Detecção de Rótulos de Risco

1st Marcus Vinícius de Faria Santos
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
mvfs@cin.ufpe.br

2nd Lucas Nascimento Brandão
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
lnb@cin.ufpe.br

3rd Jeferson Severino de Araújo
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
jsa2@cin.ufpe.br

4th Rodrigo Rocha Moura
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
rrm2@cin.ufpe.br

5th Matheus Júlio Boncsidai de Oliveira
Centro de Informática
Universidade Federal de Pernambuco
Recife, Brasil
mjbo@cin.ufpe.br

***Resumo*—Este documento tem por objetivo a explicação, de modo sucinto, sobre a utilização de uma rede neural treinada para detecção de objetos e sobre a avaliação, que diferentes dados para o treinamento, terão no impacto da rede. Ele foi desenvolvido para a disciplina ES413 - Sinais e Sistemas e será dividido em 'Introdução', 'Desafios', 'Motivação', 'Metodologia', 'Resultados', 'Conclusões' e 'Referências'.**

***Palavras-chave*—Rede Neural, Detecção de objetos, treinamento.**

I. Introdução

Em recipientes que contêm materiais perigosos, é comum encontrar placas de sinalização que indicam a presença dessas substâncias. Essas placas são importantes para garantir a segurança das pessoas que manuseiam esses materiais. No entanto, uma vez as informações nessas placas são mal compreendidas, acidentes e tragédias tornam-se possíveis de acontecer.

Desse modo, levando em conta esse contexto e a crescente utilização de veículos autônomos, é essencial que esses sistemas sejam capazes de identificar e reagir a ambientes com materiais perigosos.

Portanto, o projeto tem como objetivo o desenvolvimento de uma rede neural capaz de classificar placas de sinalização de risco. O processo de aprendizado dessa inteligência artificial utiliza-se de um extenso banco de imagens, de filtros estatísticos e da Transformada de Fourier.

II. Desafios

Inicialmente houve uma certa dificuldade de achar filtros que pudessem ser utilizados no domínio da frequência de uma imagem, além da própria manipulação da imagem em sua forma complexa. Porém, após algumas pesquisas e testes, foi possível implementar dois tipos de filtros desejados. O primeiro possui o objetivo de borrar a imagem, já o segundo aplica um ruído na imagem. Além disso, notou-se que o filtro de ruído não trouxe o resultado esperado, pois não fez o efeito de corrosão da imagem, apenas deixando-a menos nítida e, em alguns casos, alterando o espectro de cores da imagem.

III. Motivação

A principal motivação do nosso trabalho é a redução no número de acidentes evitáveis por meio da identificação de rótulos de risco em situações reais. Uma aplicação prática seria a identificação desses rótulos em transportadores de carga por veículos autônomos, com objetivo de definir quais comportamentos e decisões devem ser tomadas quando transitando próximo a algum tipo específico de material. Com isso desenvolvido, teríamos uma maior segurança e confiabilidade no trânsito de veículos autônomos em vias públicas, ou mesmo em setores internos de uma empresa.

IV. Metodologia

Para a resolução do problema proposto, optamos por utilizar uma rede neural. A escolhida foi a *MobileNetV2*, por ser uma rede que um dos membros do projeto já tem certa afinidade e por ser uma rede leve e facilmente configurada em computadores embarcados. O segundo fator de escolha é de extrema importância para o projeto, pois a ideia original é que nossa solução fosse capaz de rodar em carros autônomos e, caso escolhêssemos uma rede mais robusta e, portanto, computacionalmente mais exigente, não conseguiríamos essa parte da solução.

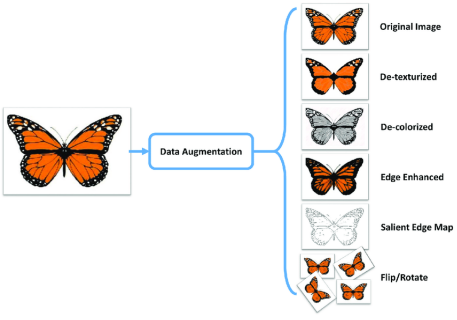

Fig. 1. Exemplo de Data Augmentation

A rede foi treinada com um *dataset* já existente, o *HAZMAT13* [1]. Essa fonte contém 1686 imagens sem melho-

ramento, com *bounding boxes* e máscaras segmentadas, das quais estas últimas não foram utilizadas. São diversas imagens de placas e sinais de riscos, com diferentes iluminações e ambientes, divididas em 13 classes de risco: tóxico, oxidante, gás inflamável, sólido inflamável, corrosivo, perigoso, gás não inflamável, peróxido orgânico, explosivo, radioativo, risco de inalação, combustão espontânea, substância infecciosa. Exemplos do banco de imagens são vistos nas Figuras 1 e 2.

Fig. 2. Exemplo 1 de imagem do *dataset* original.

Fig. 3. Exemplo 2 de imagem do *dataset* original.

A. Filtro Estatístico

Utilizamos aqui filtros lineares, que, ao contrário dos filtros espacialmente invariantes, se comportam de diferentes maneiras dependendo da partição da imagem, essa diferença acontece porque os valores dos pixeis são definidos a partir de combinações lineares. Para Szeliski [3] (2021), filtros lineares são um tipo de *neighborhood operator*, os valores dos pixeis são calculados pela soma ponderada dos pixeis em seu entorno.

O filtro estatístico, tipo de filtro linear, utilizado neste projeto foi um filtro para detecção de arestas, mais especificamente, o filtro Sobel. Este operador lança mão de duas matrizes 3×3 que são convolvidas com a imagem original para calcular aproximações das derivadas. Uma derivada é para as variações horizontais, já a outra é para as verticais. Esses valores eventualmente se unem para a formação de uma imagem com ambas as arestas verticais e horizontais em destaque.

As matrizes 3x3 que utilizamos para realce das arestas horizontais e verticais. respectivamente, foram:

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

A aplicação do filtro estatístico é vista nas Figuras 3 e 4.

Fig. 4. Figura 1 após a aplicação do filtro estatístico.

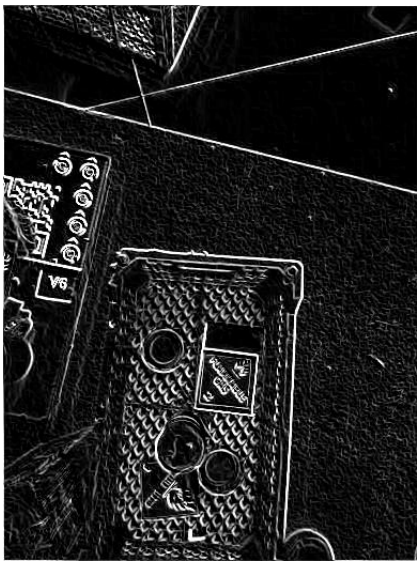

Fig. 5. Figura 2 após a aplicação do filtro estatístico.

B. Transformada de Fourier [2]

Com o objetivo de complementar o banco de imagens de treino da rede neural, utilizamos da transformada de Fourier para aplicar filtros nas imagens no domínio da frequência. A transformada de Fourier utilizada nos códigos é a da biblioteca *numpy*, a qual é implementada usando o algoritmo da Transformada Rápida de Fourier (FFT).

Na primeira metade do banco de imagens original aplicamos o filtro de borramento (filtro *low pass*, filtro passa baixa). Esse filtro consiste em, no domínio da frequência, eliminar as frequências altas. Visualmente, isso implica em uma imagem borrada, ou seja, com os contornos das placas menos definidos. A aplicação do filtro de borramento é vista na Figura 5.

Na segunda metade do banco de imagens original aplicamos o filtro de ruído. Esse filtro consiste em, no domínio da frequência, multiplicar a imagem por uma matriz ruído (matriz preenchida por números aleatórios). Visualmente, isso implica em alterar as cores dos rótulos de risco e até escurecer a imagem. A aplicação do filtro de ruído é vista na Figura 6.

Fig. 6. Figura 1 após a aplicação do filtro de borramento.

Fig. 7. Figura 2 após a aplicação do filtro de ruído.

V. Resultados

Para avaliar o impacto que os filtros e a transformada tiveram no desempenho da rede, treinamos três redes distintas. Suas descrições, bem como o resultado na classificação podem ser encontrados abaixo:

A. HM v1

Nesta, somente foram utilizadas para treino as 1685 imagens originais do dataset. No notebook utilizado para o treinamento, demorou aproximadamente 1 minuto por época usando GPU (passar por todas as imagens e atualizar os pesos - ”aprender”) ou 20 minutos usando CPU. A rede foi treinada por aproximadamente 120 épocas.

Os resultados obtidos pela rede HM v1 são vistos na Figura 8.

```
Average Precision Per-class:
    poison: 0.9004329004329005
    oxygen: 0.9870129870129871
    flammable: 0.9500067500389366
    flammable-solid: 0.992822966507177
    corrosive: 0.9970674486803521
    dangerous: 0.9886738179421108
    non-flammable-gas: 0.9896103896103898
    organic-peroxide: 0.9946524064171125
    explosive: 0.9548228580486647
    radioactive: 0.9786357786357788
    inhalation-hazard: 0.9786354246970685
    spontaneously-combustible: 1.0000000000000002
    infectious-substance: 0.9939393939393941
Mean Average Precision (mAP):  0.9774087016894518
```

Fig. 8. Resultados HM v1

B. HM v2

Para essa versão, foram utilizadas mais imagens para o treinamento. Aqui usamos o princípio de *augmentation*, ou seja, variamos a imagem com o objetivo de obrigar a rede a procurar pelos parâmetros que desejamos, evitando, por exemplo, que ela decore uma placa pela cor dela. Aqui, além das 1685 originais, foram adicionadas mais 1685 com um filtro estatístico aplicado diretamente e outras 1685 em que primeiro foi feita a conversão para o domínio da frequência e, depois, foi passado um filtro, no caso metade para o de borramento e metade para o de ruído. Aqui, cada época demorou 2.5 minutos e, ela também foi treinada por 120 épocas.

Os resultados obtidos pela rede HM v2 são vistos na Figura 9.

C. HM v3

Por fim, fizemos uma última versão em que treinamos com um *dataset* com 52845 imagens. Nesse caso, as imagens passaram por vários processos de *augmentation*, mas não foram definidos pelo grupo, mas sim pelos criadores originais do artigo [1] em que nos inspiramos. Aqui, cada época passou a demorar 27.5 minutos, com isso, não conseguimos treinar a rede por 120 épocas como as demais, mas sim por 57 épocas.

Os resultados obtidos pela rede HM v3 são vistos na Figura 10.

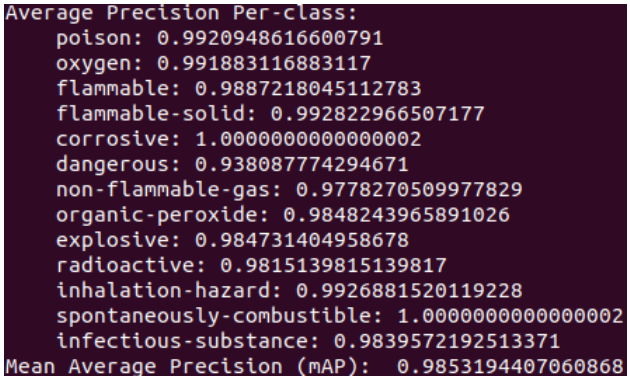

Fig. 9. Resultados HM v2

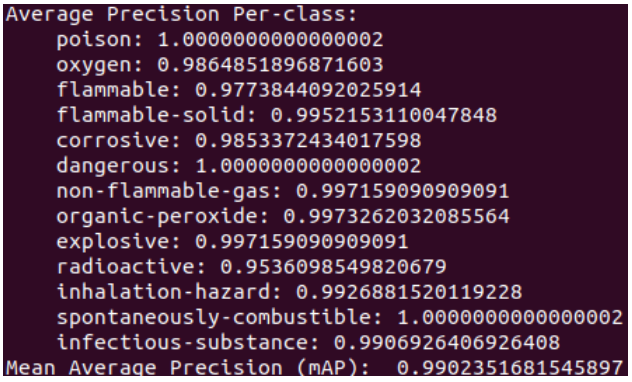

Fig. 10. Resultados HM v3

VI. Conclusões

O objetivo desse projeto era a construção de um detector de placas que pudesse classificar e identificar rótulos de risco com uma certeza razoável e fosse capaz de rodar em computadores embarcados. Usando a rede *MobileNetV2* conseguimos obter uma rede que atende esses requisitos, sendo ela capaz de ser rodada em uma *Jetson* Nano, portanto, capaz de rodar em um carro.

Por meio da análise dos resultados das três versões da rede, também foi possível perceber que as redes que contam com estratégias de *Data Augmentation* apresentam, em geral, resultados melhores. A versão 1 da rede, que apresenta apenas as imagens originais tem o pior resultado, a rede 2 com alguns filtros, alguns feitos no domínio da frequência e depois retornados ao domínio da imagem, apresentou resultados intermediários e a rede com uma forte utilização de *Data Augmentation* apresentou o melhor resultado.

Com isso, fica evidente que não é necessário tirar novas fotos para o treinamento, se podemos apenas variar as que já temos, o que enfatiza a importância de se estudar sobre filtros e formas de transformar as imagens, como a transformada de Fourier.

Referências

[1] A. M. R. Laboratory, "HAZMAT-13: Hazaradous Materials Signs Dataset", GitHub. Available: https://github.com/mrl-amrl/HAZMAT13. [Accessed: 15-Abr-2023].

[2] OPENCV. OpenCV: image transforms in opencv. Fourier Transform. Available: $https://docs.opencv.org/3.4/de/dbc/tutorial_py_fourier_transform.html$ [Accessed: 13-Abr-2023].

[3] SZELISKI, Richard. Computer Vision: Algorithms and Applications. 2 ed. Springer, 2022. Available: https://szeliski.org/Book. [Accessed: 17-Abr-2023].