Aplicabilidade de uma rede neural convolucional para classificação sinalizações de trânsito: busca semi-exaustiva pelos hiperparâmetros ideais

Ronaldo D. F. Pachico[1], **Marlon F. Pereira**[1], **Igor K. Gris**[1]

[1] Ciência da Computação – Universidade Estadual do Oeste do Paraná (UNIOESTE)
Cascavel, PR – Brasil.

{ronaldo.pachico,marlon.pereira1,igor.gris}@unioeste.br

Abstract. *This paper presents the development of a traffic signs classification model using convolutional neural networks and multilayer perceptrons, preceeded by a semi-exhaustive search of hyperparameters looking for the ideal configuration of the proposed model within a reduced search scope.*

Resumo. *Este artigo apresenta o desenvolvimento de um modelo de classificação de sinalizações de trânsito com o uso de redes neurais convolucionais e perceptrons de multicamadas, precedido por uma busca semi-exaustiva dos hiperparâmetros a serem utilizados a fim de encontrar, dentre um escopo reduzido de busca, a configuração ideal para o treinamento da rede neural proposta.*

1. Introdução

O reconhecimento de sinais de trânsito, em geral, placas, desempenha um papel crucial no sistema visual de veículos autônomos. A capacidade de identificar e interpretar corretamente os sinais de trânsito é essencial para garantir a segurança e a eficiência das operações de condução autônoma. No entanto, o reconhecimento e a classificação de sinais de trânsito são tarefas complexas, especialmente quando se baseiam apenas em imagens 2D.

As imagens 2D apresentam uma série de desafios que podem afetar negativamente a precisão do reconhecimento de sinais de trânsito. Entre esses desafios estão a má iluminação, o desfoque de movimento, a deterioração das cores e a oclusão parcial. Estes fatores podem dificultar a identificação correta dos sinais de trânsito e comprometer a segurança do sistema de condução autônoma.

Este trabalho aborda especificamente o desafio do reconhecimento de sinais de trânsito com base em imagens 2D utilizando redes neurais convolucionais (CNNs). As CNNs têm se mostrado altamente eficazes em tarefas de visão computacional, especialmente no reconhecimento de padrões em imagens. Ao explorar a capacidade das CNNs de extrair e aprender características discriminativas das imagens, esperamos desenvolver um sistema robusto e preciso para o reconhecimento de sinais de trânsito em ambientes reais.

Ao longo deste trabalho, descreveremos a metodologia utilizada para treinar e testar as CNNs, bem como os resultados obtidos em relação à precisão do reconhecimento de sinais de trânsito. Visamos mostrar a aplicabilidade do uso de CNNs para o reconheminto de sinais de transito.

2. Redes neurais

As redes neurais artificiais (RNA) estão sendo amplamente aplicadas nas mais diversas áreas para resolução de diversos problemas como a classificação de problemas multivariáveis, visão computacional e processamento de sinais naturais.

2.1. Multilayer Perceptron

O perceptron, a RNA mais simples, é considerado um bom classificador binário [Aggarwal 2018]. Um único perceptron consegue separar apenas duas classes e, para funcionar adequadamente, as duas classes C1 e C2 devem ser linearmente separáveis. Isso significa que os padrões das diferentes classes a serem classificados precisam estar suficientemente distantes entre si, garantindo que a superfície de decisão seja um hiperplano [Haykin 2001].

Para resolver problemas não linearmente separáveis, pode-se usar o Multilayer Perceptron (MLP), ou Perceptron de multicamadas. Segundo [Haykin 2001], o MLP possui três características: a função de ativação deve ser não linear e sua não linearidade deve ser suave, a rede deve conter ao menos uma camada oculta e exibir um alto grau de conectividade.

Na rede neural de camada única, o processo de treinamento é relativamente simples porque o erro (ou função de perda) pode ser calculado diretamente a partir dos pesos, facilitando o cálculo do gradiente. No entanto, em redes multicamadas, a função de perda torna-se uma função composta complexa dos pesos nas camadas anteriores. Para treinar uma rede MLP, utiliza-se o algoritmo de retropropagação (backpropagation), que aplica a regra da cadeia do cálculo diferencial para calcular os gradientes de erro em termos de somas de produtos dos gradientes locais ao longo dos vários caminhos de um nó para a saída [Aggarwal 2018].

O algoritmo de retropropagação consiste em duas fases:

- **Feed Forward (avançar)**: As entradas são propagadas pelos neurônios e a saída é calculada, sendo usada como entrada para a próxima camada. Esse processo se repete até a última camada.
- **Back Forward (retroceder)**: O erro é calculado e os pesos são atualizados, começando da última camada até a primeira.

2.2. Redes convolucionais

As redes neurais convolutivas (Convolutional Neural Networks - CNN) são variações das redes MLP, com uma arquitetura inspirada na percepção visual biológica. Elas são especialmente eficazes no reconhecimento de formas bidimensionais, mantendo a invariância a distorções [Haykin 2001].

A arquitetura das Redes Neurais Convolutivas (CNN) pode ser descrita como uma sequência de diferentes camadas, cada uma com um objetivo específico. Segundo [O'Shea and Nash 2015], a arquitetura de uma CNN geralmente inclui três tipos de camadas:

- **Convolutional layers (Camadas convolucionais)**: Realizam o processo de convolução. Cada neurônio nesta camada recebe a entrada de uma seção retangular (campo receptivo) da camada anterior.

- **Pooling layers (Camadas de pooling)**: Realizam a redução da dimensionalidade espacial dos dados de entrada (sub-amostragem), diminuindo a complexidade do modelo. Em geral, essa camada recebe um grupo de valores e retorna o máximo entre eles.
- **Fully-connected layers (Camadas totalmente conectadas)**: Geralmente são as camadas finais da rede. Seus neurônios estão conectados a todos os neurônios da camada anterior e são responsáveis por calcular o resultado da classificação, similarmente às redes neurais comuns.

Sendo uma extensão do MLP, o algoritmo para o treinamento ainda é o back-propagation, antes de entrar nas camadas totalmente conectadas a rede necessita de uma camada para fazer o achamento da matriz, podendo assim ser usada na rede.

3. Metodologia

3.1. Ferramentas Utilizadas

Para o desenvolvimento do modelo, foram utilizadas as seguintes ferramentas:

- **Keras**: Keras. Segundo [Chollet et al. 2015], Keras é uma biblioteca voltada para implementação de redes neurais, em especial aos algoritmos de Deep Learning (aprendizado profundo). Desenvolvida na linguagem Python, permitindo a implementação de redes neurais de forma simplificada e eficiente. Pode ser obtida em https://keras.io, sob a licença MIT.
- **Scikit-learn**: Biblioteca de aprendizado de máquina em Python que fornece ferramentas simples e eficientes para análise de dados.
- **NumPy**: Biblioteca fundamental para computação científica em Python, oferecendo suporte para arrays multidimensionais e uma vasta coleção de funções matemáticas.

3.2. Base de Dados

Para a realização do processo de classificação proposto, utilizou-se uma base de dados pública de placas de trânsito [Sichkar 2024]. Esta é uma adaptação da base de dados GTSRB para o formato f5hd, contendo imagens de sinais de trânsito alemão. A base é composta por 51.839 imagens únicas, com resolução de 48x48 pixels em RGB. No entanto, essa versão não possui uniformidade nas classes, como pode ser visualizado na Figura 1.

A base de dados está dividida em três partes: treinamento, validação e teste, contendo 36.288, 12.440 e 3.111 imagens em cada uma dessas categorias, respectivamente. Para a criação da matriz de confusão foi usado a versão light, que torna uniforme a distribuição entre as classes.

Para que a utilização na rede fosse possível, se realizou uma normalização da base de dados, dividindo cada valor RGB por 255, assim mantendo-os entre 0 e 1.

3.3. Testes

Para definir a estrutura da rede, foi realizado um teste utilizando a função GridSearch da biblioteca Scikit-learn, que testa múltiplas combinações de hiperparâmetros para encontrar a configuração ideal, na Tabela 1 é possível visualizar quais os hiperparâmetros

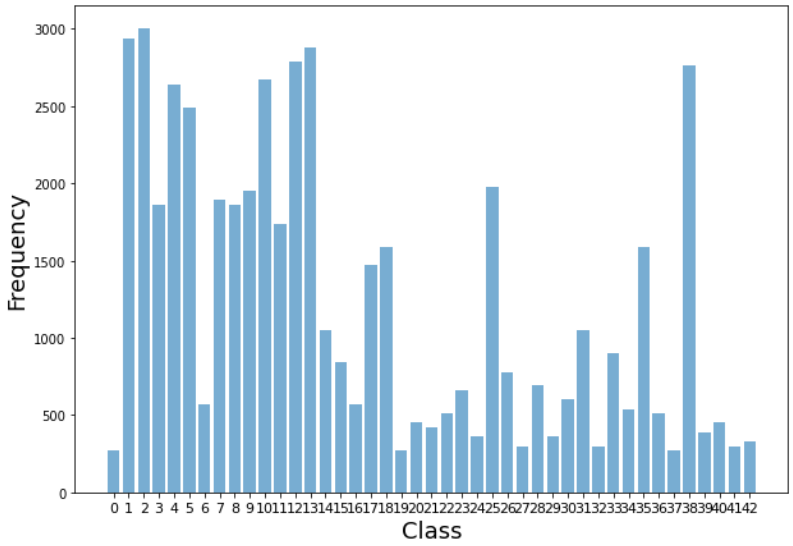

Figure 1. Histograma da versão original do conjunto de dados de sinais de trânsito.

variaram, e como variaram. A partir dos resultados gerados por essa busca foi então executada outra busca, agora para uma configuração de épocas que não cause overfiting, o melhor modelo foi submetido a teste de 10 a 50 épocas a um passo de 10.

Table 1. Hiperparâmetros e suas variações

Hiperparâmetro	Variações
Batch Size	50, 100
Optimizer	SGD, Adam
Conv Layers	1, 2
Conv Filters	16, 32
Dense Layers	1, 2
Dense Units	128, 256

Os demais hiperparâmetros da rede foram definidos sem a necessidade de testes, a função de ativação nas camadas ocultas foi a ReLU, pois segundo [Schmidt-Hieber 2020] ela supera estatisticamente outras funções como a sigmóide. Na camada de saída foi utilizada a função SoftMax, a qual é geralmente utilizada em problemas de classificação onde as classes são mutuamente exclusivas e deseja-se obter um grau de confiança de cada classe predita. A função de loss foi a categorical cross entropy (entropia cruzada categórica). A taxa de aprendizado foi mantida em seu valor padrão (0.001) e as épocas foram mantidas em cinco para a definição da estrutura da rede com o GridSeach.

As métricas avaliadas compreendem a acurácia, que representa a proporção entre as previsões corretas e o número total de exemplos, e a loss (perda), que mensura a discrepância entre as previsões e os rótulos, isto é, as respostas corretas.

A estrutura da rede neural consistirá inicialmente em uma camada de convolução como camada de entrada, que receberá uma imagem de dimensão 48 por 48 pixels em três canais RGB. Posteriormente, o modelo pode incluir mais camadas convolucionais e de pooling, dependendo dos resultados dos testes realizados pelo GridSearch.

Após as camadas convolucionais e de pooling, será utilizada uma camada de achatamento para mapear a saída das camadas anteriores para a rede totalmente conectada. A rede totalmente conectada incluirá uma ou mais camadas ocultas, cujo número e configuração serão determinados pelos resultados do GridSearch.

Finalmente, a camada de saída terá 43 neurônios, representando uma classe para cada tipo de sinal de trânsito na base de dados. Essa estrutura flexível permitirá ajustar a arquitetura da rede com base na otimização dos hiperparâmetros e nos resultados dos testes realizados.

4. Resultados

Com base nos resultados do GridSearch, foi determinado o modelo ideal entre várias combinações possíveis. A estrutura final da rede neural consiste em uma camada de convolução com 32 filtros, seguida por uma camada de MaxPooling2D de tamanho 2x2. Em seguida, há uma camada de achatamento (Flattening), seguida por duas camadas ocultas na rede totalmente conectada, cada uma com 256 neurônios. Por fim, a camada de saída é composta por 43 neurônios, correspondendo a cada classe na base de dados. Essa arquitetura foi selecionada com base na otimização dos hiperparâmetros e nos resultados dos testes realizados.

Em posse do melhor modelo testado, foi realizado o teste de épocas para determinar em que momento a rede deveria parar, os resultados desse teste podem ser observados nas Figuras 2 e 3. Observa-se um aumento na acurácia à medida que o número de épocas aumenta. Entretanto, é importante notar que após 50 épocas, a loss começa a subir, sugerindo a possibilidade de overfitting na rede neural. A diferença entre 40 e 50 épocas é pequena, o que pode indicar que 50 épocas não oferece benefícios significativos em relação a 40. Portanto, considerando esses resultados, o número ideal de épocas pode ser estabelecido em 40, sendo esta a solução final adotada neste trabalho.

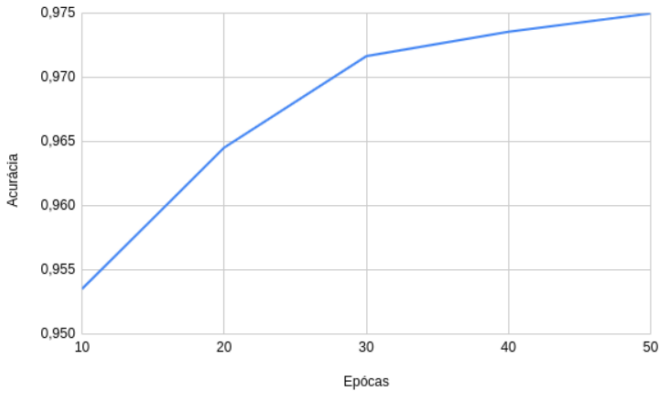

Figure 2. Gráfico de épocas por acurácia.

Dado a estrutura da rede definida pelo GridSeach e o número de épocas, foi salvo o modelo e realizado o teste da matriz de confusão na base de dados em sua versão light que tem a todas as classes em uma distribuição uniforme, a matriz resultante pode ser vista na Figura 4. Um exemplo de onde a rede ainda apresenta dificuldades em classificar de forma adequada é a classe 6, que ela classifica como 32. Essas placas são as ”Fim

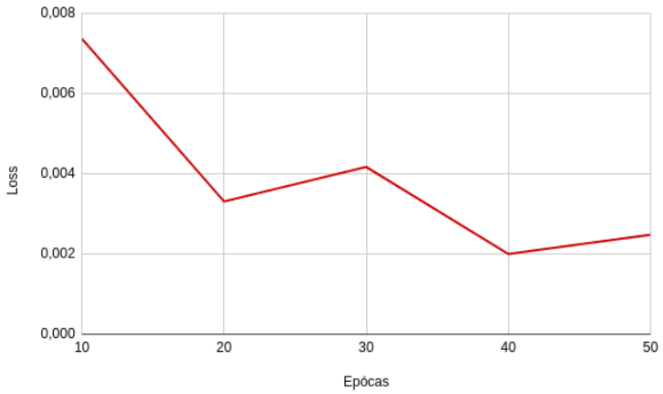

Figure 3. Gráfico de épocas por loss.

do limite de velocidade de 80km/h" e "Fim de todos os limites de velocidade", como pode ser visualizada na Figura 5, essas duas placas são escuras e praticamente idênticas, variando apenas o número 80 dentro delas.

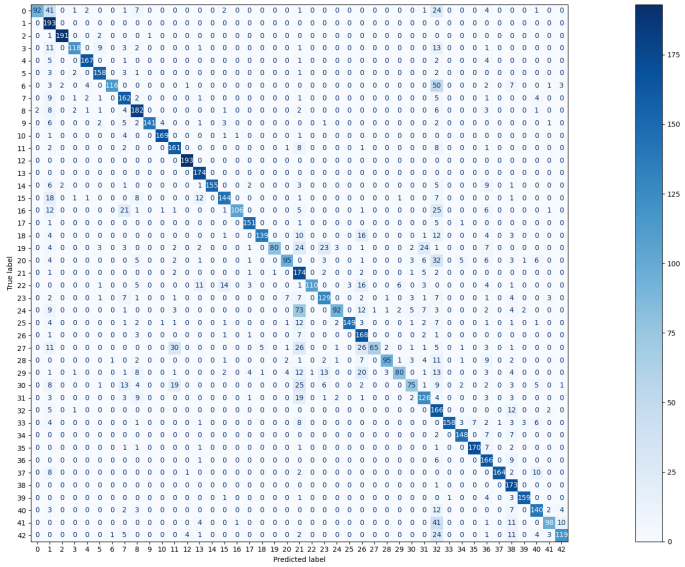

Figure 4. Matriz de confusão do modelo treinado em 40 épocas e aplicado aos dados de teste da base de dados em sua versão light.

5. Conclusões

Os resultados obtidos evidenciam a capacidade de uma rede neural convolucional para a classificação de imagens. Por meio de uma breve análise buscando o refinamento dos hiperparâmetros utilizados foram obtidas uma alta acurácia e baixa perda. Entretanto, a classificação de sinalizações de trânsito é uma tarefa crítica que deve ser realizada em tempo real em condições muito mais adversas que as apresentadas pelo banco de dados, e erros como o apresentado pela Figura 5 devem ser considerados inaceitáveis.

A fim de melhorar os resultados obtidos, os valores de acurácia e perda podem ser aprimorados por meio de uma busca por hiperparâmetros mais robusta e pelo uso de um banco de testes mais amplo, objetivos que podem ser contemplados em trabalhos futuros.

Figure 5. Representação de duas placas comumente confundidas pelo modelo de rede neural treinado.

References

Aggarwal, C. C. (2018). *Neural Networks and Deep Learning*. Springer, New York.

Chollet, F. et al. (2015). Keras. `https://github.com/keras-team/keras`.

Haykin, S. (2001). *Redes Neurais: Princípios e Prática*. Bookman, Porto Alegre.

O'Shea, K. and Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with relu activation function. *Annals of Statistics*, 48(4):1875–1897.

Sichkar, V. (2024). Traffic signs 1 million images for classification. `https://www.kaggle.com/datasets/valentynsichkar/traffic-signs-1-million-images-for-classification`.