Atividade Orientada de Ensino: detecção de pontos-chave em imagens de carros

Edson Takashi Matsubara[1], **Lucas Santana Escobar**[1]

[1]Faculdade de Computação – Universidade Federal de Mato Grosso do Sul (UFMS)
Cidade Universitária, Av. Costa e Silva - Pioneiros, MS, 79070-900 – Campo Grande – MS – Brasil

{edson.matsubara, santana.escobar}@ufms.br

Resumo. *Este relatório descreve o trabalho de pesquisa realizado acerca de detecção de pontos-chave (keypoints) em para-choques de carros, realizado com a intenção de construir um estágio inicial para um sistema maior de predição de velocidade média. Para tal, foi treinado um algoritmo baseado na implementação em PyTorch*[1] *do algoritmo apresentado em [Sun et al. 2019], utilizando um conjunto de dados coletados de imagens aéreas e anotados manualmente via Labelbox*[2]*. Observou-se no fim um normalized mean error (NME) próximo de zero, no entanto, por conta da impossibilidade de aplicação da solução em cenários onde o veículo está rotacionado, outra abordagem se faz necessária.*

1. Introdução

Quando se tratando de soluções de estado-da-arte para tarefas de Visão Computacional, Redes Neurais Convolucionais (*CNNs*) são, via-de-regra, a técnica mais utilizada. Isso se dá pela alta capacidade no reconhecimento de padrões apresentada por essas arquiteturas, comumente dirigidas por uma abordagem que reduz a representação dos componentes (*features*) da imagem analisada para, por fim, performar a classificação dos elementos apresentados. Para algumas tarefas, no entanto, obtém-se melhores resultados quando utilizadas representações de alta resolução, as quais são brevemente discutidas neste relatório.

Há, comumente, duas maneiras de computar representações de alta resolução: recuperar estas representações de saídas produzidas por uma rede, utilizando representações de baixa resolução, ou manter as representações de alta resolução por meio de convoluções de alta resolução. Neste trabalho, optou-se pela utilização de um modelo que segue a segunda abordagem, a HRNet, que mantém as representações conectando convoluções de alta e de baixa resolução em paralelo e realizando a fusão de *features* em diferentes escalas repetidamente. Um exemplo dessa arquitetura é mostrado na Figura 1.

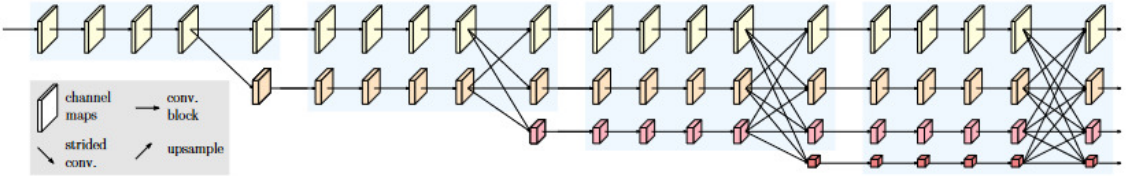

Figura 1. Exemplo de arquitetura de uma HRNet, como demonstrado em [Sun et al. 2019].

[1]https://pytorch.org/
[2]https://labelbox.com/

A detecção de pontos de interesse (*keypoints*) em imagens, tarefa discutida neste trabalho, consiste na localização de conjuntos de coordenadas únicas da imagem a partir de uma série de pontos selecionados previamente. Essas técnicas são utilizadas, por exemplo, em sistemas de reconhecimento facial [Mian et al. 2008] e estimativa de pose [Kocabas et al. 2018], este relatório, no entanto, foca na adaptação de uma solução de detecção de *keypoints* em faces para detecção de pontos em para-choques de carros, visando a criação de um sistema capaz de identificar a peça em imagens variadas.

2. Conjunto de Dados

Para o treinamento do algoritmo escolhido, foi criado um novo conjunto de dados chamado *Bumper Keypoints (BK) Dataset*. Os exemplos que compõem o conjunto de dados são provenientes de:

- imagens aéreas coletadas utilizando um drone *DJI Phantom 4*, por meio do qual gravou-se 3 vídeos resolução 4K/30fps, em 3 alturas diferentes: 5 metros, 10 metros e 15 metros.
- imagens da Av. Costa e Silva coletadas via celular, em qualidade HD/30fps, de cima do pontilhão de acesso ao Estádio Universitário Pedro Pedrossian (Morenão)
- um subconjunto das imagens da base de dados *Stanford Cars Dataset*[Krause et al. 2013], com exemplos nos quais os para-choques dos carros estão visíveis.

Com os quadros dos vídeos e todas as imagens em mãos, iniciou-se o processo de anotação dos dados utilizando a ferramenta Labelbox, no qual foram definidas as *bouding-boxes* dos carros e quatro pontos do para-choque dos mesmos. Tanto na parte traseira, quanto na parte dianteira dos carros presentes nas imagens.

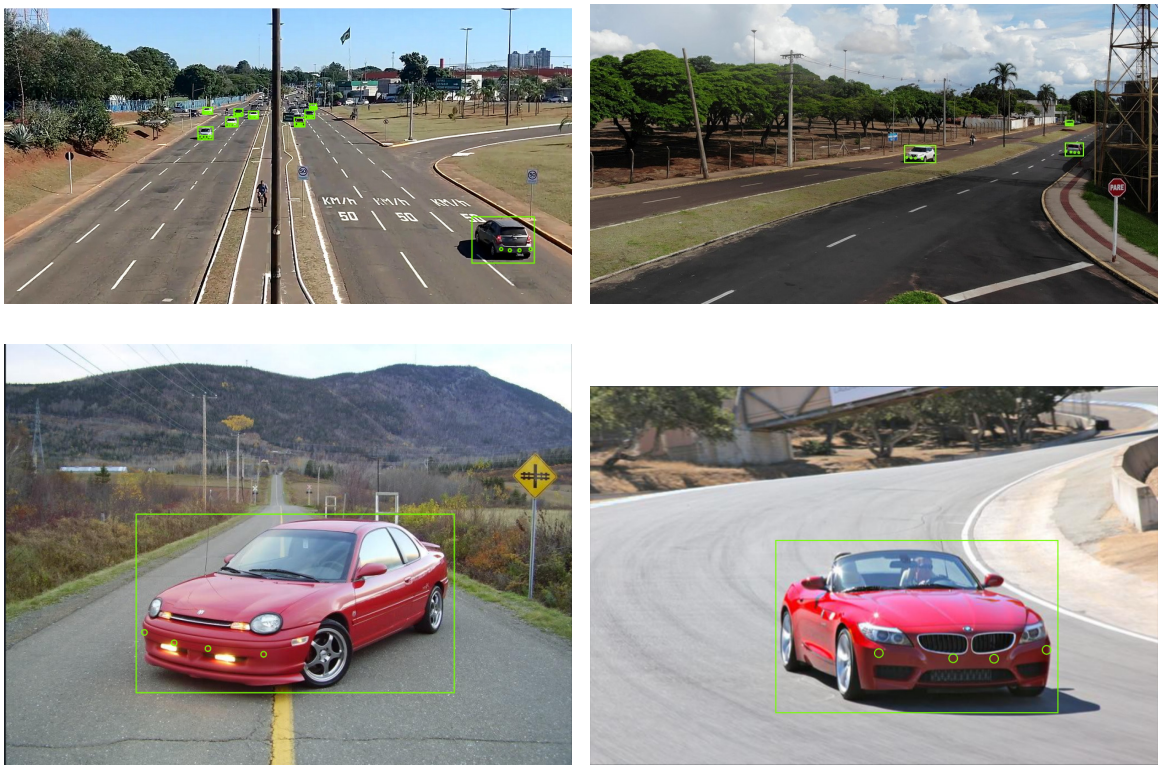

Figura 2. Exemplos anotados do conjunto de treinamento.

No total, foram anotados 4529 exemplos distintos retirados dos conjuntos acima. Destes exemplos, 3168 (aproximadamente 70%) foram utilizados para treinamento e os outros 30% foram divididos em 680 exemplos de teste e 679 de validação.

3. Treinamento

O treinamento se deu em duas etapas principais: preparação dos exemplos do conjunto de dados e o processo de treinamento da rede. O código utilizado como base está disponível no GitHub[3], e possui suporte para alguns *datasets*, como AFLW [Köstinger et al. 2011] e *300 Faces-In-The-Wild*[Sagonas et al. 2013]. Para a primeira etapa, foi adicionada uma classe chamada BK que estende a classe *torch.utils.data.Dataset*[4] da biblioteca PyTorch e uma arquivo *yml* com a configuração de treinamento necessária para dar suporte ao novo conjunto de dados.

A escolha do modelo utilizado se deu com base nos experimentos apresentados em [Sun et al. 2019] sobre detecção de pontos faciais de referência, uma vez que este é o problema decidido como mais próximo à tarefa discutida nesse relatório. A arquitetura em questão, chamada de HRNetV2-W18, é uma modificação da arquitetura apresentada na Seção 1, na qual as representações de baixa resolução passam por um processo de aumento de resolução e concatenação dos resultados, gerando a representação que é, por fim, utilizada para a estimativa dos pontos.

[3] https://github.com/HRNet/HRNet-Facial-Landmark-Detection

[4] https://pytorch.org/tutorials/beginner/basics/data_tutorial.html#creating-a-custom-dataset-for-your-files

Feito isto, deu-se início ao treinamento da rede. O modelo foi treinado por 80 épocas, utilizando *batches* de tamanho 16 e apenas uma GPU Nvidia M40, com 12GB de memória de vídeo. Os carros foram destacados das imagens originais seguindo as *bouding boxes* e redimensionados para 256×256. Antes de serem fornecidos como entrada para a rede, os mesmos também passam por um processo de *agumentation* que rotaciona os exemplos $\pm 30°$, aplica uma escala usando um fator que varia de 0.25 e vira a imagem randomicamente.

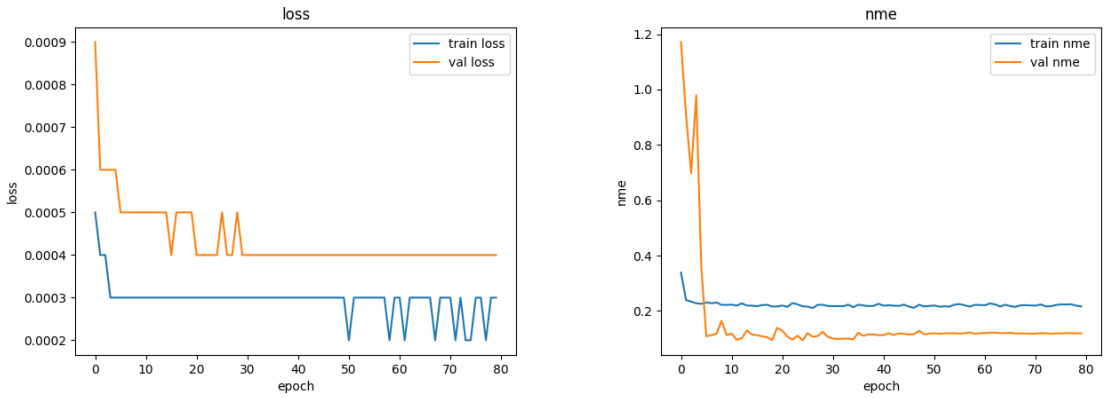

Figura 3. Evolução do erro e do *NME* durante o treinamento.

Por fim, foi utilizado um *learning rate* inicial de 0.0001, diminuído sistematicamente para 0.00001 e 0.000001, respectivamente depois de 30 e 50 épocas. A métrica utilizada para avaliação do desempenho do modelo foi o *Normalized Mean Error (NME)* e *optimizer* Adam. Na Figura 3 é demonstrada a evolução destes valores ao longo do treinamento.

Figura 4. Resultados das pontos estimados em exemplos de teste.

Para teste, foram utilizados *batches* de tamanho 8 e o valor final do *NME* atingido foi de 0.0910. É possível notar que, apesar da posição dos pontos preditos estar próxima do ideal, em exemplos onde o carro aparece levemente rotacionado, o modelo não é capaz de reproduzir a rotação.

Referências

Kocabas, M., Karagoz, S., and Akbas, E. (2018). Multiposenet: Fast multi-person pose estimation using pose residual network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 417–433.

Krause, J., Stark, M., Deng, J., and Fei-Fei, L. (2013). 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia.

Köstinger, M., Wohlhart, P., Roth, P. M., and Bischof, H. (2011). Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 2144–2151.

Mian, A. S., Bennamoun, M., and Owens, R. (2008). Keypoint detection and local feature matching for textured 3d face recognition. *International Journal of Computer Vision*, 79:1–12.

Sagonas, C., Tzimiropoulos, G., Zafeiriou, S., and Pantic, M. (2013). 300 faces in-the-wild challenge: The first facial landmark localization challenge. In *2013 IEEE International Conference on Computer Vision Workshops*, pages 397–403.

Sun, K., Zhao, Y., Jiang, B., Cheng, T., Xiao, B., Liu, D., Mu, Y., Wang, X., Liu, W., and Wang, J. (2019). High-resolution representations for labeling pixels and regions.