Aplicação da Arquitetura YOLOv11 Nano na Classificação de Raças de Cães

Luize Cunha Duarte
Universidade Federal do Paraná - UFPR
luize.duarte@ufpr.br

Paulo Ricardo Lisboa de Almeida
Universidade Federal do Paraná - UFPR
paulorla@ufpr.br

Abstract

This study evaluates the effectiveness of the YOLOv11 Nano neural network architecture for dog breed classification. YOLOv11 Nano is a small convolutional neural network (CNN) model specifically designed for image classification task, particularly suited for objects with similar characteristics, such as dog breeds. The proposed architecture presents results comparable to previously applied networks, such as NASNet-A Mobile, and slightly underperforms in comparison to Inception-ResNet-V2, despite its substantially smaller model size. A comprehensive evaluation was conducted, incorporating pre-trained models alongside optimization strategies, including data augmentation and K-Fold cross-validation. The results highlight the potential of YOLOv11 Nano for the dog breed classification task, offering a balance between accuracy and size.

Keywords

REDES NEURAIS PROFUNDAS, CLASSIFICAÇÃO, REDE NEURAL CONVOLUCIONAL, YOLOV11, RAÇAS DE CÃES

1 Introdução

A classificação de raças de cães a partir de imagens é um desafio que envolve a identificação de padrões sutis entre classes semelhantes [1]. Como solução para o problema de processamento de imagens, surgem as redes *Convolutional Neural Networks* (CNNs). Essas redes são compostas por camadas convolucionais que aplicam filtros sobre as imagens para detectar diferentes características, além de camadas de *pooling* para reduzir a dimensionalidade. Essa combinação melhora a generalização do modelo e permitindo a classificação de imagens com diferenças pequenas entre as classes.

Este trabalho apresenta uma análise detalhada da rede neural YOLOv11 Nano (*You Only Look Once*) [2], de arquitetura CNN, e sua aplicação no problema específico da classificação de raças de cães do *dataset Stanford Dogs* [3]. A pesquisa demonstra a acurácia do modelo comparada com redes anteriores, como NASNet-A *Mobile* [4] e Inception-ResNet-V2 [5], originalmente propostas em [1]. A comparação entre as diferentes arquiteturas aponta as vantagens da rede YOLOv11, cujos resultados demonstram um desempenho competitivo na classificação de imagens de raças de cães.

2 Trabalhos Relacionados

Dentre as redes baseadas na arquitetura de CNN, amplamente reconhecidas como solução para problemas de processamento de imagens, está a YOLO *(You Only Look Once)* [6]. Nesse contexto, diversos estudos têm explorado a eficácia dos modelos YOLO em diferentes problemas de classificação. Sharma et al. (2024) [7], por exemplo, realizou uma comparação entre as versões oito a onze dos modelos YOLO e Faster R-CNN para a detecção de múltiplas espécies de ervas daninhas. Na área da saúde, Awad et al. (2024) [8] demonstrou a aplicação de modelos YOLO no diagnóstico precoce de leucemia linfoblástica aguda. Por fim, Ganesan et al. (2022) [9] propôs um modelo híbrido que combina a rede ResNet e um classificador YOLO para o reconhecimento automatizado de doenças em folhas de arroz.

No domínio do problema da identificação de raças de cães, Wang et al. (2015) [10] traz abordagens baseadas em *landmarks* para melhorar a precisão na classificação. Já o trabalho de Ráduly et al. (2018) [1] aplica redes profundas para a classificação de raças caninas utilizando um conjunto de dados diversificado e técnicas modernas de *deep learning*.

3 Solução Proposta

A solução proposta aplica a rede YOLOv11 para a classificação de raças de cães. A escolha do modelo se baseou em seu tamanho reduzido e a capacidade de equilibrar precisão e tempo de execução [7], características que o tornam ideal para aplicações em tempo real. Para adaptar a rede ao problema de classificação de raças, é utilizado um modelo pré-treinado e realizado treinamento no *dataset Standford Dogs*. Por fim, a solução proposta emprega técnicas de otimização, como *data augmentation* e validação cruzada com *K-Folds*, durante o treinamento, visando melhorar os resultados nas métricas avaliadas.

4 Experimentos

Nesta seção, são apresentados os procedimentos realizados para o treinamento das redes neurais no problema de classificação de raças de cães. seção está organizada em três subseções. Primeiramente, na subseção 4.1, são descritas as características principais das arquiteturas utilizadas. Em seguida, na subseção 4.2, o conjunto de dados utilizado é abordado, detalhando as etapas de pré-processamento e técnicas aplicadas. Por fim, na subseção 4.3, são especificados os parâmetros adotados durante o treinamento, como funções de perda, otimizadores e configurações de aprendizado.

4.1 Redes Neurais

A arquitetura selecionada, YOLOv11 Nano, é a menor versão disponível da família YOLOv11. Ela possui 151 camadas e 1.684.824 parâmetros, sendo significativamente menor em comparação com a maior versão, a YOLOv11-X, que conta com mais de 28 milhões de parâmetros. Para uma análise mais detalhada do desempenho da rede YOLOv11 Nano, também serão utilizadas as redes NASNet-A *Mobile* e Inception-ResNet-V2, as quais estruturas diferentes, conforme mostra a Tabela 1.

A NASNet-A *Mobile* teve a arquitetura de suas camadas convolucionais projetada por meio de *Neural Architecture Search* (NAS) [11]. O NAS utiliza *reinforcement learning* para explorar e identificar arquiteturas completas de redes neurais ideais para datasets

específicos. No caso da NASNet-A, a busca foi simplificada para encontrar apenas a melhor camada convolucional para um dataset menor, aprimorando a generalização da célula [4]. Ela possui 389 camadas e 5,3 milhões de parâmetros.

Já a Inception-ResNet-V2 é uma rede neural profunda que combina blocos do tipo *Inception* com blocos residuais (ResNet) buscando melhorar a eficiência e a precisão em tarefas de classificação de imagens. A estrutura Inception divide o processamento em múltiplos filtros convolucionais de diferentes tamanhos e depois combina os resultados. Ao adicionar a abordagem de aprendizado residual da ResNet, a Inception-ResNet-v2 acrescenta conexões residuais entre camadas, o que facilita o fluxo de informações por meio de redes muito profundas e ajuda a evitar o problema do gradiente desaparecendo durante o treinamento. Ela possui 449 camadas e mais de 55 milhões de parâmetros [5].

Considerando a discrepância no tamanho das redes, as redes NASNet-A e Inception-ResNet-V2 demoraram cerca de quatro vezes mais tempo do que a YOLOv11 para realizarem o treinamento.

Modelos	Camadas	Parâmetros
NASNet-A	389	5.300.000
Inception-ResNet-V2	449	55.900.000
YOLOv11	151	1.684.824

Tabela 1: Especificações dos modelos.

Visando otimizar o treinamento dessas redes para a identificação de diferentes raças, são realizados ajustes em modelos pré-treinados, uma técnica chamada *transfer learning*. Nesse contexto, as redes utilizadas foram pré-treinadas no *dataset ImageNet* [12], o qual é amplamente utilizado para o pré-treinamento de diversos modelos de classificação devido à sua variedade de classes e qualidade de dados. Ademais, essas redes pré-treinadas estão disponíveis publicamente e podem ser acessadas por meio de bibliotecas, como Keras [13] e Ultralytics [14].

4.2 Dataset e Pré-Processamento

Para o problema de classificação de raças caninas, foi utilizado o conjunto de dados rotulado *Stanford Dogs* [3], composto por 20.580 imagens de cães de 120 raças distintas. Este *dataset* foi criado a partir do *ImageNet*, tornando os modelos pré-treinados uma ótima base para o aprimoramento das redes no problema proposto. O *Stanford Dogs* é dividido em dois subconjuntos: um conjunto de treinamento, contendo 12.000 imagens com distribuição uniforme entre as raças, e um conjunto de teste, composto por 8.580 imagens, no qual a quantidade de imagens de cada classe é distribuída aleatóriamente.

Para maximizar a utilização do conjunto de dados durante o treinamento, foi adotada a técnica de validação cruzada com *K-Folds* como parte do pré-processamento. Essa técnica consiste na criação de k permutações aleatórias do conjunto de dados, cada uma dividida em um conjunto de treinamento, com 9.200 imagens, e um conjunto de validação, composto por 800 imagens. Neste caso, foram criados 5 *folds* distintos, ou seja, $k = 5$. Dessa forma, o processo de treinamento resulta em cinco redes distintas.

Além da validação cruzada com *K-Folds*, a técnica de *data augmentation* foi aplicada em todas as classes para melhorar a generalização do modelo. Conforme proposto em [1], no treinamento das redes NASNet-A *Mobile* e Inception-ResNet-V2, foi utilizado o algoritmo *Distorted Bounding Boxes*, o qual realiza cortes aleatórios nas bordas das imagens e, posteriormente, elas são invertidas aleatoriamente no sentido vertical. Para o conjunto de dados de validação, foi realizado apenas um corte central de 87,5% das imagens. Após as transformações realizadas tanto no conjunto de treinamento quanto no de validação, foi aplicada a função de pré-processamento *Inception*, a qual normaliza as imagens no intervalo $[-1, 1]$.

No treinamento da rede YOLOv11, foram aplicadas também as técnicas de corte e espelhamento das imagens, além de uma normalização para o intervalo $[0, 1]$, apropriada para sua arquitetura.

4.3 Treinamento e Parâmetros

Conforme proposto em [1], para o treinamento das redes NASNet-A *Mobile* e Inception-ResNet-V2, foram adotados os seguintes parâmetros gerais, os quais também foram aplicados ao YOLO: 32 épocas, um *batch size* de 64 e um *weight decay* configurado em 0,0001. A função de perda utilizada foi a *Softmax Cross-Entropy*, capaz de transformar as saídas do modelo em probabilidades interpretáveis e de medir a discrepância entre as previsões e os rótulos reais [15].

No caso das redes NASNet-A *Mobile* e Inception-ResNet-V2, foi utilizada uma taxa de aprendizado variável, com uma redução de 10% em seu valor a cada 3 épocas. O otimizador utilizado foi o *Nesterov Momentum*, que é uma variante do método de gradiente estocástico com *momentum*. Durante o treinamento, a última camada totalmente conectada da rede foi "descongelada", permitindo o ajuste de seus pesos, enquanto as camadas restantes permaneceram inalteradas. Esta técnica visa aproveitar o conhecimento dos modelos pré-treinados, ajustando apenas as camadas finais para o novo conjunto de dados.

Por outro lado, as camadas da rede YOLOv11 estavam todas liberadas para treinamento, permitindo ajustes durante o processo de aprendizado. Para as redes YOLOv11, os otimizadores mais comuns são o AdamW ou SGD. Dado o tamanho do conjunto de imagens de treinamento e a escolha de *batches* de tamanho 64, foi selecionado o otimizador AdamW, que combina as vantagens do Adam com a adição de um decaimento de peso para melhorar a regularização. A taxa de aprendizado inicial foi definida como 0,000714, valor calculado pela biblioteca da Ultralytics [14], com base no tamanho do dataset e no otimizador selecionado. Por fim, uma queda baseada no método de decaimento cosseno, ajustando dinamicamente a taxa de aprendizado ao longo das épocas para otimizar a convergência.

5 Resultados Parciais

Para a avaliação dos modelos, considerando que o método de *k-folds* gera cinco redes distintas, foram selecionadas para análise aquelas que apresentaram os menores valores de *loss* nos conjuntos de validação. A evolução do *loss* durante o treinamento de cada modelo é apresentada na Figura 1.

A curva de perda do modelo NASNet-A *Mobile* demonstra um aprendizado inicial rápido. No entanto, a taxa de melhoria diminui rapidamente, estabilizando-se no maior valor entre os modelos, indicando dificuldades em otimizar mais o modelo. Por outro lado, o *loss* da rede YOLOv11 diminui mais rapidamente do que os outros modelos e permanece com valores inferiores até o final do treinamento.

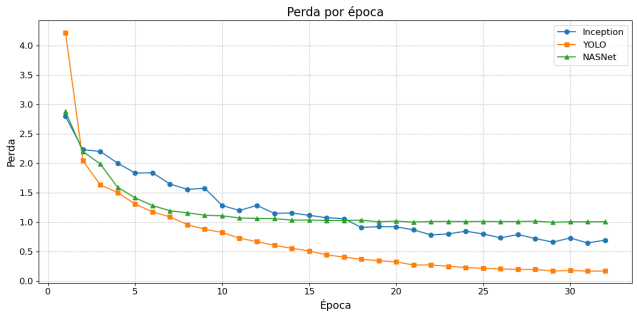

Figura 1: Resultado da função de perda nos conjuntos de validação em cada época.

Outra métrica relevante para a comparação entre os modelos é a acurácia na classificação do conjunto de teste, conforme mostrado na Tabela 2. Embora a rede YOLOv11 tenha apresentado melhores valores de perda durante o treinamento, sua acurácia foi muito similar à da rede NASNet-A, com 75% e 74%, respectivamente. Isso pode ocorrer devido a uma dificuldade de generalização do modelo para novos dados, como os presentes no conjunto de teste. Por outro lado, o modelo Inception-ResNet-V2 obteve a maior acurácia, alcançando 83%. Por fim, as métricas *Precision* e o *Recall* foram

Modelos	Acurácia
NASNet-A	74%%
Inception-ResNet-V2	83%
YOLOv11 Nano	75%

Tabela 2: Tabela da acurácia dos modelos no conjunto de teste.

utilizadas para avaliar o desempenho do modelo no conjunto de teste. Considerando que a classificação das raças de cães é um problema de múltiplas classes, a métrica de desempenho foi calculada por meio de um processo conhecido como *macro-averaging*. Nesse método, cada classe recebe o mesmo peso, independentemente do número de instâncias que possui.

Conforme apresentado na Tabela 3, todas as redes possuem um bom equilíbrio entre as métricas de *precision* e *recall*. A rede Inception-ResNet-V2 obteve os melhores resultados, com valores iguais a 81%. Contudo, as redes NASNet-A e YOLOv11 apresentaram valores menores, com *precision* de 74% e *recall* de 72% para a NASNet-A e 74% em ambas as métricas para a YOLOv11.

Modelos	Precision	Recall
NASNet-A	74%	72%
Inception-ResNet-V2	81%	81%
YOLOv11	74%	74%

Tabela 3: Tabela de *Precision* e *Recall* nos testes.

6 Considerações Finais

Os resultados obtidos neste estudo demonstram a competitividade da rede neural YOLOv11 Nano na tarefa de classificação de raças de cães. A abordagem proposta, que combina o uso de modelos pré-treinados e outras técnicas de treinamento, como validação cruzada com *K-Folds* e *data augmentation*, mostrou-se eficaz para o problema. Embora a YOLOv11 Nano tenha apresentado resultados inferiores à rede Inception-ResNet-V2, ela obteve resultados comparáveis à NASNet-A *Mobile*, a qual possui uma arquitetura com três vezes mais parâmetros. Esses resultados semelhantes foram observados tanto nas métricas de *precision* e *recall* quanto na acurácia, apontando que, mesmo com um modelo extremamente menor, a YOLOv11 Nano é capaz de alcançar resultados consideráveis, demonstrando seu potencial para aplicações com restrições de recursos computacionais.

Acknowledgments

Este projeto foi financiado pelo Ministério da Saúde através de uma TED para PD&I entre SAPS/MS e C3SL/UFPR.

Referências

[1] Zalán Ráduly, Csaba Sulyok, Zsolt Vadászi, and Attila Zölde. Dog breed identification using deep learning. In *2018 IEEE 16th International Symposium on Intelligent Systems and Informatics (SISY)*, pages 000271–000276, 2018. doi: 10.1109/SISY.2018.8524715.

[2] Glenn Jocher and Jing Qiu. Ultralytics yolo11, 2024. URL https://github.com/ultralytics/ultralytics.

[3] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Stanford dogs dataset. URL http://vision.stanford.edu/aditya86/ImageNetDogs/. Accessed: 2024-11-17.

[4] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning transferable architectures for scalable image recognition, 2018. URL https://arxiv.org/abs/1707.07012.

[5] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning, 2016. URL https://arxiv.org/abs/1602.07261.

[6] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. doi: 10.1109/CVPR.2016.91.

[7] Akhilesh Sharma, Vipan Kumar, and Louis Longchamps. Comparative performance of yolov8, yolov9, yolov10, yolov11 and faster r-cnn models for detection of multiple weed species. *Smart Agricultural Technology*, page 100648, 2024. ISSN 2772-3755. doi: https://doi.org/10.1016/j.atech.2024.100648. URL https://www.sciencedirect.com/science/article/pii/S2772375524002533.

[8] Alaa Awad, Mohamed Hegazy, and Salah A. Aly. Early diagnoses of acute lymphoblastic leukemia using yolov8 and yolov11 deep learning models, 2024. URL https://arxiv.org/abs/2410.10701.

[9] Gangadevi Ganesan and Jayakumar C Dr. Hybridization of resnet with yolo classifier for automated paddy leaf disease recognition: An optimized model. *Journal of Field Robotics*, 39, 06 2022. doi: 10.1002/rob.22089.

[10] Punyanuch Borwarnginn, Worapan Kusakunniran, Sarattha Karnjanapreechakorn, and Kittikhun Thongkanchorn. knowing your dog breed: Identifying a dog breed with deep learning . *Machine Intelligence Research*, 18(1):45–54, 2021. ISSN 2731-538X. doi: 10.1007/s11633-020-1261-0. URL https://www.mi-research.net/en/article/doi/10.1007/s11633-020-1261-0.

[11] Barret Zoph and Quoc V. Le. Neural architecture search with reinforcement learning. *CoRR*, abs/1611.01578, 2016. URL http://arxiv.org/abs/1611.01578.

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[13] François Chollet et al. Keras. https://keras.io, 2015.

[14] Ultralytics. Ultralytics github repository, 2024. URL https://github.com/ultralytics. Accessed: 2024-11-17.

[15] Paras Dahal. Softmax and cross entropy loss. *parasdahal.com*, Jun 2017. URL https://parasdahal.com/softmax-crossentropy.