Um Estudo sobre Redes Neurais Convolucionais e sua Aplicação em Detecção de Pedestres

Ana Caroline Gomes Vargas, Aline Paes e Cristina Nader Vasconcelos
Instituto de Computação
Universidade Federal Fluminense
Niterói, Brasil
Email: carol.gomes.vargas@gmail.com, alinepaes@ic.uff.br, crisnv@ic.uff.br

***Resumo*—Este trabalho aborda o tema de Aprendizado Profundo com um olhar da área de Visão Computacional, traçando um paralelo entre as duas áreas para a tarefa de detecção de pedestres. Para tanto, os resultados de duas abordagens clássicas de detecção de pedestres são comparados experimentalmente com uma abordagem de Redes Neurais Convolucionais, cujos resultados são considerados o estado da arte para o problema em questão.**

***Keywords*-Detecção de Pedestres, Visão Computacional, Aprendizado Profundo, CNN, SVM, HOG, Adaboost, Haar features**

***Abstract*—This work addresses Deep Learning from the point of view of the Computer Vision area, drawing a parallel between the two areas, considering the pedestrian detection task. To achieve that, the experimental results of two classical Computer Vision approaches are compared to the results obtained from a Convolutional Neural Network, which is known for obtaining the state of the art to the problem of pedestrian detection.**

***Keywords*-Pedestrian Detection, Computer Vision, Machine Learning, CNN, SVM, HOG, Adaboost, Haar features**

I. Introdução

O Aprendizado Profundo com Redes Neurais é um tema bastante discutido e difundido atualmente, devido ao seu grande sucesso na obtenção de resultados no estado da arte para diversos problemas, principalmente na área de Visão Computacional. Uma das principais abordagens utilizadas são as Redes Neurais Convolucionais[1] (ou *Convolutional Neural Network* - CNN).

Esse trabalho busca visitar os conceitos abordados em Aprendizado Profundo sob o ponto de vista da Visão Computacional. Para efeito de comparação, abordaremos o problema de detecção de pedestres usando duas técnicas clássicas e uma Rede Neural Convolucional, que apresenta os melhores resultados neste problema.

Contribuições: Uma das contribuições esperadas deste trabalho é estabelecer um material didático introdutório sobre as Redes Neurais Convolucionais e comparar os resultados desta com duas técnicas tradicionais de Visão Computacional, tendo como foco o problema de detecção de pedestres.

A. Trabalhos Relacionados

Apesar da utilização de CNNs estar cada vez mais popular, seu processo de apredizado ainda não é completamente compreendido, do ponto de vista teórico. Por isso diversas pesquisas foram realizadas para se tentar elucidar o que as redes aprendem como em [2], [3].

Em relação ao problema de detecção de pedestres, diversos trabalhos propuseram arquiteturas específicas para a tarefa. Em [1], Redes Neurais Convolucionais foram utilizadas com múltiplos estágios de filtros para a detecção de pedestres e, no mesmo ano, [4] propuseram uma arquitetura de rede neural que trata deformações e oclusões de partes do pedestre.

Para este trabalho escolhemos como foco de pesquisa o artigo apresentado por J. H. Hosang et al. [5], no qual os experimentos serão detalhados na seção IV. Os mesmos autores também relacionaram os avanços em 10 anos de pesquisa na tarefa de detecção de pedestres em [6].

II. Redes Neurais Convolucionais

Uma Rede Neural Convolucional é uma variação das redes de Perceptrons de Múltiplas Camadas, tendo sido inspirada no processo biológico de processamentos de dados visuais. De maneira semelhante aos processos tradicionais de visão computacional, uma CNN é capaz de aplicar filtros em dados visuais, mantendo a relação de vizinhança entre os *pixels* da imagem ao longo do processamento da rede. A Figura 1 ilustra uma CNN. Este tipo de rede vem sendo amplamente utilizada, principalmente nas aplicações de classificação, detecção e reconhecimento em imagens e vídeos.

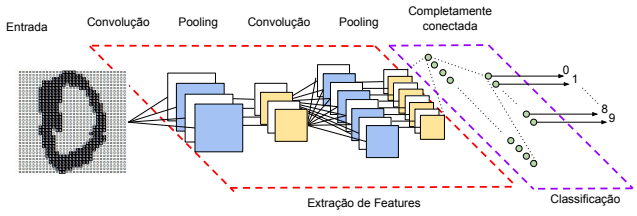


Figura 1. Exemplo de uma rede neural convolucional e suas diferentes camadas

Uma rede neural convolucional consiste em múltiplas partes com funções diferentes. Inicialmente é comum aplicar sobre o dado de entrada camadas ditas de convolução. Uma camada de convolução é composta por diversos neurônios, cada um responsável por aplicar um filtro em um pedaço específico da

imagem. Podemos imaginar cada neurônio sendo conectado a um conjunto de *pixels* da camada anterior e que a cada uma dessa conexões se atribui um peso. A combinação das entradas de um neurônio, utilizando os pesos respectivos de cada uma de suas conexões, produz uma saída passada para a camada seguinte. Os pesos atribuídos às conexões de um neurônio podem ser interpretados como uma matriz que representa o filtro de uma convolução de imagens no domínio espacial (conhecido também como *kernel* ou máscara).

Enquanto na formulação de perceptrons clássica um neurônio é conectado a todos os neurônios da camada anterior, dito completamente conectada, nas CNNs apenas um subconjunto de entradas é conectado a cada neurônio. Com a mudança de arquitetura, as redes neurais convolucionais passam a realizar análise de campos receptivos locais (*local receptive fields*). Os neurônios da mesma camada são agrupados em mapas. Um mapa é produzido pelo agrupamento das saídas de neurônios que juntos cobrem uma parte da imagem que tenha sido processado com um filtro em comum. Para que um conjunto de neurônios de um determinado mapa aplique o mesmo filtro em diferentes posições da imagem, é feito um compartilhamento dos pesos durante o processo de treinamento. Tal compartilhamento diminui significativamente o número de parâmetros a serem aprendidos e o tempo de treinamento da rede, consequentemente. De maneira a simplificar a esquematização dessas redes, um filtro a ser aplicado na imagem é representado visualmente por apenas um neurônio, trazendo a falsa sensação de que ele faria sozinho o processo de percorrimento da imagem.

O tamanho do filtro define o tamanho da vizinhança que cada neurônio da camada irá processar. Outra variável importante para a camada convolucional é o passo (ou *stride*). Este valor representa quantos *pixels* serão pulados entre cada janela, dizendo-nos qual será o tamanho da camada seguinte na mesma unidade.

Diferentemente do processo tradicional de Visão Computacional, no qual o modelo inicial parte da definição de filtros ou *features* a serem utilizadas, nas camadas de convolução das CNNs é preciso definir somente a arquitetura dos filtros: quantidade e tamanhos, *stride*, dos filtros por camada. Não é indicado em nenhum momento qual filtro deve ser utilizado. O processo de aprendizado da rede altera os pesos ao longo do treinamento, até encontrar os melhores valores dos filtros para o conjunto de dados utilizado.

Por ser um processo automático, diversas pesquisas foram realizadas para se tentar elucidar o que as redes aprendem [2]. Uma das formas de se avaliar o aprendizado é verificar as ativações dos filtros obtidos no final do treinamento [2]. Em redes voltadas para a detecção de objetos, pode-se perceber que as primeiras camadas costumam aprender filtros de arestas, bordas e cores. Analisando níveis mais profundos, detectam-se pedaços de interesse e detalhes cada vez mais complexos. Desta forma a rede produz uma hierarquia de *features*.

Outra diferença notável nas CNNs é a capacidade de criar filtros nD. Como os filtros são definidos com o processo de treinamento da rede, é possível definir filtros com largura, altura e um parâmetro de profundidade atravessando múltiplos canais. Filtros clássicos de processamento de imagens dificilmente possuem tal característica, a grande maioria deles é $2D$. A possibilidade das redes neurais convolucionais conseguirem misturar múltiplos mapas de *features* ao mesmo tempo as permite extrair características cada vez mais complexas, sendo um grande diferencial deste modelo, em comparação com técnicas clássicas. Assim, elas são capazes de sozinhas criar filtros extremamente complexos aproveitando ao máximo as informações provenientes dos dados de treinamento.

É comum, logo após a convolução, aplicar uma função de ativação. A função de ativação presente em cada neurônio é responsável por aplicar uma transformação nos dados recebidos. Normalmente, utiliza-se funções com algum grau de não-linearidade. A não linearidade das camadas intermediárias permite que as aplicações sucessivas dessas distorções tornem as categorias de saída linearmente separáveis.

Outra camada muito importante comumente utilizada após as camadas de convolução e ativação é a camada de agrupamento (*pooling*). A função dessa camada é reduzir a dimensionalidade dos dado na rede. Essa redução é importante por questão de agilidade no treinamento, mas principalmente para criar invariância espacial. A camada de *pooling* funciona agrupando um conjunto de dados, por exemplo: a entrada é dividida em janelas 4x4 e de cada uma é selecionado um valor para os representar. Essa escolha pode ser feita por diversas funções, porém a mais utilizada é a função de máximo. É importante mencionar que a camada de *pooling* não reduz a profundidade da entrada, ela apenas reduz a altura e a largura de um mapa.

As diferentes arquiteturas repetem e combinam essas e outras funções de transformações do sinal. Por exemplo a LeNet-5 [7] possui duas camadas de convolução seguidas de *pooling* e mais uma camada de convolução. A arquitetura denominada de GoogLeNet [8] possui cinco camadas de convolução sempre seguidas de um *pooling*. Pode-se pensar que o resultado da repetição dessa camadas é um conjunto de *features* especializado na tarefa na qual ela foi treinada. As próximas camadas da rede fazem o papel dos algoritmos de aprendizado de máquina tradicionalmente aplicadas para classificar os dados ou aplicar algum tipo de regressão. A Figura Figura 1 ilustra esta separação entre as camadas que fazem a extração de *features* das que as classificam.

Quando se deseja realizar uma tarefa de classificação, é acrescentada após o conjunto das camadas de convolução e *pooling* ao menos uma camada totalmente conectada. Esta camada é responsável por traçar um caminho de decisão a partir das resposta dos filtros vindos das camadas anteriores, para cada classe de resposta.

Depois da camada completamente conectada, o último passo é a função de classificação. Essa camada é fundamental no treinamento, pois influencia no aprendizado dos filtros e consequentemente no resultado da rede. Devido a sua simplicidade e bons resultados a função SoftMax pode ser utilizada nessa etapa, contribuindo para um treinamento mais rápido sem perda de qualidade.

III. O problema de detecção de Pedestres

Há mais de uma década estuda-se o problema de encontrar regiões em imagens digitais ocupadas por pedestres, chamado de detecção de pedestres. Para essa análise, são consideradas imagens de pedestres aquelas que contém seres humanos em pé. Atualmente, motivados pela grande importância e utilidade desse problema para aplicações de segurança, robótica, carros automatizados, entre outras, o aumento na precisão dos resultados divulgados em trabalhos recentes tem sido significativo.

Neste trabalho, utilizaremos a base Caltech [9] pois a maioria dos trabalhos recentes publicados para o problema de detecção de pedestres a utiliza e por ser a base principal utilizada no artigo [5] tomado como base para nossos experimentos e comparações.

IV. Experimentos

Compararemos os resultados obtidos a partir de duas técnicas clássicas (*Features* de Haar e Histograma de Gradientes Orientado - HOG [10]) com os resultado de uma CNN, definida em[5].

Frisamos que o objetivo aqui não é mostrar uma melhora de resultados em relação ao estado da arte da detecção de pedestres, mas sim fazer uma comparação entre resultados de abordagens clássicas com um dos últimos trabalhos publicados utilizando Redes Neurais Convolucionais.

A. Técnicas Clássicas

Para ambos os casos foram usados os algoritmos de treinamento da biblioteca de visão computacional OpenCV [11] para a linguagem Python.

A abordagem com as *Features* de Haar e com o classificador AdaBoost [12] não foi proposta originalmente com o objetivo de detecção de pedestres, mas sim para faces. A despeito disso, utilizaremos a mesma solução para nosso problema e avaliaremos os resultados. Os parâmetros utilizados para o treinamento da cascata do AdaBoost foram: 5 estágios (classificadores em cascata) 5, com a taxa de mínima de acerto e a taxa máxima de falso alarme sendo estimada pelo número de estágios.

A abordagem do descritor HOG juntamente com o classificador SVM [10] é uma solução proposta diretamente para o problema de detecção de pedestres. Os parâmetros utilizados no descritor HOG foram os mesmos do trabalho original: *pixels* por célula 8×8, passo 8×8, células por bloco 2×2 e 9 orientações no histograma de gradientes.

B. Redes Neurais Convolucionais

A estratégia utilizada por [5] e que replicaremos aqui é de utilizar arquiteturas, e os pesos treinados, de redes propostas para tarefa mais geréricas, como de distinção entre 1000 classes de objetos, e fazer uma calibragem, conhecida como *fine tuning*, dos pesos treinados para a tarefa.

Embora em [5] duas diferentes arquiteturas tenham sido utilizadas, a saber a CifarNet e a AlexNet, aqui replicamos apenas a abordagem com a rede AlexNet, uma vez que esta apresentou melhores resultados [5]. A rede AlexNet, ilustrada na Figura Figura 2, possui aproximadamente 6×10^7 parâmetros. Utilizamos a implementação fornecida no projeto *open source* Caffe [13], integrado ao Digits 2.2 [14]. Originalmente, a AlexNet foi treinada para a tarefa de classificação de imagens disponibilizadas na base ImageNet [15]. Para o problema de detecção de pedestres foi feito um *fine tuning* utilizando a base de dados para detecção de pedestres. Além disso, depois do ajuste, a saída SoftMax foi substituída por um SVM Linear.

Tabela I
Sensitividade e Especificidade de cada abordagem.

Abordagem	Sensitividade	Especificidade
Haar + AdaBoost	**99,62%**	57,41%
HOG + SVM	94,36%	94,96%
CNN	98,94%	**99,17%**

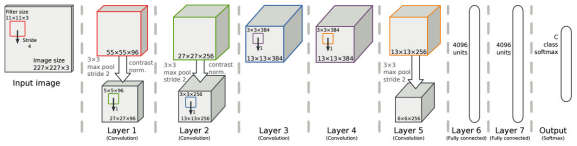


Figura 2. Arquitetura da rede Alexnet [5].

V. Resultados

Em relação aos testes, todas as técnicas foram testadas com uma porção da base Caltech já previamente recortada em casos positivos, originando 21084 exemplos de pedestres, e 22384 exemplos negativos, ou seja os casos de fundo. Para facilitar a leitura, adotaremos a seguinte nomenclatura para as abordagens:

- *Features de Haar* com AdaBoost: Haar ou técnica 1;
- HOG com SVM: HOG ou técnica 2;
- Rede Alexnet: CNN ou técnica 3;

Com relação a sensitividade e a especificidade, quanto mais próximas de 100% melhor é o classificador. Porém, para este problema, a sensitividade deve ser mais valorizada, uma vez que esta favorece o acerto na detecção de um pedestre, enquanto a especificidade favorece a classificação correta de fundo. A Tabela I mostra os resultados de cada abordagem. Como pode ser observado, todas as técnicas obtiveram excelentes resultados no quesito sensitividade, sendo o melhor valor obtido pela abordagem com as *Features* de Haar. Porém, quando avaliada a especificidade, tanto o HOG quando a CNN obtiveram bons números, mas para o caso do Haar o resultado foi muito ruim. Neste dois quesitos a técnica 3 foi a que apresentou o melhor conjunto de resultados.

A Tabela II mostra o resultado de cada técnica considerando as medidas de falso positivo e falso negativo. Para o caso da técnica 1, os resultados foram muito desbalanceados, com uma alta probabilidade de falso positivo e baixíssima de falso

Tabela II
FALSOS POSITIVOS E FALSOS NEGATIVOS DE CADA ABORDAGEM.

Abordagem	Falsos positivos	Falsos negativos
Haar + AdaBoost	42,59%	**0,38%**
HOG + SVM	5,04%	5,65%
CNN	**0,83%**	1,06%

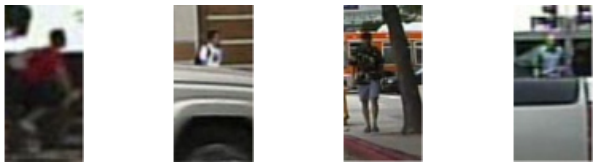
Tabela III
PRECISÃO E ACURÁCIA DE CADA ABORDAGEM.

Abordagem	Precisão	Acurácia
Haar + AdaBoost	68,78%	**96,64%**
HOG + SVM	94,63%	91,53%
CNN	**99,12%**	95,98%

negativo. Para o caso da técnica 2, os resultados foram melhores e mais balanceados, com 5% para as duas estatísticas. Para a técnica 3 os resultados foram os melhores, com valores próximos a 1%.

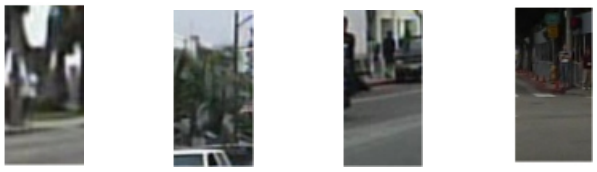
Avaliando os valores de precisão e acurácia como mostrado na Tabela III, a abordagem com Haar apresentou a pior precisão e a melhor acurácia, enquanto a abordagem com HOG apresentou boa precisão e um bom resultado na acurácia. A CNN, por sua vez, apresentou a melhor precisão e uma boa acurácia.

Para ilustrar as estatísticas encontradas a Figura 3 e Figura 4 mostram exemplos classificados errados pelas técnicas.



(a) Haar (b) Haar e HOG (c) HOG e CNN (d) HOG e CNN

Figura 3. Exemplos dados como falso negativo pelas técnicas listadas abaixo de cada imagem



(a) HOG e CNN (b) Haar e HOG (c) Haar (d) Haar

Figura 4. Exemplos dados como falso positivo pelas técnicas listadas abaixo de cada imagem

Com a análise de todas as estatísticas é possível perceber que a CNN, quando não apresenta os melhores resultados, está bem próxima do melhor, ratificando o fato desta técnica ser considerada o estado da arte. A segunda melhor técnica nesta análise foi a abordagem com HOG e SVM, obtendo resultados bem próximos a CNN. Esse fato pode ser explicado por ser uma técnica proposta desde sua original para o problema de detecção de pedestres. A abordagem com as *Features* de Haar com o AdaBoost apresentou grandes problemas para a classificação de imagens negativas, ocasionando um alto número de falsos positivos e resultando em um desempenho ruim. A explicação para isto por ser dado pelo fato das *Features* de Haar terem sido desenvolvidas originalmente para a detecção de faces.

VI. CONCLUSÃO

Atualmente inúmeras abordagens utilizando Redes Neurais com aprendizado profundo estão alcançando os melhores resultados em diversos problemas de Visão Computacional.

Para este trabalho, foram comparadas duas técnicas tradicionais de Visão Computacional com uma técnica de Rede Neural Convolucional que apresenta os resultados do estado da arte. Comparamos a utilização de *Features* de Haar com o classificador AdaBoost, Histograma de Gradiente Orientados com o classificador de Máquina de Vetores de Suporte e um ajuste da Rede Neural Convolucional AlexNet para o problema abordado nesse trabalho. Nos testes realizados, a abordagem com a CNN obteve os melhores resultados na maioria dos casos, comprovando o fato de ser considerada a técnica em estado da arte.

REFERÊNCIAS

[1] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. Lecun, "Pedestrian detection with unsupervised multi-stage feature learning," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[2] M. D. Zeiler and R. Fergus, *Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*. Springer International Publishing, 2014, ch. Visualizing and Understanding Convolutional Networks, pp. 818–833.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, 2015.

[4] W. Ouyang and X. Wang, "Joint deep learning for pedestrian detection," in *The IEEE International Conference on Computer Vision (ICCV)*, 2013.

[5] J. H. Hosang, M. Omran, R. Benenson, and B. Schiele, "Taking a deeper look at pedestrians," *CoRR*, vol. abs/1501.05790, 2015.

[6] R. Benenson, M. Omran, J. Hosang, and B. Schiele, "Ten years of pedestrian detection, what have we learned?" in *Computer Vision for Road Scene Understanding and Autonomous Driving (CVRSUAD, ECCV workshop)*, 2014.

[7] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278–2324.

[8] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR 2015*, 2015.

[9] P. Dollár, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *PAMI*, vol. 34, 2012.

[10] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *In CVPR*, 2005, pp. 886–893.

[11] "Página oficial do projeto opencv," http://opencv.org, 2016.

[12] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," 2001, pp. 511–518.

[13] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[14] "Página oficial do projeto digits da nvidia," https://developer.nvidia.com/digits, 2016.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.