Classificação de Imagens com Redes Neurais Artificiais*

Tiago C A Amorim (RA: 100675)[a], Taylon L C Martins (RA: 177379)[b]

[a] *Doutorando no Departamento de Engenharia de Petróleo da Faculdade de Engenharia Mecânica, UNICAMP, Campinas, SP, Brasil*
[b] *Aluno especial, UNICAMP, Campinas, SP, Brasil*

Keywords: Classificação, Redes Neurais Artificiais

1. Introdução

Este relatório apresenta as principais atividades realizadas no desenvolvimento das atividades propostas na Lista 03 da disciplina IA048: Aprendizado de Máquina, primeiro semestre de 2024. O foco deste exercício é de construir a avaliar o desempenho de redes neurais artificiais, MLP (densas de uma camada) e CNN (convolucionais), na classificação de imagens de células sanguíneas periféricas.

2. Tarefa Proposta

Nesta atividade, vamos abordar o problema de reconhecimento de células sanguíneas periféricas utilizando a base de dados BloodMNIST [1, 2, 3] (https://medmnist.com/), a qual possui 17.092 imagens microscópicas coloridas (3 canais de cor). O mapeamento entre os identificadores das classes e os rótulos está indicado na Tabela 1.

Id	Rótulo
0	Basófilos
1	Eosinófilos
2	Eritroblastos
3	Granulócitos imaturos
4	Linfócitos
5	Monócitos
6	Neutrófilos
7	Plaquetas

Tabela 1: Correspondência entre os identificadores numéricos das classes e os tipos de células sanguíneas.

(a) Aplique uma rede MLP com uma camada intermediária e analise (1) a acurácia e (2) a matriz de confusão para os dados de teste obtidas pela melhor versão desta rede. Descreva a metodologia e a arquitetura empregada, bem como todas as escolhas feitas.

(b) Monte uma CNN simples contendo:

- Uma camada convolucional com função de ativação não-linear.
- Uma camada de *pooling*.
- Uma camada de saída do tipo *softmax*.

Avalie a progressão da acurácia junto aos dados de validação em função:

- Da quantidade de *kernels* utilizados na camada convolucional;
- Do tamanho do *kernel* de convolução.

(c) Escolhendo, então, a melhor configuração para a CNN simples, refaça o treinamento do modelo e apresente:

- A matriz de confusão para os dados de teste;
- A acurácia global;
- Cinco padrões de teste que foram classificados incorretamente, indicando a classe esperada e as probabilidades estimadas pela rede.

Discuta os resultados obtidos.

(d) Explore, agora, uma CNN um pouco mais profunda. Descreva a arquitetura utilizada e apresente os mesmos resultados solicitados no item (c) para o conjunto de teste. Por fim, faça uma breve comparação entre os modelos estudados neste exercício.

3. Aplicação

A tarefa proposta foi desenvolvida em três *notebooks* Jupyter, em Python, um para cada arquitetura de rede neural utilizada: Rede MLP, Rede Convolucional *Simples* e Rede Convolucional *Profunda*. Foi feito o uso das bibliotecas *TensorFlow* [4] para montar as redes neurais e *Scikit-learn* [5] para realizar a otimização dos hiperparâmetros.

O código pode ser encontrado em https://github.com/TiagoCAAmorim/machine_learning.

3.1. Base de Dados

A base *BloodMNIST* foi construída com imagens de diferentes resoluções. Para este exercício foi escolhida a resolução (28, 28). As imagens são classificadas em 8 classes (Figura 1). A base de dados é composta por 17 092

*Relatório número 03 como parte dos requisitos da disciplina IA048: Aprendizado de Máquina.

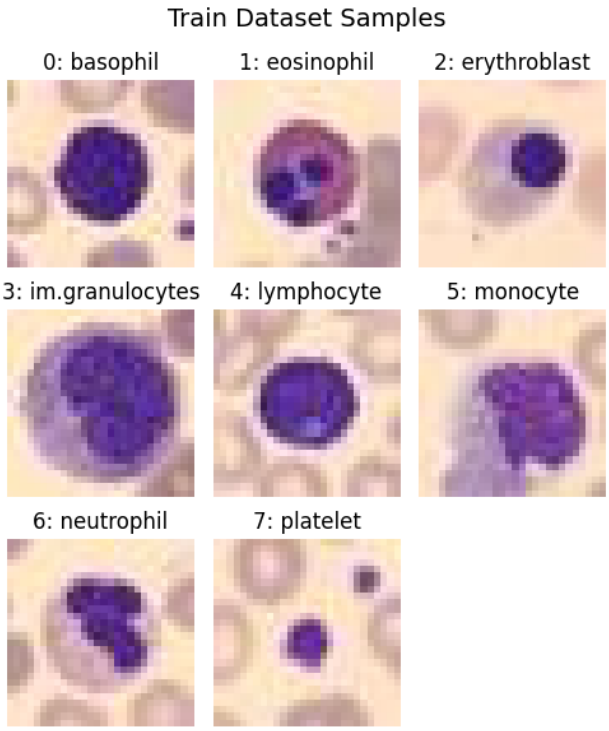

Figura 1: Exemplos de imagens por classe.

amostras, divididas em treino (11 959), validação (1 712) e teste (3 421).

Os conjuntos de dados de treino e validação não são bem distribuídos entre as classes (Figura 2). Algumas classes tem mais que o dobro de imagens que outras. Prevendo um efeito negativo no treinamento, foi proposto utilizar pesos nas diferentes classes (Figura 3). A proposta foi definir os pesos proporcionais ao inverso do número de classes. O impacto do uso de pesos por classe será avaliado para cada classificador.

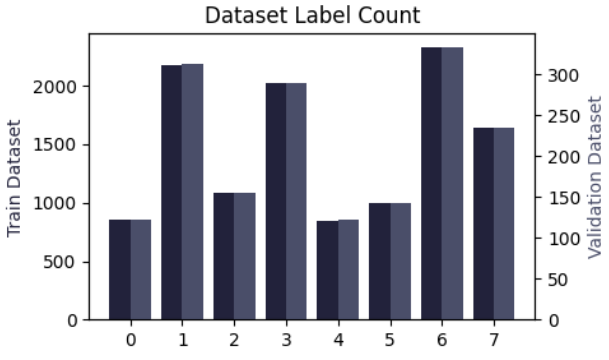

Figura 2: Número de imagens por classe.

3.2. Rede MLP

O primeiro classificador construído é uma rede neural de uma uma camada intermediária (MLP). A rede é composta por uma camada de entrada, uma camada intermediária (com função de ativação não-linear) e uma camada de saída (com função de ativação *softmax*).

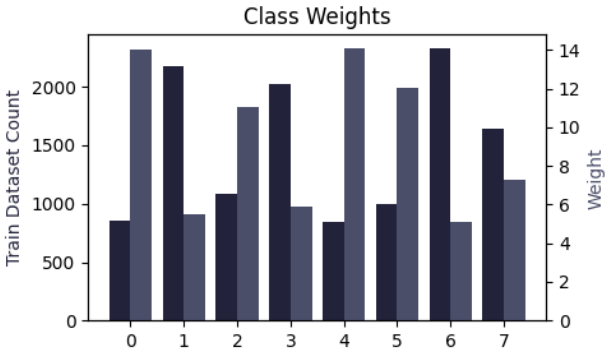

Figura 3: Pesos por classe.

Apesar de ser uma rede simples, diferentes hiperparâmetros foram avaliados para tentar encontrar um classificador mais eficiente. A avaliação dos hiperparâmetros foi feita com busca em grade (*GridSearch*).

Como o número de possíveis combinações é alto, a busca foi feita em subconjuntos de hiperparâmetros. A partir de um subconjunto de hiperparâmetros pré-estabelecido, é feita a otimização de uma parte dos hiperparâmetros. Ao final de cada passo o subconjunto de hiperparâmetros é atualizado com os valores ótimos encontrados. A Tabela 2 mostra a ordem e os subconjuntos de hiperparâmetros (separados por linhas horizontais).

Hiperparâmetro	**Opções**
Usar *data augmentation*	Sim, Não
Usar pesos por classe	Sim, Não
Número de neurônios	64, 128, 256, 512, 1024
Otimizador	SGD, RMSprop, Adam
Função de ativação	relu, tanh
Tamanho do *batch*	16, 32, 64, 128

Tabela 2: Hiperparâmetros da rede MLP (parâmetros ótimos sublinhados).

O ajuste de cada classificador foi feito por até 50 épocas, com parada antecipada (*early stopping*) caso a acurácia dos dados de validação não melhore após 10 épocas. O classificador ajustado utiliza os pesos que deram a maior acurácia dos dados de validação durante o processo de ajuste.

Na busca em grade foi aplicada validação cruzada estratificada em 3 pastas (*StratifiedKFold*) nos dados de treino, com entropia cruzada como função objetivo. A métrica de definição do melhor classificador é a média da acurácia dos dados de validação. Para adequar o custo computacional ao *hardware* disponível, a busca em grade foi limitada a 40% dos dados de treino e validação.

Figura 4 mostra o impacto do número de neurônios da camada intermediária na acurácia média com os dados de validação.

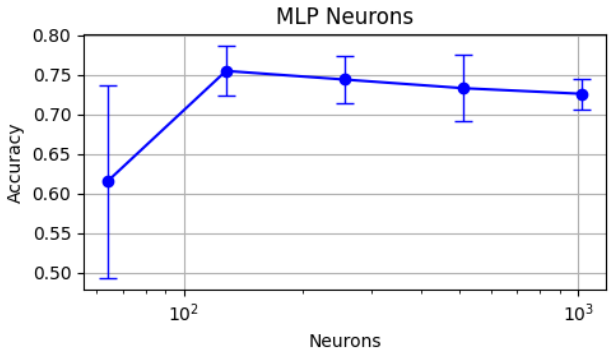

Figura 4: Efeito médio do número de neurônios da camada intermediária na acurácia da rede MLP (barras de erro são iguais ao desvio padrão estimado com validação cruzada).

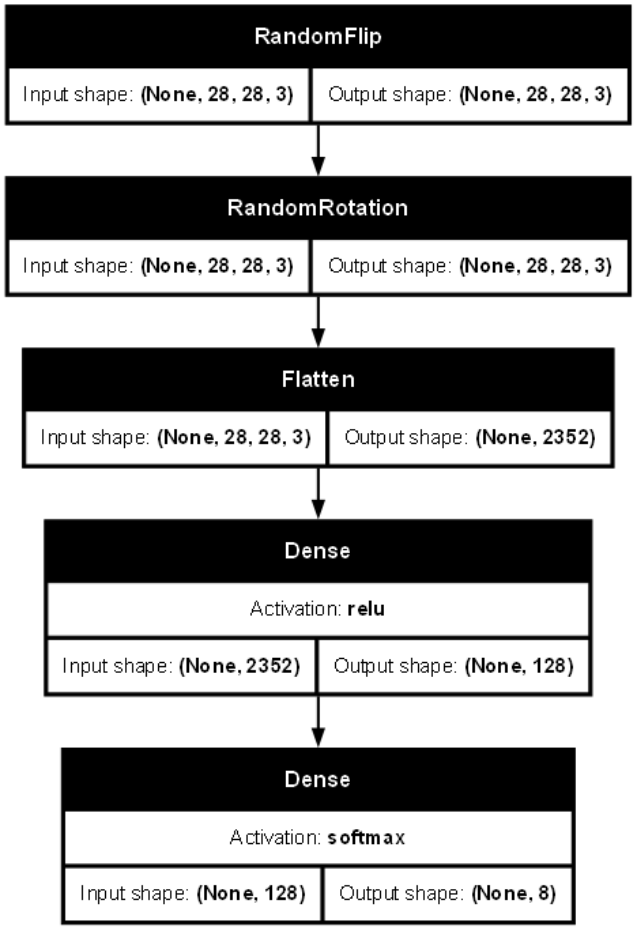

Figura 5: Modelo MLP utilizado.

Método	AUC[1]	Acurácia
ResNet-18 (28)	0.998	0.958
ResNet-18 (224)	0.998	0.963
ResNet-50 (28)	0.997	0.956
ResNet-50 (224)	0.997	0.950
auto-sklearn	0.984	0.878
AutoKeras	0.998	0.961
Google AutoML Vision	0.998	0.966

Tabela 3: Resultados reportados na publicação original [2].

3.3. Rede Convolucional Simples

3.4. Rede Convolucional Profunda

4. Análise dos Resultados

5. Conclusão

Referências

[1] J. Yang, R. Shi, B. Ni, Medmnist classification decathlon: A lightweight automl benchmark for medical image analysis, in: IEEE 18th International Symposium on Biomedical Imaging (ISBI), 2021, pp. 191–195.

[2] J. Yang, R. Shi, D. Wei, Z. Liu, L. Zhao, B. Ke, H. Pfister, B. Ni, Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification, Scientific Data 10 (1) (2023) 41.

[3] A. Acevedo, A. Merino González, E. S. Alférez Baquero, Á. Molina Borrás, L. Boldú Nebot, J. Rodellar Benedé, A dataset of microscopic peripheral blood cell images for development of automatic recognition systems, Data in brief 30 (article 105474) (2020).

[4] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, X. Zheng, TensorFlow: Large-scale machine learning on heterogeneous systems, software available from tensorflow.org (2015).
URL `https://www.tensorflow.org/`

[5] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[1] AUC = *Area Under Curve*. É a área sob a curva da Taxa de Verdadeiros Positivos (*Recall*) em função da Taxa de Falsos Positivos (1-*Especificidade*).