Dog breed recognition using CNN

Ante Bilić
FER, Zagreb
Zagreb, Croatia
ante.bilic@fer.hr

Domagoj Planjar
FER, Zagreb
Zagreb, Croatia
domagoj.planjar@fer.hr

Nino Poletan
FER, Zagreb
Zagreb, Croatia
Nino.Poletan@fer.hr

Marita Radić
FER, Zagreb
Zagreb, Croatia
marita.radic@fer.hr

Lana Šprajc
FER, Zagreb
Zagreb, Croatia
lana.sprajc@fer.hr

Filip Pešut
FER, Zagreb
Zagreb, Croatia
filip.pesut@fer.hr

I. Introduction and Motivation

The subject of this project is dog breed recognition using CNN. The idea is to develop convolutional neural network which will be able to locate a dog on an image and classify it according to its breed. As part of the project, we will research the literature in order to get a better understanding of convolutional neural networks and find existing solutions of mentioned breed classification problem.

II. Review of Existing Solutions and Literature Summary

CNNs (Convolutional Neural Networks) are feed-forward neural networks typically comprised of multiple convolutional and subsampling layers followed by multilayered perceptron. Convolutional layers apply filters to a smaller part of input neurons. The result of each filter is a feature map. Their purpouse is to find different features regardless of their position, rotation or other transformation. Subsampling layers reduce the dimensionality of the problem by combining different neuron outputs into one. [**?**] Examples include max-pooling, average-pooling... CNNs have applications in image recognition, natural language processing, image classification...

Examples of CNN architectures are: LeNet, AlexNet, VGGNet, ResNet, EfficientNetB0, YOLO...

Now follows an overview of different existing solutions to dog breed recognition using CNN. Using a calibrated Xception model, [**?**] achieved accuracity of 98%. [**?**] aimed to recognised whether a picture is of a human face or dog and if it is a dog, which breed it belongs to. It used OpenCV to detect human faces, and for dog face detection, the VGG16 model. The final accuracity was 81%. [**?**] used YOLO-v7 to detect dog and cat breeds with F1 score of 85.6% and mAP of 82.57%.

III. Proposed Solution by the Project Team

In our pursuit to create an effective dog breed classification system, our project team employed a multi-model approach, utilizing three powerful neural network architectures: EfficientNetB0, ResNet50, and YOLOv8. This comprehensive strategy aimed to capture a diverse range of features and enhance the overall accuracy and robustness of the classification system.

A. Dataset

To train and evaluate our models, we utilized the "70 Dog Breeds-Image Data Set" sourced from Kaggle. This dataset, compiled by Piotr Głowacki, contains a collection of high-resolution images spanning 70 different dog breeds, with a total of over 9,400 images. Each image is labeled with its corresponding dog breed, enabling our models to learn intricate patterns and characteristics associated with each breed. The dataset contains 7946 train, 700 test, 700 validation images in 224x224x3 jpg format.

B. Models

1) EfficientNetB0: EfficientNetB0, a part of the EfficientNet family, is known for its remarkable efficiency in terms of parameter size and computational requirements. Developed by Tan and Le in 2019, EfficientNetB0 achieves state-of-the-art performance in image classification tasks. Its architecture is based on a compound scaling method.Unlike conventional practice that arbitrary scales these factors, the EfficientNet scaling method uniformly scales network width, depth, and resolution with a set of fixed scaling coefficients. [**?**]

In our implementation we used an EfficientNetB0 model with pretrained weights on the ImageNet dataset from the Keras python library as a base model. At the end of the model we added a flattening layer and 2 deeply connected (Dense) layers, one with size of 512 and a ReLU activation function and the other serving as an output layer with a softmax activation function. In our optimisation process we used the Adam optimizer with a learning rate of 0.001. As our loss function we used sparse categorical crossentropy.

2) ResNet50: ResNet50, a variant of the ResNet architecture proposed by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, is distinguished by its deep structure incorporating residual connections. ResNets, learn residual functions with reference to the layer inputs, instead of learning unreferenced functions. Instead of hoping each few stacked layers directly fit a desired underlying mapping, residual nets let these layers fit a residual mapping. They stack residual blocks ontop of each other to form network (ResNet50 has fifty layers using these blocks). These connections mitigate the vanishing gradient problem, enabling the model to learn complex features effectively. [**?**]

Similarly to the EfficientNet model, we used a ResNet50 model with pretrained weights on the ImageNet dataset from the Keras python library as a base model. At the end of the

model we added a flattening layer and 2 deeply connected (Dense) layers, one with size of 512 and a ReLU activation function and the other serving as an output layer with a softmax activation function. In our optimisation process we used the Adam optimizer with a learning rate of 0.001. As our loss function we also used sparse categorical crossentropy.

3) YOLOv8: The YOLOv8 model is popular object detection algorithm, whose architecture is divided in three parts. First part is called backbone, which is responsible for extracting image features. Extracted features are passed to the next part called neck. This component is used to fuse features from different scales, which are then forwarded to prediction. Prediction is carried out in part called head, which is used to predict classes.

For backbone, YOLO uses Cross Stage Partial (CSP) networks with convolutional layers. CSP networks addresses multiple gradient problems, like vanishing gradient and duplicate gradient, used in back-propagation algorithm for training. CSP networks also encourage network to reuse features and reduce number of parameters. YOLO uses BiFNP neck architecture for feature fusion. This network aims to aggregate features at different resolutions. Design of this network is published in EfficientDet: Scalable and Efficient Object Detection" [**?**]. It improves previous methods by adding weighted feature fusion, since different input features are at different resolutions, and they usually contribute to the output feature unequally.

Last component is called detection head, which is used for predicting object classes and confidence scores. It is composed from convolutional layers, that are encoding features relevant to object detection, and spatial reduction layers, that reduce dimensions and increase each node perceptual field. [**?**]

For the YOLO model training, we used stochastic gradient descent (SGD) with a learning rate of 0.01 and a momentum of 0.9. The loss function used was the binary crossentropy loss (BCELoss)

IV. Experimental Results Overview

A. Model results comparison with other models

In the pursuit of a comprehensive evaluation of image classification models, we undertook a methodical approach to model selection. Our objective was to ensure a fair and unbiased comparison, predicated on the models' performance on a standardized dataset. To this end, we sourced a collection of pre-trained models from Kaggle, a platform renowned for its extensive repository of datasets and machine learning models.

The criteria for selection were precise: models must have been trained on the same dataset to warrant comparability of results. Among the numerous models available, we employed a randomized selection method to choose three distinct architectures: MobileNetV3, ResNet-18, and DenseNet-121. This randomization process precluded any selection bias and provided an eclectic mix of model complexities and architectures.

MobileNetV3 was included as a representative of lightweight, efficient architectures designed for speed and low-memory footprint. ResNet-18 was selected to represent moderately sized networks that balance complexity and performance, while DenseNet-121 offered insight into densely connected networks that emphasize feature reuse.

TABLE I
OTHER MODELS

Model	MobileNetV3	ResNet-18	DenseNet-121
Params	4.3M	11.2M	7.0M
Params Memory	17.167 MB	44.85 MB	28.102 MB
Precision	0.798	0.728	0.717
Recall	0.792	0.736	0.719
f1-score	0.854	0.795	0.799
Accuracy	0.854	0.795	0.799

TABLE II
OUR MODELS

Model	EfficientNetB0	ResNet50	YOLOv8n
Params	5.3M	26.7M	1.5M
Params Memory	20.22 MB	101.8 MB	5.72 MB
Precision	0.92	0.89	0.96
Recall	0.91	0.88	0.95
f1-score	0.90	0.87	0.95
Accuracy	0.91	0.88	0.96

YOLOv8n emerges as the most efficient model among those compared, achieving the highest scores in all performance metrics with a relatively lower parameter count and memory usage. This analysis suggests that YOLOv8n's design provides a good balance between size and performance, making it a strong candidate for resource-constrained environments without compromising on model accuracy. EfficientNetB0's performance, while robust, may not justify its significant resource requirements in situations where efficiency is a priority.

B. Discussion and Comparative Analysis

One of the key findings from our analysis is that more complex models like ResNet50, characterized by a higher number of parameters and memory usage, tend to offer superior accuracy and robust feature extraction capabilities. This enhancement in performance, however, comes at the cost of increased computational resources. The EfficientNetB0, despite its higher resource requirements compared to lighter models, did not demonstrate a corresponding leap in performance metrics.

On the other hand, YOLOv8n stands out for its exceptional balance between size and performance. With a relatively lower parameter count and memory footprint, it has achieved the highest scores across all performance metrics. This efficiency makes YOLOv8n an ideal choice for environments where resources are constrained, without compromising on the accuracy of the model.

V. Conclusion

In conclusion, this comparative study of various image classification models highlights the importance of considering both efficiency and complexity in model selection. While complex models offer high accuracy, they require substantial

resources, which may not be feasible in all scenarios. Models like YOLOv8n, which provide a balance between efficiency and performance, are increasingly relevant in today's diverse and resource-aware application landscape. Future research could focus on further optimizing the balance between model complexity and resource efficiency, catering to the evolving needs of real-world applications.

REFERENCES

[1] S. Lončarić and M. Subašić, "Neuronske mreže: Duboke neuronske mreže i duboko učenje", Fakultet elektrotehnike i računarstva, Sveučilište u Zagrebu

[2] Dog breed classification using convolution neural network Amit Kumar Jakhar, Mrityunjay Singh, and Anjani Kumar Shukla International Journal of Swarm Intelligence 2021 6:2, 130-142

[3] Prasanth Vaidya S., et al. "A Novel Dog Breed Identification using Convolutional Neural Network". PriMera Scientific Engineering 2.1 (2023): 16-21.

[4] S. S, D. S and S. N. Reddy, "Efficient Real-time Breed Classification using YOLOv7 Object Detection Algorithm," 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT), Delhi, India, 2023, pp. 1-6, doi: 10.1109/ICCCNT56998.2023.10306688.

[5] Tan, Mingxing, and Quoc Le. "Efficientnet: Rethinking model scaling for convolutional neural networks." In International conference on machine learning, pp. 6105-6114. PMLR, 2019.

[6] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.

[7] Tan, Mingxing, Ruoming Pang, and Quoc V. Le. "Efficientdet: Scalable and efficient object detection." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10781-10790. 2020.

[8] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[9] Doe, J. (2024). MobileNet V3, 70 Dog Breeds-Image Classification. Version 1. Kaggle.

[10] Doe, J. (2024). ResNet-18, 70 Dog Breeds-Image Classification. Version 1. Kaggle.

[11] Doe, J. (2024). DenseNet-121, 70 Dog Breeds-Image Classification. Version 2. Kaggle.