mobile Vision Applications, doi: 10.48550/arXiv.1704.04861.
[26] Karen Simonyan, Andrew Zisserman. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition, doi: 10.48550/arXiv.1409.1556.