Estudo Comparativo de Modelos de Aprendizagem Profunda para Reconhecimento de Expressões Faciais

1st Pedro Henrique M. de S. Silva
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
phmss@cin.ufpe.br

***Resumo*—Este documento detalha a implementação de um projeto na área de aprendizagem profunda, focado na tarefa de reconhecimento de expressões faciais. Ele apresenta uma introdução ao tema, os objetivos alcançados e a justificativa para a realização do projeto, seguida de uma descrição das várias etapas executadas durante seu desenvolvimento. Por fim, inclui uma conclusão detalhada acerca do resultado dos experimentos.**

Keywords—reconhecimento de expressões faciais, aprendizagem profunda, aprendizagem de máquina, análise comparativa

I. Introdução

A tarefa de reconhecimento de expressões faciais (FER) ganhou destaque nos últimos anos, devido ao aumento no número de pesquisas relacionadas ao tópico, além do crescente investimento na área. Com isso, tornou-se necessário ter uma visão mais apurada sobre quais modelos são mais relevantes para uma questão específica de FER, levando em consideração fatores como o poder computacional disponível e a dificuldade do problema.

II. Objetivo

Este projeto teve como objetivo avaliar e comparar o desempenho de diferentes modelos de aprendizagem profunda no reconhecimento de uma ampla gama de expressões faciais. O objetivo foi alcançado por meio da análise da capacidade discriminativa desses modelos em classificar corretamente várias amostras em um conjunto de rótulos referentes às emoções humanas. Além disso, buscou-se compreender os efeitos da variação de complexidade de diferentes arquiteturas no desempenho apresentado, fornecendo um guia para a escolha de determinado modelo dadas as circunstâncias.

III. Justificativa

A escolha do tema deste projeto se justifica pela conhecida relevância do reconhecimento de expressões faciais em diversas áreas, incluindo interação humano-computador, análise de sentimentos e detecção de emoções em contextos clínicos e comerciais. Em um mundo cada vez mais digitalizado, a capacidade de compreender e interpretar as expressões faciais dos usuários é fundamental para aprimorar a interação e a personalização de sistemas computacionais. Além disso, a utilização de técnicas de aprendizagem profunda demonstrou resultados promissores nessa área, impulsionando avanços significativos no desenvolvimento de modelos mais precisos e eficientes.

IV. Base de Dados

O conjunto de dados escolhido para a análise foi o Real-world Affective Faces Database (RAF-DB), que, na versão livremente disponibilizada na plataforma Kaggle (que contém a modalidade da classificação em 7 emoções: supresa, medo, nojo, alegria, tristeza, raiva e neutralidade), contém 15.339 imagens rotuladas com expressões básicas ou compostas utilizando 40 marcadores independentes. O dataset já está dividido em 12.271 amostras de treino e 3.068 amostras de teste. A escolha desse dataset se deve à ampla variedade de expressões catalogadas nas imagens e à diversidade dos sujeitos das amostras em termos de idade, gênero, etnia, pose, oclusões (como a presença de óculos ou pelo facial) e transformações pós-processamento (filtros e efeitos especiais).

V. Análise Exploratória dos Dados

Após uma análise mais superficial do conjunto de dados, foram realizadas as seguintes etapas:

1) Criação do conjunto de validação: esta estapa consistiu de dividir o conjunto de treino do dataset em treino (80%) e validação (20%), a fim de verificar a capacidade de generalização dos modelos durante o treinamento dos modelos.

2) Visualização das amostras iniciais: esta etapa envolveu a exibição das primeiras 24 imagens do conjunto de dados de treinamento, juntamente com seus respectivos rótulos. Isso ajuda a ter uma ideia inicial do conteúdo do dataset, das diferentes expressões faciais presentes e da qualidade das imagens.

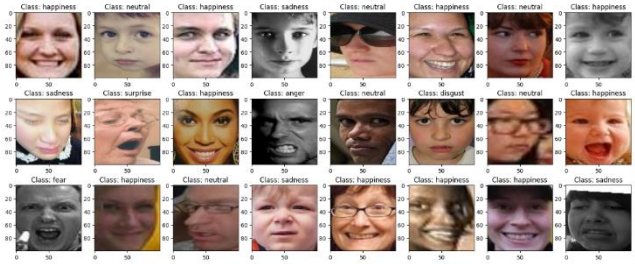

Fig. 1 Visualização das primeiras amostras.

3) Visualização das amostras por classe: o objetivo desta etapa foi garantir que cada classe (expressão facial) esteja representada no conjunto de dados. Para isso, uma imagem de exemplo de cada classe é visualizada, o que permite verificar a diversidade e a clareza das expressões faciais em cada categoria.

Fig. 2 Visualização de amostras por classes

4) Verificação da contagem e distribuição das amostras por classes: esta etapa consiste em contar o número de amostras disponíveis para cada classe nos conjuntos de treinamento, validação e teste, bem como do conjunto de dados no geral. Os resultados são apresentados em gráficos de barras, permitindo visualizar a distribuição das amostras entre as diferentes classes. Dessa forma, percebe-se como o dataset é desbalanceado, o que influenciará nos experimentos realizados.

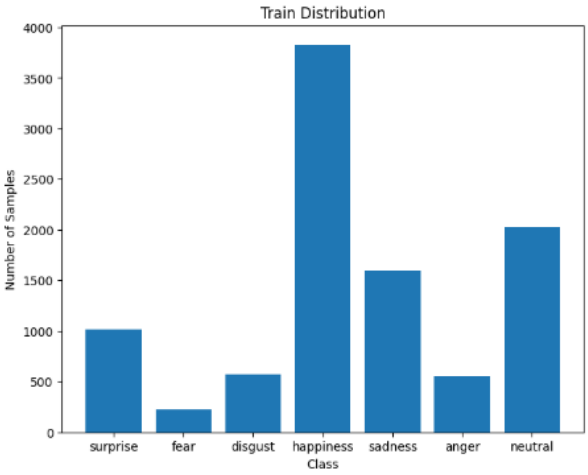

Fig. 3 Distribuição das classes no conjunto de treinamento.

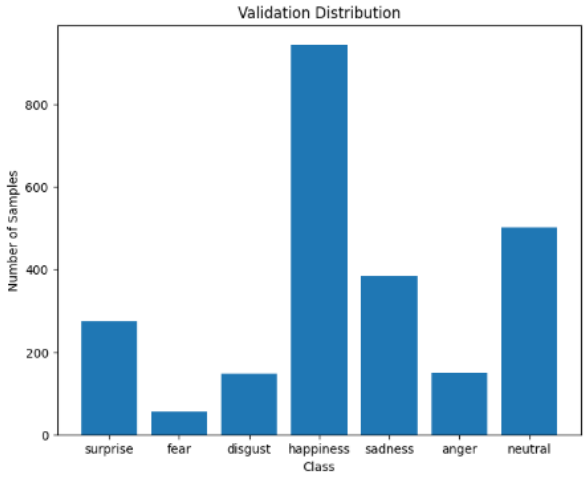

Fig. 4 Distribuição das classes no conjunto de validação.

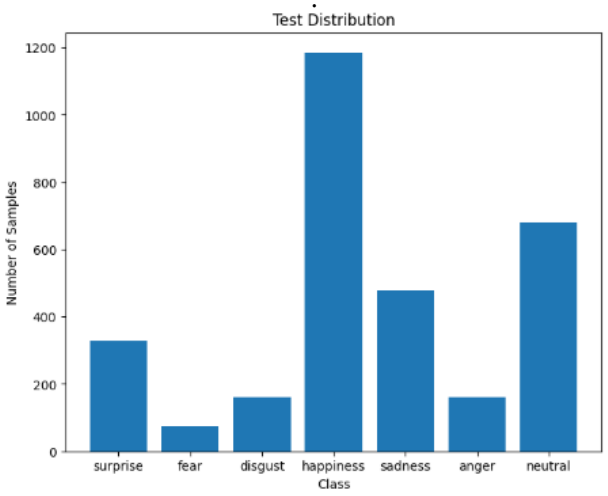

Fig. 5 Distribuição das classes no conjunto de teste.

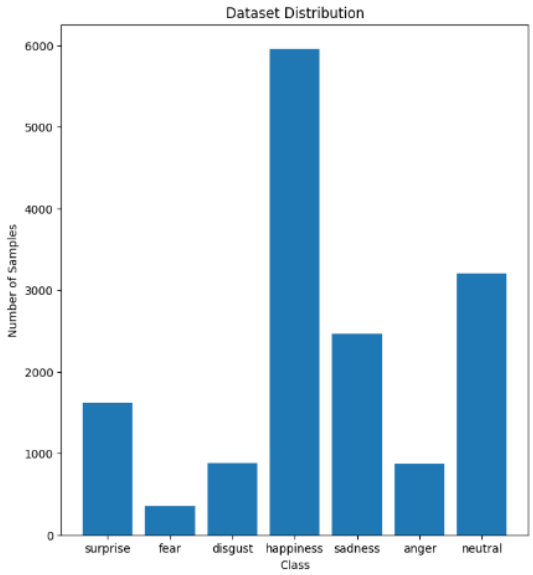

Fig. 6 Distribuição de classes em relação ao conjunto de dados inteiro.

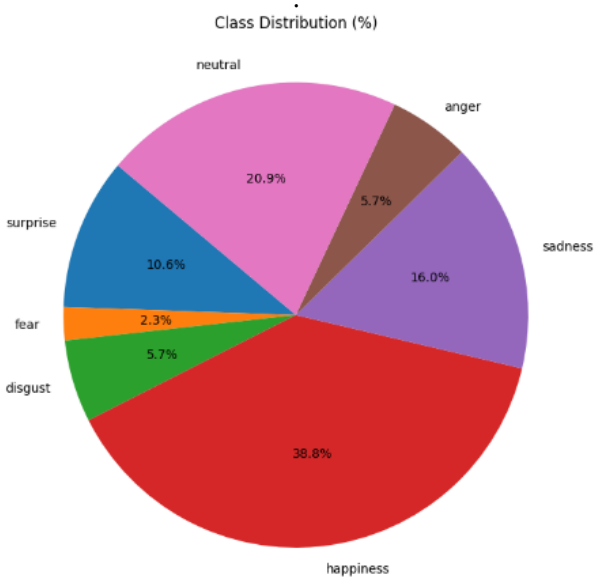

Fig. 7 Distribuição de classes em porcentagens.

5) Aplicação de técnicas de Data Augmentation: nesta etapa, são aplicadas transformações de dados para aumentar a diversidade do conjunto de treinamento. Técnicas como redimensionamento, rotação, recorte, espelhamento horizontal e normalização são utilizadas para gerar novas variações das imagens, ajudando a melhorar a robustez e a generalização dos modelos utilizados.

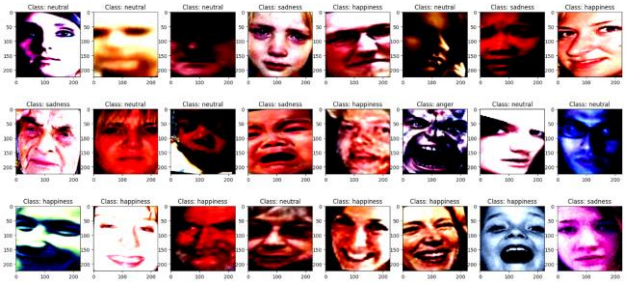

Fig. 8 Visualização das amostras após técnicas data augmentation.

VI. Modelos

Os modelos utilizados durante a comparação foram os seguintes:

1) VGG19: A arquitetura VGG19 consiste em 19 camadas profundas, das quais 16 são camadas convolucionais e 3 são camadas totalmente conectadas. Ela utiliza filtros de tamanho 3x3 nas camadas convolucionais, com convoluções sequenciais seguidas por funções de ativação ReLU. Operações de Max-Pooling (2x2) são aplicadas após blocos de convoluções para redução da dimensionalidade. No final, três camadas totalmente conectadas (com 4096, 4096 e 1000 neurônios, respectivamente) precedem a camada de saída. A simplicidade e a eficiência da VGG19 a tornam uma escolha popular para tarefas de classificação de imagens, oferecendo um bom equilíbrio entre profundidade e desempenho.

2) ResNet18: A arquitetura ResNet18 é composta por 18 camadas e se destaca pelo uso de blocos residuais, que permitem a propagação direta de informações através de conexões de atalho. Esses blocos mitigam o problema do gradiente desaparecendo, permitindo a construção de redes muito mais profundas. A ResNet18 é composta por uma combinação de convoluções 3x3 e 1x1, além de operações de pooling e camadas totalmente conectadas no final. A capacidade de treinar redes profundas com eficiência faz da ResNet18 uma escolha atraente para reconhecimento de padrões complexos em imagens.

3) EfficientNet B0: A EfficientNet B0 é uma das variantes da família EfficientNet, projetada com base na técnica de compound scaling, que ajusta uniformemente a profundidade, a largura e a resolução da rede. Ela utiliza convoluções convolucionais padrão, alongadas por blocos MBConv, e incorpora técnicas de regularização como o dropout. A EfficientNet B0 é altamente eficiente em termos de desempenho e uso de recursos, oferecendo um ótimo equilíbrio entre precisão e custo computacional. Sua eficiência a torna adequada para aplicações onde recursos de hardware são limitados.

4) Vision Transformer: A arquitetura Vision Transformer aplica o mecanismo de autoatenção dos Transformers, originalmente desenvolvido para processamento de linguagem natural, em imagens. O ViT divide a imagem em patches fixos, transforma cada patch em uma sequência linear e, em seguida, alimenta essas sequências em uma rede Transformer. Essa abordagem permite capturar relações de longo alcance entre diferentes partes da imagem, tornando-a particularmente eficaz em tarefas que requerem análise contextual detalhada. A capacidade do ViT de lidar com relações complexas entre pixels faz dele uma escolha poderosa para reconhecimento de padrões visuais complexos.

VII. Experimentos

Durante a implementação dos modelos, foram realizados os seguintes experimentos:

1) Experimento 1 (base): A metodologia do experimento 1 baseia-se na implementação e treinamento de modelos de aprendizado profundo para o reconhecimento de expressões faciais. Inicialmente, o modelo é preparado e seus parâmetros são salvos utilizando a função de inicialização do modelo. Esta função salva o estado inicial do modelo, do otimizador e dos parâmetros de avaliação, estabelecendo um ponto de partida consistente para o treinamento.

O treinamento do modelo é realizado pela função de treino, que carrega o modelo, o otimizador e as métricas de um checkpoint pré-existente, permitindo a retomada do treinamento de onde parou. O modelo é então treinado ao longo de várias épocas, onde, para cada época, ele é ajustado utilizando o conjunto de dados de treinamento. Durante o treinamento, a perda é calculada e os pesos do modelo são atualizados através do otimizador. A precisão do treinamento é avaliada a cada época, e as métricas de perda e precisão são armazenadas para análise posterior.

Para a validação, o modelo é colocado em modo de avaliação e testado no conjunto de dados de validação, onde a perda e a precisão são calculadas sem atualizar os pesos do modelo. Se a precisão de validação obtida for superior à melhor precisão registrada até o momento, o estado do modelo é salvo como o melhor modelo.

As métricas de treinamento e validação são salvas após cada época, e os checkpoints do modelo são atualizados para permitir a continuidade do treinamento em execuções futuras. Após o término do treinamento, a função de teste é utilizada para avaliar o modelo no conjunto de dados de teste. As previsões do modelo são comparadas com os rótulos verdadeiros, e as métricas de avaliação, como o relatório de classificação e a matriz de confusão, são geradas pela função de avaliação do modelo. Essas métricas fornecem uma visão detalhada do desempenho do modelo, destacando as classes nas quais o modelo se sai melhor e aquelas que apresentam maiores desafios.

Esse processo metodológico garante um treinamento consistente, avaliações regulares e checkpoints periódicos, permitindo tanto a continuidade do treinamento quanto a análise detalhada do desempenho do modelo em diferentes estágios.

2) Experimento 2 (tencrop): O experimento 2 expande a metodologia do experimento 1 adicionando a técnica de data augmentation chamada TenCrop, que aumenta a robustez e precisão do modelo durante o processo de teste. O código tencrop utiliza a transformação TenCrop para gerar dez

diferentes recortes de uma imagem de entrada. Cada recorte é processado individualmente pelo modelo, e a média das previsões de todos os recortes é utilizada para a previsão final da classe.

O treinamento do modelo no experimento 2 segue a mesma metodologia do experimento 1, utilizando a função de treino. No entanto, a função de teste foi modificada para incorporar a técnica TenCrop através da nova função de teste. Durante o teste, a função tencrop é aplicada a cada imagem do conjunto de teste, gerando dez recortes, relativos a cada ponta da imagem, bem como o centro, além da versão espelhada desses cortes. O modelo faz previsões para cada um desses recortes individualmente, e a média dessas previsões é usada como a predição final.

Ao utilizar a técnica TenCrop durante o teste, o experimento 2 visa melhorar a robustez do modelo, reduzindo o impacto de variações na posição e na escala das expressões faciais nas imagens. O obejtivo foi alcançar uma maior generalização dos modelos em dados de teste, resultando em maior precisão e confiabilidade nas predições finais.

3) Experimento 3 (tencrop + oversampling): O experimento 3 expande a metodologia do experimento 2 ao incorporar o uso de amostras ponderadas para lidar com desbalanceamentos nas classes do conjunto de dados. Isso é realizado através da criação e aplicação de pesos de amostras no DataLoader para o conjunto de treinamento.

Primeiramente, são calculados os pesos das classes com base na frequência de cada classe no conjunto de treinamento. Para cada classe, o peso é definido como o inverso do número de amostras dessa classe, ajudando a compensar a desigualdade na representação das classes durante o treinamento.

Um WeightedRandomSampler (da biblioteca PyTorch) é então criado usando os pesos das amostras. Este sampler é integrado ao DataLoader para o conjunto de treinamento, substituindo a amostragem aleatória padrão. Com essa abordagem, o DataLoader seleciona amostras com base em seus pesos, permitindo que as classes menos representadas sejam mais frequentemente incluídas em cada batch de treinamento.

Portanto, o experimento 3 combina a técnica TenCrop, utilizada no experimento 2, com uma abordagem avançada de amostragem ponderada para abordar o desbalanceamento de classes. Essa metodologia visa melhorar ainda mais a precisão do modelo, especialmente em cenários onde algumas classes são significativamente menos representadas do que outras. Ao integrar amostragem ponderada, o experimento 3 procura garantir que os modelos aprendam de maneira mais equilibrada e eficaz, levando a uma maior generalização e robustez na classificação de expressões faciais.

VIII. Análise dos Resultados

Para avaliar os resultados obtidos, foram utilizadas as seguintes métricas quantitativas:

1) Acurácia: a acurácia representa a proporção de previsões corretas (tanto positivas quanto negativas) em relação ao total de casos analisados. É a métrica de avaliação mais comum e fornece uma visão geral da eficácia do modelo, mas pode ser enganosa em casos de desbalanceamento de classes.

2) Precisão: a precisão mede a proporção de previsões corretas positivas em relação ao total de previsões positivas realizadas para cada classe. Esta métrica avalia a precisão das previsões e indica a frequência com que as previsões positivas do modelo são corretas, refletindo o erro causado pelos falsos positivos.

3) Recall: o recall avalia a proporção de previsões corretas positivas em relação ao total de elementos reais da classe. Indica a capacidade do modelo de identificar corretamente todas as instâncias da classe, refletindo o erro causado pelos falsos negativos.

4) F1 Score: o F1 Score é a média harmônica entre precisão e recall, proporcionando uma métrica equilibrada quando há necessidade de considerar tanto a precisão quanto o recall.

Modelo	Experimento	Acurácia	Precisão (macro)	Recall (macro)	F1-score (macro)
VGG19	1	82.14%	76.45%	69.92%	72.10%
	2	82.14%	78.45%	67.73%	71.21%
	3	82.50%	79.56%	71.92%	74.46%
ResNet18	1	81.32%	74.28%	69.98%	71.25%
	2	83.41%	**79.69%**	72.70%	**75.20%**
	3	83.31%	76.45%	74.19%	74.93%
EfficientNet B0	1	80.93%	73.58%	73.98%	73.46%
	2	83.02%	76.12%	73.66%	74.66%
	3	83.05%	75.28%	**74.81%**	74.97%
Vision Transformer	1	81.36%	75.19%	70.73%	70.98%
	2	**83.44%**	78.54%	72.76%	73.95%
	3	81.84%	73.95%	72.17%	71.93%

Fig. 9 Valor das métricas em relação a modelos e experimentos.

Para a avalição por meio de métricas qualitativas, foram utilizados os modelos dos experimentos com maior F1-Score, além da implementação de uma aplicação Flask para permitir a avaliação qualitativa por meio da interface web. O código cria um aplicativo web que permite aos usuários fazer upload de imagens e obter previsões de diferentes modelos de aprendizagem profunda.

Primeiramente, é definida uma função de pré--processamento de imagens que redimensiona a imagem, converte-a para tensor e a normaliza. Em seguida, são definidos os modelos que são carregados com os pesos treinados. Cada modelo é ajustado para classificar imagens em uma das sete classes de emoções.

O aplicativo Flask possui várias rotas: a rota principal ("/") que lida com uploads de imagens e a rota de previsão ("/prediction/<filename>") que processa a imagem carregada. Ao receber uma imagem, o aplicativo a processa utilizando a função de pré-processamento e a passa pelos modelos carregados para obter as probabilidades de cada classe.

As previsões de cada modelo são obtidas utilizando a função softmax para transformar os logits em probabilidades. As três classes mais prováveis, juntamente com suas

probabilidades, são exibidas na interface web para cada um dos modelos.

A utilização da F1-Score como métrica principal é justificada pela sua capacidade de balancear precisão e recall, oferecendo uma medida mais completa do desempenho do modelo, especialmente em casos de desbalanceamento de classes. Enquanto a acurácia pode ser enganosa em conjuntos de dados desbalanceados, a F1-Score garante que tanto os falsos positivos quanto os falsos negativos sejam considerados, proporcionando uma avaliação mais robusta e confiável dos modelos. A interface web facilita a visualização e compreensão das previsões dos modelos, permitindo uma análise qualitativa complementar às métricas quantitativas.

IX. Conclusão

A conclusão deste projeto revela que, embora os modelos apresentem desempenhos similares, a ResNet18 destacou-se durante o Experimento 2, obtendo as maiores métricas em precisão e recall, além da segunda maior métrica em acurácia, perdendo apenas 0,03% para o Vision Transformer. Os experimentos mostraram que no Experimento 2 houve um aumento considerável na performance dos modelos, com exceção da VGG19. Já no Experimento 3, observou-se que apenas a VGG19 teve um aumento notável de performance, enquanto a EfficientNet teve um aumento pequeno e a ResNet18 e o Vision Transformer perderam performance. Esse declínio pode ser atribuído ao overfitting que pode ser causado por técnicas de oversampling.

A avaliação qualitativa demonstrou que os modelos apresentaram uma performance satisfatória, sendo capazes de identificar expressões estereotípicas com facilidade. No entanto, algumas previsões erradas ocorreram devido à subjetividade na classificação de expressões, o que pode ser uma área de melhoria futura.

Para futuros estudos baseados neste projeto, recomenda-se a aplicação de técnicas de ensemble learning, a experimentação de novas técnicas de data augmentation e a utilização de novos modelos. A combinação de diferentes abordagens pode proporcionar uma melhoria significativa na performance geral. Concluímos que os objetivos do projeto foram alcançados com sucesso, proporcionando conclusões valiosas sobre a aplicação de diferentes arquiteturas de redes neurais na classificação de expressões faciais.

Referências

[1] X. Li, "RAF-DB: Real-world Affective Faces Database," [Online]. Available: https://www.kaggle.com/datasets/sinanuzun/rafdb. [Accessed: Jul. 25, 2024].

[2] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA, May 2015, pp. 1-14.

[3] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, June 2016, pp. 770-778.

[4] M. Tan and Q. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, Long Beach, CA, USA, June 2019, pp. 6105-6114.Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2024). *Dive into deep learning*. Cambridge University Press.

[5] A. Krizhevsky, I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS)*, Lake Tahoe, NV, USA, December 2012, pp. 1097-1105.

[6] A. Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Proceedings of the 33rd Conference on Neural Information Processing Systems (NeurIPS)*, Vancouver, BC, Canada, December 2019, pp. 8024-8035.

[7] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2018.

.