Reconhecimento de Placas de Trânsito Usando Redes Pré Treinadas NIN-ImageNet Deep-Learning

Rânik Guidolini, Vinicius Cardoso

Resumo — Neste trabalho é apresentado uma abordagem de reconhecimento de placa usando a rede deep-learning pré-treinada NIN-ImageNet. Utilizamos a estratégia de fine-tuning para utilizar o aprendizado da rede treinada para outro dominio para aplicar ao problema de reconhecimento de placas de trânsito. Para o treino e teste da rede, foi escolhida a base de dados German Traffic Sign Recognition Benchmark (GTSRB), contendo 43 tipos de placas de trânsito da Alemanha. Nossos experimentos mostraram que a rede foi capaz de aprender os padrões das placas de trânsito com uma taxa de acerto de 98,2%, ficando em quarto lugar no ranking da base de dados utilizada, sendo que o segundo colocado foi o ser humano.

I. Introdução

O problema de reconhecimento de placas de trânsito tem se popularizado com a evolução do desenvolvimento de veículos autônomos e incorporação de sistemas de assistência ao motorista em veículos normais. As placas de trânsito são destacadas para serem facilmente notadas pelos motoristas, com alto contraste e cores brilhantes. O problema de reconhecimento consiste em dada uma placa de trânsito, dizer qual o tipo da placa, e apresenta uma série de desafios como variação do ponto de vista, variações na iluminação, desgaste natural da placa, pichações e danos físicos. O problema de detectar a existência ou não de uma placa de trânsito em uma dada imagem é uma outra linha de pesquisa e não será tratado neste trabalho. A base GTSRB possui mais de 50 mil imagens de placas de trânsito da Alemanha dividas em 43 classes diferentes, algumas delas são mostradas na Figura 1. Esta base apesar de disponibilizada em formato de vídeo não fornece informação temporal do conjunto de treino, ou seja, uma sequência temporal de imagens da mesma placa, o que impossibilita a acumulação da informação.

Figura 1- Exemplos de imagens da base GTSRB.

II. Trabalhos relacionados

O problema de reconhecimento de placas de transito foi abordado de diversas formas utilizando várias técnicas computacionais que foram comparadas com a capacidade do ser humano na competição IJCNN 2011. A única técnica que foi capaz de alcançar resultados melhores que o ser humano, e que ficou em primeiro lugar na competição com 99,46% de acerto, utilizou uma implementação em GPU completamente parametrizável Multi-Column Deep Neural Network (MCDNN) [1].

Em segundo lugar na competição ficou o desempenho do ser humano com 98,84% de acerto [2]. A técnica utilizada para avaliar a capacidade do ser humano em reconhecer placas de trânsito, seguia o método de apresentar um conjunto com 400 imagens selecionadas randomicamente para cada um dos 32 participantes na fase de treino, depois se apresentava uma única placa de cada classe para cada um dos participantes para o teste.

O terceiro lugar ficou com 98,31% de acerto, e usou Multi-Scale Convolutional Networks [3] que são inspiradas biológicamente.

Abordagens usando redes neurais sem peso que são biologicamente inspiradas foram propostas em [4] e [5]. Estas abordagens conseguiram alcançar um desempenho máximo de acerto de 98,73% e não usam redes profundas.

Nossa abordagem utiliza a técnica de fine-tuning usando uma Network in Network ImageNet, e será descrita na próxima seção.

III. Metodologia

Para a tarefa de reconhecimento de placas de trânsito, utilizamos a rede pré treinada NIN-ImageNet que foi treinada com a base de imagens ImageNet [6].

A rede AlexNet ImageNet usa uma arquitetura deep convolutional neural network (CNN) que foi treinada para classificar 1.2 milhões de imagens de alta resolução em 1000 classes diferentes na Large Scale Visual Recoginition Challenge 2010 LSVRC-2010 [7], cada uma das imagens é a representação de um objeto e foi classificada por um humano.

Network in Network (NIN) é uma estrutura criada para simplificar as redes profundas de forma a diminuir a memória ocupada [8]. As redes mais comuns utilizam a camada convolucional convencional com filtros lineares seguidos por uma função de ativação não-linear para fazer a varredura de entrada. Em vez disso, a técnica NIN utiliza micro redes neurais com estruturas mais complexas para abstrair os dados.

A rede NIN-ImageNet usada possui 4 camadas NIN treinadas com a base de dados ImageNet e pode ser baixada pelo seguinte link: https://gist.github.com/mavenlin/d802a5849de39225bcc6.

Graças à substituição da camada completamente conectada por uma camada de agrupamento médio global, este modelo tem parâmetros muito reduzidos, o que resulta em um snapshot de tamanho 29MB, em comparação com 230MB da AlexNet, ou seja, um oitavo do tamanho.

O modelo da rede NIN-ImageNet foi construido usando o Deep learning open source framework Caffe [9] desenvolvido pelo Berkeley Vision and Learning Center (BVLC). Com esse framework, é possível definir uma rede (estrutura, camadas, etc) utilizando um arquivo de configuração, e facilmente treinar a rede na CPU ou na GPU apenas mudando uma flag. Entre as facilidades da Caffe, esta a possíbilidade de utilizar um modelo da rede com seus arquivos de configuração e uma cópia do estado (pesos da rede) da rede (snapshot) que possibilita retomar o treinamento do momento em que foi capturado. Mais detalhes podem ser encontrados no site: http://caffe.berkeleyvision.org/.

Nosso código esta disponível no seguinte link: https://github.com/vinibc/Trab_Cognicao_RanikVinicius.git

IV. Experimentos

Para avaliar a abordagem de reconhecimento de placas de trânsito usando redes pré-treinadas, usamos o modelo NIN-Imagenet em todos os experimentos e para o fine-tuning o dataset da GTSRB, com 43 tipos de placas de trânsito, contendo 39.209 imagens separadas para a fase de treino, e outras 12.630 imagens para fase de teste. As imagens do dataset variam em tamanho de 15x15 até 250x250 com diferentes posições e iluminação, e em ambientes reais (Figura 1).

O experimento foi dividido em duas abordagens de fine-tuning: (i) treinar todas as camadas do modelo, (ii) treinar apenas a última camada.

A. *Preparação dos dados*

Para ambas abordagens, o método da entrada dos dados na rede foi a mesma. O GTSRB disponibiliza o conjunto de treino com as imagens em formato ppm sendo 43 tipos de placas classificadas de 0 a 42, e separadas em pastas de cada classe. Para adaptar à entrada da rede, selecionamos o caminho, nome da imagem e a classe e misturamos a ordem para que a rede treinasse com imagens de classes variadas usando um script. Todas as imagens de treino ficaram então organizadas em um arquivo de texto, que é passado para a camada de entrada de dados que lê esse arquivo para acessar as imagens. O conjunto de teste já é disponibilizado pela GTSRB com um arquivo contendo nome, tamanho e classe das imagens, deste arquivo foi extraido o nome e classe das imagens para a fase de teste e passado o arquivo texto para a camada de dados de teste da rede. Tanto as imagens de teste quanto as imagens de treino foram redimensionadas para o tamanho de 224 x 224 usando um paramêtro na configuração da camada de entrada de dados, o framework Caffe já oferece essa funcionalidade.

B. *Configuração e treinamento da rede*

Para adaptar a configuração da rede para o reconhecimento de placas, primeiramente foi adicionada a camada de entrada de dados para imagens (type: IMAGE_DATA) que requer um arquivo de texto como fonte para as imagens como explicado na sessão anterior. Esse tipo de camada também permite a definição de tamanho da imagem, assim redimencionando todas para um tamanho escolhido (224x224 neste caso), tanto para o treino quanto para o teste.

Outra mudança foi a alteração da última camada para uma totalmente conectada (InnerProduct) com 43 saídas (0 a 42 classes) e a camada de foi mantida para visualização dos resultados e a softmax-loss para o treino.

Na primeira abordagem, treinando todas as camadas, não houve alteração na taxa do multiplicador de aprendizado das camadas. Na segunda abordagem, um valor infimo foi aplicado nas outras camadas, apenas para evitar que o multiplicador fosse zero, assim apenas a última manteve a taxa de aprendizado.

No treinamento é necessário definir a taxa e a política de decaimento do aprendizado da rede. A política escolhida foi a step, nesta política, a cada determinada quantidade de iterações (step_size), a taxa de aprendizado (base_lr) é reduzida por um fator gama, a Figura 2 mostra o comportamento da politica step.

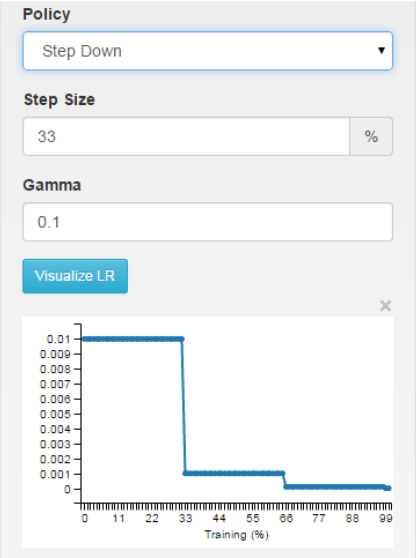

Figura 2- Exemplo de decaimento usando a politica step.

Além disso, um fator para suavizar a atualização do peso (momentum) também é definido.

O treinamento e teste foram feitos usando o modo de processamento em GPU, em uma placa Nvidia GeForce GTX 660 com 1152 núcleos cuda e 1,5 GB de memória.

V. Resultados e Discussões

Para fine-tuning treinando todas as camadas foram feitos dois testes variando os valores de aprendizado: o primeiro com taxa de aprendizado da rede de 0.0001, a politica de decaimento em step, com step_size: 200.000, gama = 0.1, momentum: 0.9 e weight_decay: 0.0005. Com esses valores, já em 10.000 iterações a rede alcançou precisão de 96%, o treino foi finalizado com 112.000 iterações, com testes a cada 20 iterações, chegando a um acerto de 98,2%. Por não ter completado um step (200.000 iterações), neste caso no teste a taxa de aprendizado foi fixa.

O segundo teste, foi aplicado uma taxa de aprendizado de: 0,01, a politica de decaimento step, gama: 0,001, momento: 0,9 e weight_decay: 0,0005. Nesta configuração o treino foi finalizado pela baixa evolução do treino e teste, mantendo um acerto de 0,71% após 12 horas de treino e 11000 iterações. Assim os valores finais escolhidos foram o do primeiro teste.

Já na segunda abordagem de fine-tuning, treinando apenas a última camada, dois testes foram realizados, no primeiro os parâmetros de taxa de aprendizado da rede foi 0.0001, a politica de decaimento em step, gamma = 0.1, momentum: 0.9 e weight_decay: 0.0005. Neste teste após 18.000 iterações a rede teve um acerto de no máximo 89% e a precisão do treino ainda não tinha se estabilizado chegando a no máximo 90%, por este motivo o teste foi parado. Dado esse resultado, no segundo teste, o step_size foi reduzido para 10.000 iterações, reduzindo também a taxa de aprendizado para 0.01 e mantendo os outros parâmetros do ultimo teste. Neste caso após 32000 iterações, a precisão foi de 74% porém, não houve estabilização do treino variando entre 80% e 90%.

Os resultados dos experimentos foram satisfatórios e mostraram que é possível usar uma rede treinada para um determinado domínio, e aplica-la para outro domínio, com bons resultados. Nestes experimentos, para o reconhecimento de placas de trânsito, também é possível notar que fazer o fine-tuning treinando todas as camadas foi melhor que apenas treinar a última.

Utilizar modelos de redes neurais convolucionais já treinadas, permite o desenvolvimento de soluções eficientes, e contorna o problema da necessidade de muitos dados, para treinar esse tipo de rede neural.

Comparando o resultado obtido, com os resultados do benchmark do GTSRB em 08 de julho de 2016, alcançamos o quarto lugar no ranking da base de dados utilizada, sendo que o segundo colocado foi o ser humano.

Bibliografia

[1] CireşAn, D., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. Neural Networks, 32, 333-338.

[2] Stallkamp, J., Schlipsing, M., Salmen, J., & Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. Neural networks, 32, 323-332. H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.

[3] Sermanet, P., & LeCun, Y. (2011, July). Traffic sign recognition with multi-scale convolutional networks. In Neural Networks (IJCNN), The 2011 International Joint Conference on (pp. 2809-2813). IEEE.

[4] Berger, Mariella, Avelino Forechi, Alberto F. De Souza, Jorcy de Oliveira Neto, Lucas Veronese, and Claudine Badue. "Traffic sign recognition with VG-RAM weightless neural networks." In 2012 12th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 315-319. IEEE, 2012.

[5] Berger, Mariella, Avelino Forechi, Alberto F. De Souza, J. De Oliveira Neto, Lucas Veronese, Victor Neves, Edilson de Aguiar, and Claudine Badue. "Traffic sign recognition with WISARD and VG-RAM Weightless Neural Networks." Journal of Network and Innovative Computing 1, no. 2013 (2013): 87-98.

[6] Deng, J., Dong, W., Socher, R., Li, L.-J., Li K., & Fei-Fei, L., ImageNet: A Large-Scale Hierarchical Image Database. IEEE Computer Vision and Pattern Recognition (CVPR), 2009.

[7] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[8] Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.

[9] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., and Darrell, T., Caffe: Convolutional Architecture for Fast Feature Embedding, arXiv preprint arXiv:1408.5093, 2014.