Gradus Vol 11, No 1 (2024)
ISSN 2064-8014

SOLVING A CLASSIFICATION PROBLEM USING TRANSFER LEARNING ON SMALL IMAGE DATASETS

Rajmund Drenyovszki 0000-0002-9462-2729 1*

[1] Department of Information Technology, GAMF Faculty of Engineering and Computer Science,
John von Neumann University, Hungary

https://doi.org/10.47833/2024.1.CSC.011

Keywords:
Classification
Deep Learning
CNN
VGG16
Transfer Learning

Article history:

Received	11 December 2023
Revised	24 March 2024
Accepted	27 March 2024

Abstract
In this study, we explore the efficiency of transfer learning in training convolutional neural networks (CNNs) for image classification tasks, focusing on a dataset with limited size. Utilizing the VGG16 model, we investigate the impact of varying the number of trainable layers on classification accuracy. Our specific classification challenge involves distinguishing between images of cars with open and closed hoods. The study demonstrates that employing transfer learning enables the pretrained VGG16 model to achieve over 97% accuracy in this binary classification task, even with a training set of just 1000 image samples. This research was conducted solely by the undersigned, who also represents the sole author of this paper.

1 Introduction

Machine learning, a subset of artificial intelligence (AI), involves algorithms capable of autonomously learning and evolving through the analysis of data, without explicit programming instructions. This process enables machines to independently recognize patterns and make decisions, continuously enhancing their performance over time. Among machine learning methods, deep learning is notable, particularly employing deep neural networks such as Convolutional Neural Networks (CNNs). These networks utilize multi-layered, complex structures especially effective in analyzing and processing visual data, thus capable of learning intricate features and patterns. In a broader sense, artificial intelligence encompasses computer systems with capabilities mimicking human intelligence, such as learning, problem-solving, and decision-making. Machine learning, and specifically deep learning techniques like CNNs, are pivotal tools in AI applications, allowing machines to independently evolve and adapt.

The application of Transfer Learning in CNNs entails using a model previously trained on a similar problem as a foundation for developing a new model, designated for a distinct but related task. This method facilitates the transfer of previously acquired knowledge to the new model, thereby reducing the data requirements and computational resources needed for training the new model. This concept aligns with foundational works in the field, such as [1] study on how transferable are features in deep neural networks and the seminal work [2] on backpropagation in neural networks, which laid the groundwork for advancements in deep learning and CNNs.

In this research, we examine the implementation of the VGG16 model through transfer learning for a binary classification task within an automotive repair workshop setting. Our objective is to accurately identify whether cars in images are displayed with their hoods open or closed. This determination provides insights into the various stages of repair or maintenance activities in the workshop. By adapting the VGG-16 model, renowned for its efficient feature extraction capabilities, we tailor our approach to meet the specific visual analysis needs of the automotive repair environment.

* Corresponding author.
E-mail cím: drenyovszki.rajmund@nje.hu

2 Literature review

In the realm of computer vision, Convolutional Neural Networks (CNNs) like VGG16 have shown remarkable efficiency, particularly when applied to transfer learning tasks. This summary highlights key models and their applications in transfer learning.

1. VGG16: Developed by Simonyan and Zisserman, VGG16 is renowned for its simplicity and depth, with 16 convolutional layers. It's widely used in transfer learning due to its excellent feature extraction capabilities, particularly in image classification tasks [3].

2. ResNet: The Residual Network (ResNet), introduced by He et al., is another significant architecture in deep learning. With its unique residual blocks, ResNet allows training of substantially deeper networks. Its variants, especially ResNet-50 and ResNet-101, have been instrumental in transfer learning applications [4].

3. Inception (GoogLeNet): Szegedy et al. introduced Inception, a network characterized by its inception modules that allow it to learn multi-level feature representations. It has been effectively used in transfer learning for tasks requiring recognition of complex patterns [5].

4. Xception: As a variant of Inception, Xception, proposed by Chollet, utilizes depthwise separable convolutions and has been shown to outperform Inception in certain scenarios. It is particularly effective in scenarios where data is limited and diverse [6].

5. MobileNet: Designed by Howard et al., MobileNet is optimized for mobile and edge devices, balancing efficiency and accuracy. Its lightweight architecture makes it a popular choice for transfer learning in real-time applications [7].

Each of these models has contributed to the advancement of transfer learning, providing efficient and robust solutions for various tasks in computer vision and beyond.

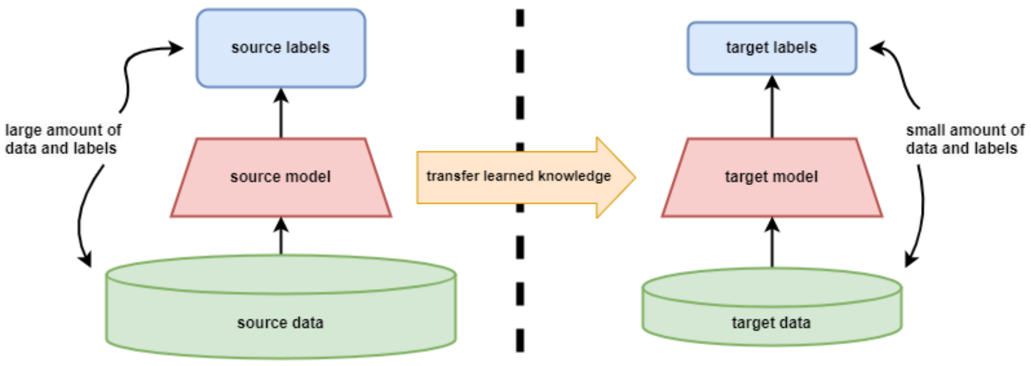
source labels
large amount of data and labels
source model
source data
transfer learned knowledge
target labels
small amount of data and labels
target model
target data

Figure 1. Idea of Transfer Learning

Transfer learning (Figure 1), a significant concept in the field of machine learning, pertains to the method where a model developed for a specific task is reused as the starting point for a model on a second task. This approach is especially pivotal in scenarios where labeled data for the target task is scarce. At its core, transfer learning involves transferring knowledge from a previously trained model to a new model. This is particularly useful in deep learning, where substantial computational resources and large datasets are often required for training from scratch [8]. One of the primary benefits of transfer learning is its efficiency. By utilizing pre-trained models, significant reductions in computational time and resources can be achieved. This makes advanced machine learning models more accessible and cost-effective, especially for small-scale applications or projects with limited resources [9].

Transfer learning has found applications across various domains, from image and speech recognition to natural language processing. The flexibility of transfer learning makes it a valuable tool in applying sophisticated models to new, but related, problems [10]. While transfer learning offers numerous advantages, it also presents challenges, primarily in adapting a model trained on one task to a different, albeit related, task. Techniques such as fine-tuning and feature extraction are commonly used to address these challenges, adapting the pre-trained model to the new task with minimal data [11]. The field of transfer learning continues to evolve, with ongoing research exploring more efficient and effective methods to transfer knowledge across tasks and domains. This is particularly relevant in the era of big data, where the ability to leverage existing datasets and models can significantly accelerate the development of new and innovative applications [12].

Without claiming to be exhaustive, we list below a few studies on the use of VGG16 transfer learning. The study [13] addresses the challenge of image classification with the limitation of limited training samples for each category. It utilizes a dataset comprising images of dogs and cats, aiming to develop a heuristic and resilient model capable of accurately sorting images into distinct categories for dogs and cats. The paper [14] explores the use of the VGG16 architecture, enhanced by transfer learning, to automate the classification of tobacco leaves damaged by pests. Demonstrating high accuracy, this method efficiently differentiates between healthy and affected leaves, significantly improving the quality control process in tobacco leaf sorting. The research in [15] improves fruit ripeness classification by employing a modified VGG16 model with transfer learning, replacing its top layer with a Multilayer Perceptron (MLP) that includes regularization methods to prevent overfitting. The approach significantly outperformed traditional methods, particularly using Dropout in the MLP block, which led to an 18.42% increase in accuracy. This highlights the effectiveness of deep learning with transfer learning over conventional feature extraction techniques in detecting fruit ripeness.

3 Problem description

Our job in an auto repair shop is to determine each work process, including whether the hoods of the cars being repaired are open or closed. Below (Figure 2) are pictures taken in the workshop. Video recordings and pictures are taken of the cars from the front view, from a height of 3 meters, which are then used to analyze the processes.

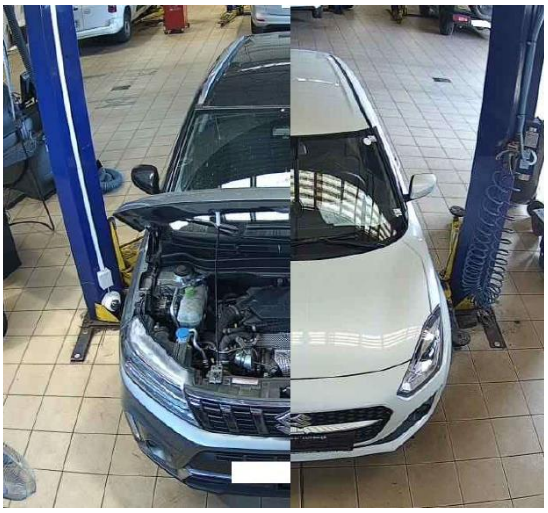

Figure 2. Pictures of cars with open/closed hood

Currently, the domain lacks a pre-trained model specifically designed to differentiate between vehicles with open and closed hoods. However, for general vehicle detection tasks, the YOLO (You Only Look Once) model represents a viable option. The YOLOv8 framework includes a variety of models specialized in detection, segmentation, and pose estimation, all of which have been pre-trained on the COCO dataset. Additionally, the framework offers classification models that have been pre-trained on the ImageNet dataset. Furthermore, a tracking mode is integrated into all detection, segmentation, and pose estimation models.

4 Methods

This section delineates the dataset employed for transfer learning, encompassing the images utilized for the purposes of training, validation, and testing. Additionally, it elaborates on the adoption of the VGG16 model, specifically addressing the application of transfer learning to each of its constituent layers.

Table 1. Source and quantity of images in the Dataset used for training, validation and testing

	Source of images	*Quantity of images*
Training	Internet	500, 500
Validation	Internet	150, 150
Testing	Workshop	150, 150

4.1 Dataset

Table 1 summarizes the dataset properties. For training (500 samples per class) and validation (150 samples per class), we utilized images of cars with both closed and open hoods sourced from the internet. These images were subsequently curated to assemble the highest quality dataset, with sample representations shown in Figure 3. For testing, exclusively workshop images (Figure 2) were employed (150 samples per class). In the training of deep neural networks, a critical factor is the model's generalization ability. Equally important is the prevention of overfitting, which refers to a scenario where the network excessively learns from the specific training images at the expense of its ability to generalize to new data. To ensure a broad applicability of our findings, we used images from the repair shop as our testing dataset. The positioning of vehicles was standardized to maintain a consistent region of interest for the object detection process. Images were used in their original form without any preprocessing.

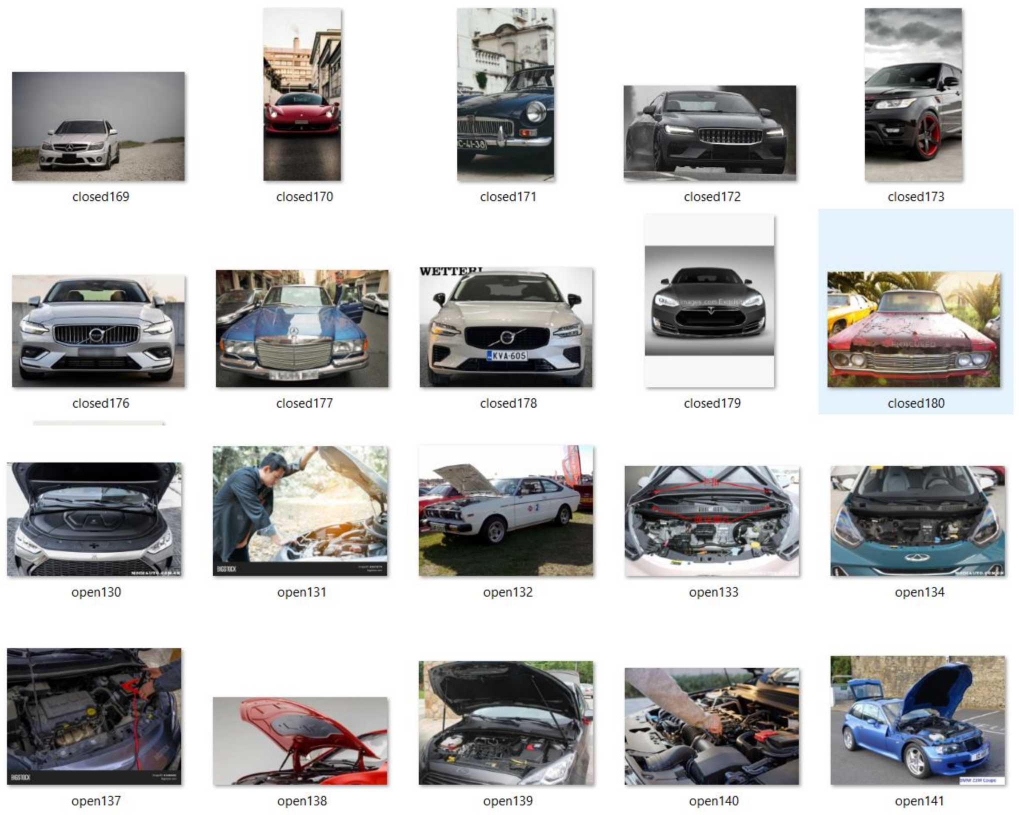

Figure 3. Example images of train and validation data

4.2 VGG-16 model

The VGG16 model (Figure 4), a convolutional neural network architecture, was introduced by K. Simonyan and A. Zisserman of the University of Oxford in their seminal work [3]. This model is distinguished by its performance, achieving a top-5 test accuracy of 92.7% on the ImageNet dataset, which comprises over 14 million images across 1000 different classes. Notably, VGG16 was a prominent entry in the ILSVRC-2014 competition. Its design advances beyond the AlexNet architecture by substituting large kernel-sized filters (specifically, 11 and 5 in the initial two convolutional layers) with successive layers of 3×3 kernel-sized filters. The training of VGG16 spanned several weeks, utilizing NVIDIA Titan Black GPUs for computational power.

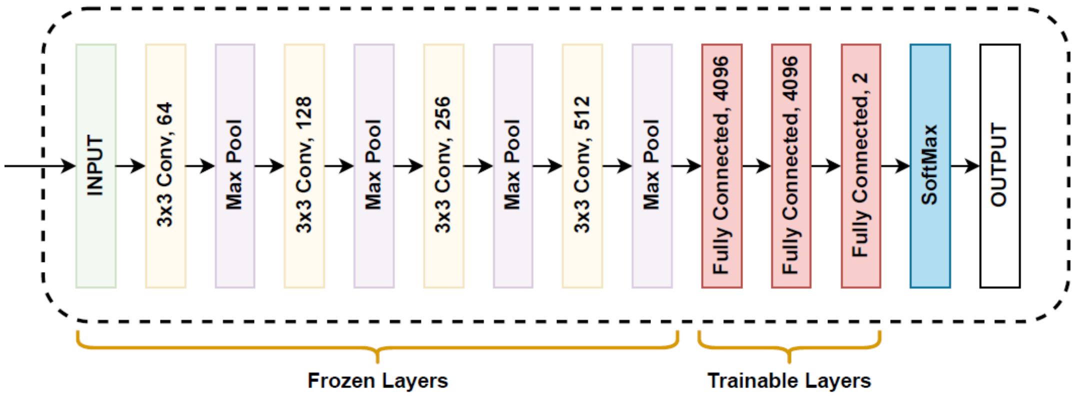

Figure 4. VGG-16 model architecture

ImageNet itself is a substantial dataset containing over 15 million labeled high-resolution images, categorized into approximately 22,000 groups. These images were sourced from the internet and annotated by individuals through Amazon's Mechanical Turk [16], a platform for crowd-sourced labeling tasks. Within the architecture of the model, frozen layers, including convolutional and max pooling layers, serve the purpose of feature extraction. The concluding segment of the model comprises a densely connected neural network, culminating in a softmax layer responsible for performing the classification task.

4.3 Selected layers and parameter sets

In the accompanying Table 2, a comparative analysis of two distinct architectures derived from the VGG16 model is presented. For the initial architecture, the retention of the original model's fully connected final layers was complemented by the integration of dropout layers. Conversely, the second architecture employed a more compact model, examining the feasibility of a network with significantly reduced parameters in learning the specified task.

Table 2. Architectures

Architecture 1	*Architecture 2*
Frozen Layers	
	Dense (512)
	dropout (0.25)
Dense (4096)	Dense (256)
dropout (0.5)	dropout (0.25)
Dense (4096)	Dense (256)
dropout (0.5)	dropout (0.25)
softmax (2)	softmax (2)
output	output

Architecture 1 was characterized by a comprehensive parameter profile, encompassing a total of 134,268,738 parameters, equating to a memory footprint of approximately 512.19 megabytes. Within this architecture, the subset of trainable parameters amounted to 119,554,050 corresponding to 456.06 megabytes of memory usage. The remaining 14,714,688 parameters, occupying 56.13 megabytes, were designated as non-trainable. Architecture 2 exhibited a total parameter count of 27,757,890, which translated to a memory allocation of 105.89 megabytes. Within this framework, the trainable parameters were quantified at 13,043,202, accounting for 49.76 megabytes of memory. The non-trainable component of the architecture comprised 14,714,688 parameters, occupying an additional 56.13 megabytes of memory.

5 Results

Training of the models was conducted utilizing the EarlyStopping method, configured to monitor 'val_loss' with a patience setting of 3. EarlyStopping serves as a regularization technique in the training of deep learning models, aiming to avoid overfitting. This method is available within the TensorFlow/Keras framework, functioning as a callback that oversees a chosen metric, such as validation loss or accuracy, throughout the training phase. Should there be no improvement in this metric for a set number of epochs, EarlyStopping terminates the training prematurely. This intervention ensures the model remains generalizable, preventing it from excessively adapting to the training dataset and thus maintaining its performance on novel, unseen data. Additionally, the Adam optimizer was employed, featuring a learning rate of 0.0001, utilizing 'categorical_crossentropy' as the loss function, and 'accuracy' as the performance metric. The training duration for the initial model (Architecture 1) was approximately 147 seconds, whereas the subsequent model (Architecture 2) required around 132 seconds. This process was executed within a Colab environment, utilizing a Tesla T4 GPU for computational support.

5.1 Architecture 1: original trainable layers with dropouts

Figure 5 displays the training and validation outcomes for the initial configuration. The left section of the figure presents the model accuracy values across 6 epochs.

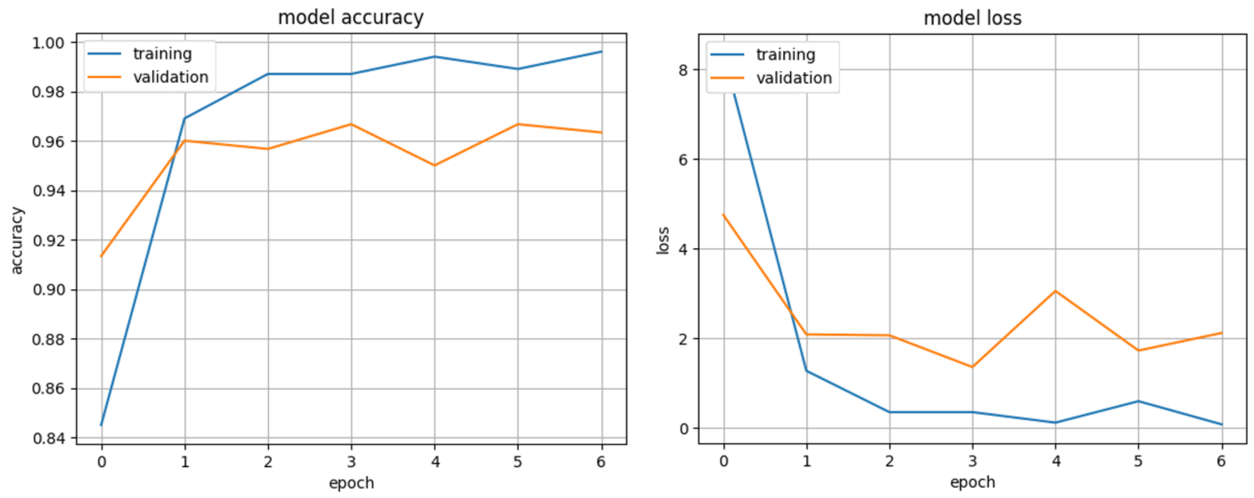

Figure 5. Model accuracy and loss for Architecture 1

For the testing phase, as outlined earlier, we utilized images obtained from the repair workshop. These images were entirely separate and independent from those used in the training and validation stages, ensuring a distinct dataset for testing purposes.

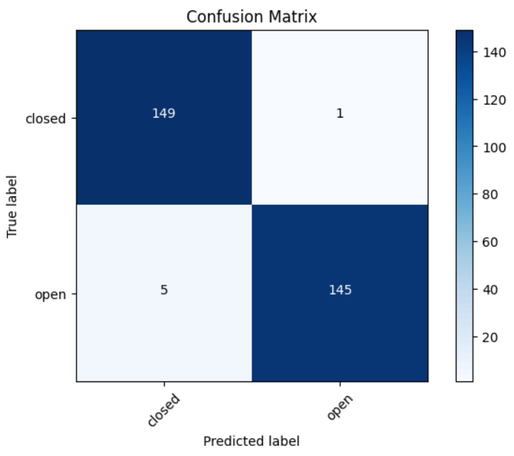

Figure 6. Confusion matrix for Architecture 1

The Confusion Matrix, a highly effective instrument in machine learning, is employed for assessing the efficiency of classification models. Consequently, it has been utilized in this study to present the results of our testing phase (Figure 6). In our examination of architecture 1, we attained an accuracy rate of 98%. This metric of accuracy is derived by dividing the total count of correct predictions by the aggregate number of predictions made. The identification of vehicles with a closed hood yielded more favorable results, with only 1 instances of misclassification, compared to the recognition of an open hood, which presented 5 cases of incorrect detection.

5.2 Architecture 2: smaller trainable layers with dropouts

In its updated iteration, Architecture 2 has been significantly downsized, constituting only one-fifth the size of its earlier version. Correspondingly, the count of trainable and learnable parameters has been drastically reduced, amounting to nearly one-tenth of what was previously available (see section 4.3). The outcomes of these modifications are presented in Figures 7 and 8. The left portion of Figure 6 demonstrates that the model achieves greater accuracy in epochs 1,2,3 in the validation phase compared to the training stage, a result attributable to the incorporation of dropout layers.

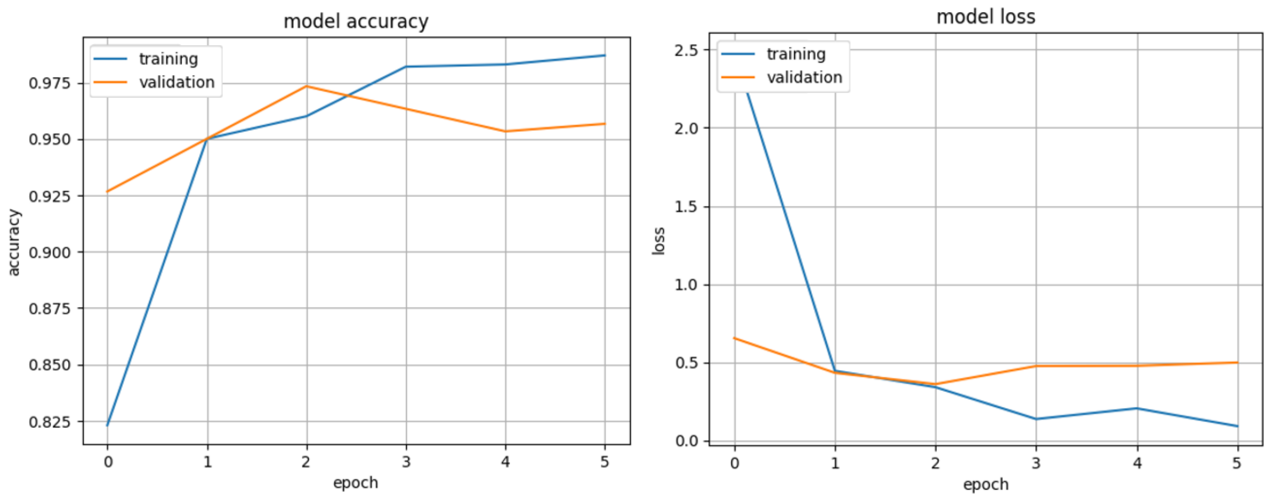

Figure 7. Model accuracy and loss for Architecture 2

Dropout is a regularization technique that randomly deactivates certain neurons in a neural network during training [17]. This process prevents the model from excessively adapting to the training data (overfitting), which could result in a decrease in training accuracy. However, it can enhance the model's generalization capability, potentially leading to improved validation accuracy.

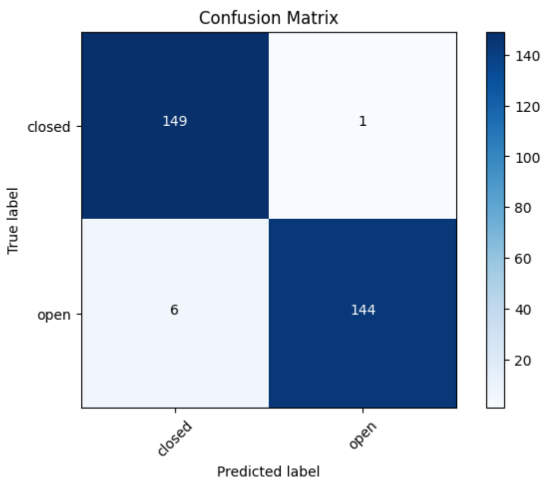

Figure 8. Confusion matrix for Architecture 2

In this research, the second architecture, which is 25% smaller than its predecessor, demonstrated a comparable level of accuracy, achieving 97.67%. This was accomplished with a lower dropout rate of 0.25, in contrast to the 0.5 dropout rate used in the first architecture. The task of correctly identifying vehicles with a closed hood showed more promising results, with a notably low misclassification rate—only a single instance of error (Figure 8). In contrast, the task of detecting vehicles with an open hood was more challenging, resulting in six instances of incorrect identification. We experimented with various parameter sizes for the trainable layers, yet failed to achieve enhanced accuracy.

6 Summary

This paper investigates the use of transfer learning with convolutional neural networks (CNNs), particularly the VGG16 model, for image classification in scenarios with limited data. Focusing on differentiating between cars with open and closed hoods, the study demonstrates that high accuracy (above 97%) can be achieved with a dataset of just 1000 images. The research compares two distinct architectural designs, noting the efficiency of a smaller model with lower dropout rates. This work underscores the potential of transfer learning in efficiently solving classification problems with small image datasets. For future research in CNN transfer learning with small datasets, consider these directions: testing various CNN models like Inception or ResNet; examining transfer learning across different fields; exploring incremental learning with new class additions; assessing advanced data augmentation impacts; researching transfer learning in unsupervised or semi-supervised contexts; applying models to practical scenarios like autonomous driving or medical imaging; and focusing on making CNN models more interpretable, especially in critical applications.

Acknowledgment

2020-1.1.2-PIACI-KFI-2020-00070 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2020-1.1.2-PIACI-KFI funding scheme.

References

[1] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, “How transferable are features in deep neural networks?,” 2014, doi: 10.48550/ARXIV.1411.1792.

[2] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. Neural Computation, 1(4), 541-551.

[3] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv 1409.1556. doi:10.48550/arXiv.1409.1556

[4] K. He, X. Zhang, S. Ren, és J. Sun, „Deep Residual Learning for Image Recognition”, in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, jún. 2016, o. 770–778. doi:10.1109/CVPR.2016.90.

[5] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going Deeper with Convolutions.

[6] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 1800-1807, doi: 10.1109/CVPR.2017.195.

[7] Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

[8] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," in IEEE Transactions on Knowledge and Data Engineering, vol. 22, no. 10, pp. 1345-1359, Oct. 2010, doi: 10.1109/TKDE.2009.191.

[9] Weiss, K., Khoshgoftaar, T.M. & Wang, D. A survey of transfer learning. J Big Data 3, 9 (2016). doi:10.1186/s40537-016-0043-6

[10] Torrey, L. and Shavlik, J. (2010) Transfer Learning. In: Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques, IGI Global, Hershey, 242-264. doi:10.4018/978-1-60566-766-9.ch011

[11] Yosinski, Jason & Clune, Jeff & Bengio, Y. & Lipson, Hod. (2014). How transferable are features in deep neural networks?. Advances in Neural Information Processing Systems (NIPS). 27.

[12] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C. (2018). A Survey on Deep Transfer Learning. In: Kůrková, V., Manolopoulos, Y., Hammer, B., Iliadis, L., Maglogiannis, I. (eds) Artificial Neural Networks and Machine Learning – ICANN 2018. ICANN 2018. Lecture Notes in Computer Science(), vol 11141. Springer, Cham. doi:10.1007/978-3-030-01424-7_27

[13] Tammina, Srikanth. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. International Journal of Scientific and Research Publications (IJSRP). 9. p9420. 10.29322/IJSRP.9.10.2019.p9420.

[14] D. I. Swasono, H. Tjandrasa and C. Fathicah, "Classification of Tobacco Leaf Pests Using VGG16 Transfer Learning," 2019 12th International Conference on Information & Communication Technology and System (ICTS), Surabaya, Indonesia, 2019, pp. 176-181, doi: 10.1109/ICTS.2019.8850946

[15] Pardede, Jasman & Sitohang, Benhard & Akbar, Saiful & Khodra, Masayu Leylia. (2021). Implementation of Transfer Learning Using VGG16 on Fruit Ripeness Detection. International Journal of Intelligent Systems and Applications. 13. 52-61. 10.5815/ijisa.2021.02.04.

[16] https://www.mturk.com/ [Accessed: 15-Nov-2023].

[17] Srivastava, Nitish & Hinton, Geoffrey & Krizhevsky, Alex & Sutskever, Ilya & Salakhutdinov, Ruslan. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 15. 1929-1958.