Segmentação de Estradas em Mapas de Remissão utilizando Redes Neurais

Ludmila Dias[1], Eduardo O. Ferreira[1], Francisco de Assis Boldt[1], Marcos V. Faria[1], Thiago M. Paixão[1]

[1]Instituto Federal de Educação, Ciência e Tecnologia do Espírito Santo (IFES) - Serra, ES, Brasil

{ludmila.dias, marcos.faria}@estudante.ifes.edu.br

{thiago.paixao,franciscoa}@ifes.edu.br

{duduoliveirae11}@gmail.com

Abstract. *Advances in perception for autonomous driving have been driven by deep learning, which uses various sensors such as cameras, LiDARs, and radars to obtain a precise understanding of the environment. In this context, our work focuses on the semantic segmentation of roads in remission maps generated by LiDAR. We compare four convolutional neural network architectures—U-Net, PSPNet, FPN, and LinkNet—with the ENet model. Our goal is to evaluate the performance of these models in terms of average accuracy compared to ENet and to analyze them using other standard metrics.*

Resumo. *Avanços na percepção para direção autônoma têm sido impulsionados pelo aprendizado profundo, que utiliza diversos sensores, como câmeras, LiDARs e radares, para obter uma compreensão precisa do ambiente. Neste contexto, nosso trabalho foca na segmentação semântica de estradas em mapas de remissão gerados por LiDAR. Comparando quatro arquiteturas de redes neurais convolucionais—U-Net, PSPNet, FPN e LinkNet—com o modelo ENet, nosso objetivo é avaliar o desempenho desses modelos em termos de acurácia média em relação ao ENet.*

1. Introdução

Os acidentes de trânsito são a oitava principal causa de morte no mundo, resultando em cerca de 1,35 milhão de mortes anuais [Organização Pan-Americana da Saúde 2021]. Além disso, congestionamentos causam perda de tempo e produtividade, impactando a economia e qualidade de vida [Schrank et al. 2019]. Os Veículos Autônomos (VAs) são uma solução promissora, pois podem reduzir acidentes e otimizar o tráfego, melhorando a segurança e eficiência urbana [Martínez-Díaz and Soriguera 2018, Fagnant and Kockelman 2015].

Os sistemas de autonomia dos VAs dividem-se em sistemas de percepção e de tomada de decisão [Paden et al. 2016]. O sistema de percepção localiza o veículo, mapeia obstáculos e reconhece sinais de trânsito, enquanto o sistema de tomada de decisão planeja rotas e controla o veículo [Badue et al. 2021]. Equipados com sensores, câmeras, inteligência artificial e sistemas de comunicação, os VAs operam sem intervenção humana, necessitando de mapas detalhados para a tomada de decisão [Wong et al. 2021].

Uma abordagem para mapeamento é o uso de Mapas de Remissão, criados a partir de dados de sensores LiDAR [Carneiro et al. 2018] que capturam a intensidade da luz refletida por superfícies, gerando uma representação tridimensional do entorno do veículo. O processamento adicional desses dados requer o uso de redes neurais profundas para identificar com precisão faixas de estrada e outras características importantes [Carneiro et al. 2018].

De acordo com [Bolimera et al. 2023], existem várias abordagens eficazes para a detecção de faixas de rodagem utilizando redes neurais profundas. Uma dessas abordagens é o uso de redes neurais convolucionais profundas (CNNs) para segmentação semântica, que são adequadas para derivar informações contextuais e gerar propostas precisas de máscaras de faixas em cada pixel da imagem [Bolimera et al. 2023]. Essas redes são capazes de aprender a partir de grandes volumes de dados rotulados, permitindo uma melhor generalização e precisão em diversas condições ambientais.

A segmentação semântica é uma subárea da visão computacional, atribuindo rótulos de categoria a cada pixel de uma imagem [Singh and Rani 2020]. A capacidade de fornecer informações detalhadas em nível de pixel auxilia sistemas inteligentes a entenderem melhor o espaço e a tomarem decisões cruciais [Hao et al. 2020]. Isso diferencia a segmentação semântica de outras tarefas, como a classificação de objetos, que rotula a imagem inteira, ou a detecção de objetos, que identifica a localização dos objetos na cena [Hao et al. 2020].

Este trabalho propõe comparar diferentes arquiteturas de redes neurais convolucionais para segmentação de estradas em mapas de remissão. O objetivo é replicar e estender o experimento do artigo "Mapping Road Lanes Using Laser Remission and Deep Neural Networks" [Carneiro et al. 2018], utilizando quatro arquiteturas distintas: U-Net [Ronneberger et al. 2015], PSPNet [Zhao et al. 2016], FPN [Lin et al. 2016] e LinkNet [Chaurasia and Culurciello 2017], e compará-las com o modelo utilizado no artigo de referência, a rede E-Net [Paszke et al. 2016] utilizando o cálculo da acurácia média.

2. Referêncial Teórico

2.1. Redes Convolucionais de Segmentação Semântica

Nos últimos anos, houve um grande progresso na segmentação semântica com o uso de redes neurais convolucionais profundas (Deep Convolutional Neural Networks, DCNNs). Autoencoders são redes neurais profundas utilizadas para segmentação, compostas por dois módulos principais: codificação e decodificação. O módulo de codificação reduz a imagem de entrada a uma representação compacta, enquanto o módulo de decodificação reconstrói a imagem original a partir dessa representação compacta. Durante o processo, a imagem é reduzida a um tamanho intermediário adequado e depois ampliada novamente para seu tamanho original. Em autoencoders, as camadas de entrada e saída possuem o mesmo tamanho, sendo que a redução extrai características relevantes da imagem para a codificação, e a ampliação reconsolida essas características para recuperar a imagem original [Singh and Rani 2020].

2.2. Trabalhos Relacionados

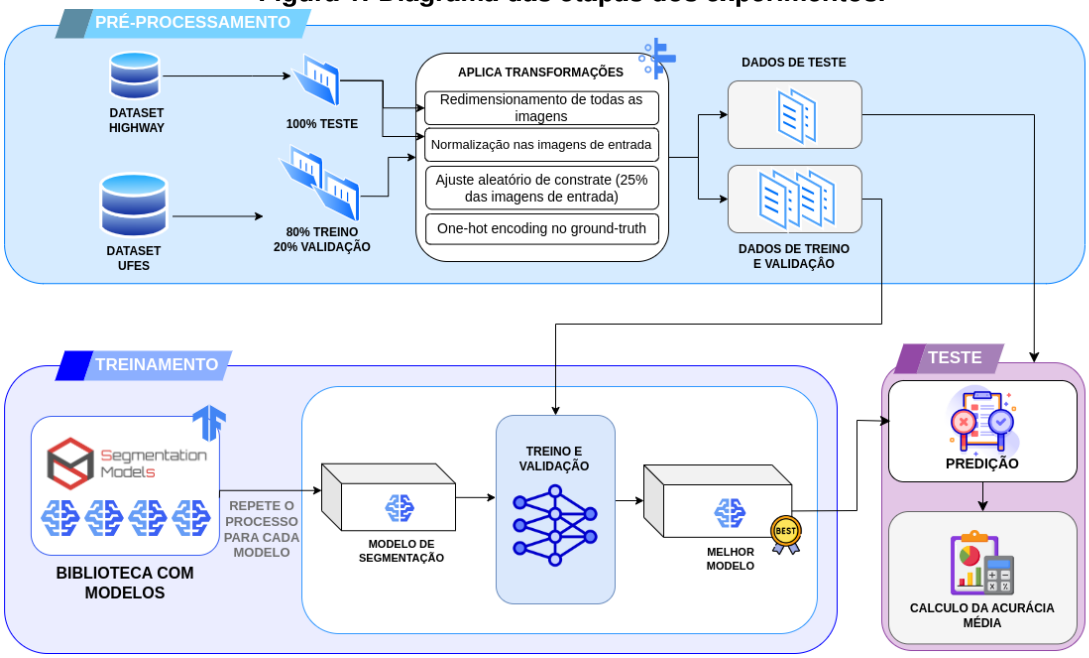
O artigo "Mapping Road Lanes Using Laser Remission and Deep Neural Networks" [Carneiro et al. 2018] aborda sobre todo o processo desde a preparação do dataset ao

uso dos Road Maps (Mapas de Estrada) preditos pelo modelo treinado utilizando um veículo autônomo. O modelo treinado descrito no artigo é a ENet [Paszke et al. 2016], que demonstrou ser eficaz para a tarefa proposta. Entretanto, existem outros modelos que superaram o estado da arte da arquitetura ENet, como por exemplo a LinkNet [Chaurasia and Culurciello 2017]. Sendo assim, podem obter resultados melhores com relação as métricas avaliadas de segmentação semântica nesse contexto.

3. Materiais e Métodos

Nesta seção é descrita a metodologia dos experimentos para o treinamento e teste dos modelos. A Figura 1 apresenta um fluxograma detalhando o passo-a-passo dessa metodologia. São três etapas macro: pré-processamento, treinamento e teste. A etapa de **pré-processamento** consiste na definição das bases de dados para o experimento e das aplicações de transformação nas imagens que o modelo receberá, ela é descrita na seção 3.2. O **treinamento** é composto pela definição dos modelos e o aprendizado desses, o processo é descrito na seção 3.3. Por fim, na etapa de **teste**, descrita na seção 4, os melhores modelos gerados pela etapa de treinamento são avaliados utilizando a acurácia média.

Figura 1. Diagrama das etapas dos experimentos.



3.1. Recursos de Software e Hardware

A linguagem Python foi utilizada para o desenvolvimento de todos os códigos necessários neste estudo. Adicionalmente, a biblioteca Segmentation Models [1], que é baseada no Keras, foi empregada para importar os modelos utilizados. O Keras, sendo uma API de alto nível do TensorFlow [2], foi essencial para a criação, treinamento, validação e teste dos modelos de aprendizado profundo. A biblioteca TensorFlow também desempenhou um

[1] https://github.com/qubvel/segmentation_models
[2] https://www.tensorflow.org/?hl=pt-br

papel fundamental, proporcionando funcionalidades para a configuração do uso da GPU nos experimentos. Outras bibliotecas importantes incluíram a Numpy [3], que foi fundamental para a manipulação de dados numéricos, a Pandas [4], que facilitou a importação dos dataframes contendo os caminhos para os dados, e a OpenCV [5], que foi utilizada para o processamento de imagens. Os experimentos foram realizados em um computador equipado com uma placa de vídeo NVIDIA GeForce RTX 2080 com 8GB de memória, proporcionando o desempenho necessário para a execução das tarefas de treinamento e validação dos modelos.

3.2. Pré-processamento

Nesta etapa, os dados são importados e subdivididos em subconjuntos. Posteriormente, são estabelecidas as transformações a serem aplicadas às imagens em cada fase do experimento. Os detalhes específicos de cada etapa são descritos a seguir.

3.2.1. Base de dados

O conjunto de imagens[6] utilizado foi o mesmo produzido e usado no artigo de referência. Todas as imagens possuem tamanho padrão de 120x120 e estão em escala de cinza. Como mostra a Figura 2, o conjunto de dados possui imagens de entrada (a) para as redes de segmentação e máscaras (b) que se referem a saída de segmentação desejada para o modelo. As imagens de entrada são mapas de remissão obtidos a partir do carro autônomo IARA da Universidade Federal do Espirito Santo (UFES). Parte do dataset refere-se à estrada de contorno da UFES, tem uma extensão de 3,7 km e é nomeado como dataset **UFES**, com um total de 110.544 imagens que são utilizadas no treino e validação. A parte restante do dataset refere-se a rodovias (distantes da UFES), com uma extensão de 32,4 km, e é nomeado como dataset **HIGHWAY** e possui 3556 imagens de teste. Os dados são do percurso de asfalto de Vila Velha - ES a Guarapari - ES.

A partir dos mapas de remissão foram produzidas as máscaras que são imagens que representam a segmentação desejada de saída que o modelo deve prever. Essas imagens foram produzidas manualmente utilizando o software de edição Inkscape e algoritmos de processamento digital de imagens [Carneiro et al. 2018]. As máscaras possuem valores que variam de 0 a 17, cada valor representando uma classe (situação): 0 = Fora da Faixa; 1 = Linha Sólida; 2 = Linha Pontilhada; 3 = Linha Sólida (50% de confiança); 4 = Linha Pontilhada (50% de confiança); 5, 6, 7, ..., 16 = distância até o centro da faixa (0, 1/22 da largura da faixa, 2/22, ..., 11/22 da largura da faixa, ou 1/2 da largura da faixa.

Seguindo os valores definidos no artigo de referência, o dataset UFES foi subdividido de forma aleatória em dois subconjuntos: 88.368 imagens para treino e 22.176 imagens para teste, utilizadas na validação. Este conjunto já passou por um processo de aumento de dados, onde as imagens foram rotacionadas e transladadas em diferentes valores. Por outro lado, o dataset HIGHWAY foi utilizado exclusivamente para testes, sem sofrer subdivisão ou qualquer processo de aumento de dados.

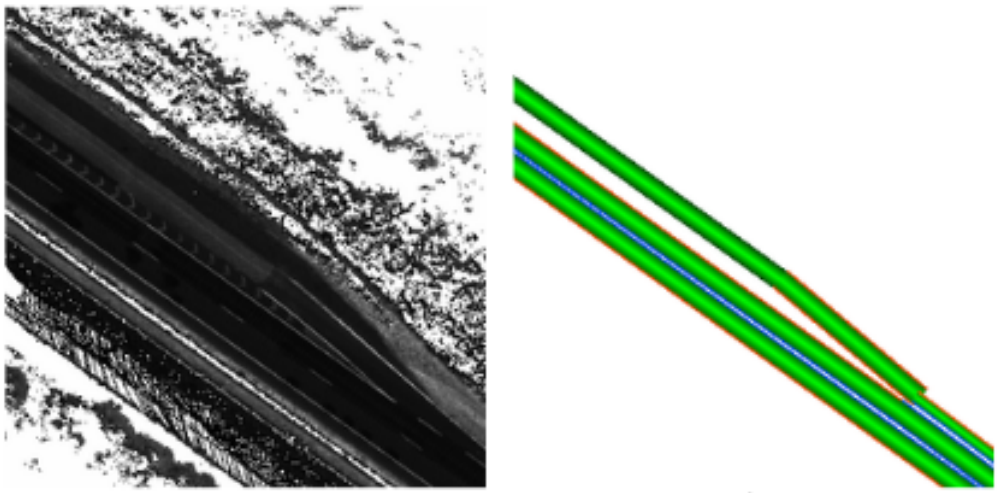
[3]https://numpy.org/

[4]https://pandas.pydata.org/

[5]https://opencv.org/get-started/

[6]https://github.com/LCAD-UFES/carmen_lcad/tree/master/src/road_mapper

Figura 2. Imagens de entrada (a) e máscaras do Dataset (b), conjunto UFES.



3.2.2. Transformações

Durante o pré-processamento são realizadas algumas transformações em cada conjunto de imagens de acordo com a necessidade:

- **Alteração de brilho e contraste:** Para ampliar a distinção entre as imagens, foram aplicadas transformações padrões de brilho e contraste em 25 por cento das imagens totais do dataset de treino. Este processo visa aumentar a variação visual no conjunto de dados.
- **Redimensionamento das imagens:** Esse é um processo necessário, uma vez que cada arquitetura de rede possui um padrão específico para o tamanho da entrada. Considerando que, na segmentação semântica, a informação de cada pixel é crucial, pois a classificação é realizada pixel a pixel, optou-se por aplicar *padding* — técnica que adiciona pixels ao redor de uma imagem — em vez de redimensionar as imagens com algoritmos padrão de bibliotecas, como o `numpy`, que podem causar deslocamentos indesejados de pixels. O *padding* foi utilizado para ajustar as imagens às dimensões exigidas, sendo esse procedimento aplicado a todos os conjuntos de imagens.
  O tamanho da imagem e o número de canais variam de acordo com a arquitetura e o formato de entrada exigido, entretanto, buscou-se manter o valor mais próximo possível do tamanho original das imagens (120x120). Sendo assim, o formato de entrada adotado foi 128x128 (3 canais) para U-NET e FPN, 128x128 (1 canal) para LinkNet e 144x144 (3 canais) para PSPNet.
- **Normalização:** A normalização utilizada divide o valor dos pixels por 255, resultando em uma escala de valores entre 0 e 1. Essa transformação é aplicada em todas as imagens de entrada. É utilizada a função *normalize* da biblioteca Keras.
- **Codificação:** A codificação de dados em redes neurais refere-se ao processo de transformar dados de entrada em um formato que possa ser facilmente utilizado pela rede neural para aprendizado e inferência. A codificação One-Hot foi aplicada apenas nas máscaras do treinamento e da validação. Sendo assim, as

máscaras passam a ter 17 canais, cada canal simbolizando uma classe. Para isso, é utilizada a função *to_categorical* da biblioteca Keras.

3.3. Treinamento

Na etapa de treinamento, os modelos de segmentação são definidos e submetidos ao processo de treino utilizando o conjunto de imagens da UFES. Nesta fase, são configurados os hiperparâmetros e os *callbacks*, que são essenciais para otimizar o desempenho do modelo. Os detalhes deste processo são descritos a seguir.

3.3.1. Modelos de Segmentação Semântica

Neste trabalho, foram investigados quatro modelos de segmentação semântica: U-NET [Ronneberger et al. 2015], Feature Pyramid Network(FPN) [Lin et al. 2016], PSPNet (Pyramid Scene Parsing Network) [Zhao et al. 2016] e LinkNet [Chaurasia and Culurciello 2017]. Os modelos (implementados a partir da biblioteca Segmentation Models) tem como base (backbone) a arquitetura Resnet34, mas cada um possui características diferenciadas na sua própria arquitetura.

U-Net É uma rede convolucional com uma arquitetura de encoder-decoder simétrica. O encoder consiste em camadas de convolução e pooling, enquanto o decoder utiliza convoluções transpostas para recuperar a resolução original da imagem. Conexões de salto entre as camadas correspondentes no encoder e decoder ajudam a preservar informações espaciais detalhadas.

FPN Esta rede utiliza uma arquitetura de pirâmide para combinar características de diferentes resoluções. Ele possui um caminho de cima para baixo que combina mapas de características de alta e baixa resolução, permitindo a segmentação de objetos em múltiplas escalas. As camadas de características são extraídas em diferentes níveis da ResNet e combinadas para criar mapas de características detalhados.

PSPNet O modelo em questão utiliza uma pirâmide de pooling para agregar contextos globais. A imagem é dividida em sub-regiões de diferentes tamanhos, e o pooling é realizado em cada sub-região. As características resultantes são concatenadas para formar um vetor de características enriquecido, que melhora a segmentação em cenários complexos.

LinkNet Este modelo é baseado em uma arquitetura de encoder-decoder com conexões de salto para combinar características de resolução baixa e alta. O encoder utiliza convoluções padrão para extrair características, enquanto o decoder aplica convoluções transpostas para recuperar a resolução da imagem. As conexões de salto ajudam a integrar informações detalhadas de diferentes níveis de resolução.

3.3.2. Configurações de Treino

Foram utilizados para treinamento dos modelos a função de perda Entropia Cruzada Focal Categórica (Focal Loss, FL) [Lin et al. 2017], que inclui um fator focal para reduzir o peso

de exemplos fáceis e focar mais em exemplos difíceis. A fórmula geral da FL é

$$\text{FL}(p_t) = (1 - p_t)^{\gamma} \cdot \log(p_t), \tag{1}$$

onde p_t é a probabilidade predita para a classe verdadeira (y_{true}). O fator $(1 - p_t)^{\gamma}$ ajusta a perda, e γ controla a ênfase em exemplos difíceis. A variante balanceada por α é dada por

$$\text{FL}(p_t) = -\alpha \cdot (1 - p_t)^{\gamma} \cdot \log(p_t), \tag{2}$$

onde α ajusta o peso das classes para lidar com o desbalanceamento.

A função de otimização utilizada foi a Adam (*Adaptive Moment Estimation*) [Kingma and Ba 2017].

Outro hiperparâmetro são os pesos das classes, ele pode ser útil para dizer ao modelo para "prestar mais atenção" em amostras de uma classe sub-representada. Este hyperparâmetro recebe um dicionário python de entrada em que cada chave é o número de uma classe, e o valor pertencente é o resultado do seguinte cálculo:

$$w_i = \frac{N}{C \cdot n_i}$$

onde w_i é o peso para a classe i, N é o número total de amostras no conjunto de dados, C é o número total de classes e n_i é o número de amostras da classe i.

O tamanho dos **batchs** para treino e validação foi de 8, esse valor foi definido com base no que era suportado pela memória do computador utilizado para o treinamento. A quantidade de **épocas** máximas executadas foi definida com o valor de 20, mas os modelos poderiam finalizar assim que convergissem com base nas análises dos callbacks. A taxa de aprendizado (Learning Rate, LR) inicial utilizada foi de 0.005, sendo ajustada de acordo com o callback de monitoramento do LR.

3.3.3. Callbacks

Callbacks são funções ou métodos que são chamados automaticamente durante o treinamento de um modelo de aprendizagem profunda para execução de ações em momentos específicos do treinamento.

Os callbacks utilizados durante o treinamento são descritos a seguir.

(i) Early Stopping: interrompe o treinamento se a métrica de validação não melhorar após várias épocas, evitando overfitting; (ii) Model Checkpoint: salva o modelo em pontos específicos do treinamento para recuperação do melhor estado observado; (iii) Reduce LR: diminui a taxa de aprendizado quando a métrica de desempenho estagnar, para uma convergência mais eficiente; (iv) Memory Cleaner: libera memória durante o treinamento para evitar erros de falta de recursos.

4. Resultados e Discussão

Na Tabela 1 estão presentes os resultados da acurácia média obtidos ao fim do treinamento utilizando o dataset de teste da UFES. Antes do cálculo da métrica, a região do *padding*

é removida das imagens de saída dos modelos e das imagens preditas com o objetivo de avaliar apenas a zona de interesse e fazer uma comparação justa entre os resultados da acurácia. Ao analisar os dados, observa-se que nenhum dos modelos conseguiu superar o resultado do modelo ENet, entretanto, o modelo U-Net ficou bem próximo com uma diferença de 4.34 p.p. Por outro lado, o modelo PSPNet foi o que teve o pior desempenho, com uma diferença de 10,45 pontos percentuais em comparação ao ENet.

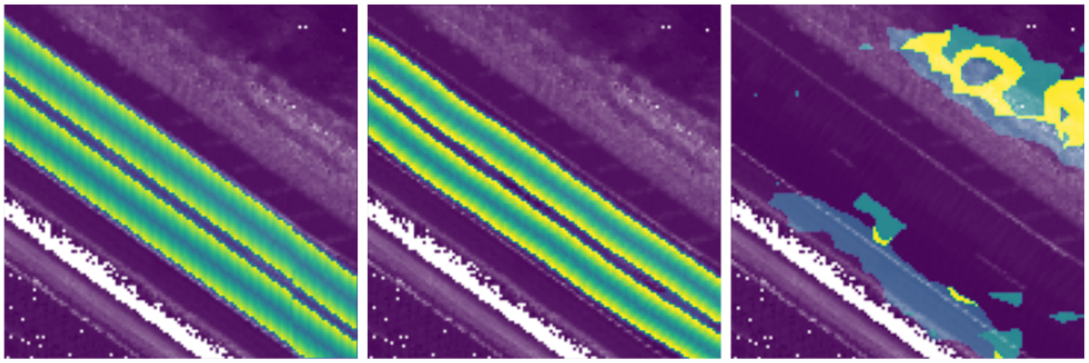
O elevado desempenho da ENet no dataset UFES pode estar relacionado ao fato de ter sido projetada especificamente para segmentação em ambientes urbanos. No entanto, apesar de a arquitetura LinkNet também ter sido desenvolvida com esse mesmo propósito, seu desempenho não foi satisfatório. As demais arquiteturas, por outro lado, foram originalmente concebidas para outros contextos: a U-Net, por exemplo, foi projetada para segmentação médica, enquanto a PSPNet e a FPN têm como foco a segmentação em aplicações mais gerais.

Tabela 1. Resultado dos testes com o dataset UFES

Modelos	Acurácia Média
ENet	83.70%
U-Net	79.34%
PSPNet	73.25%
FPN	77.20%
LinkNet	77.82%

Resultados diferentes foram obtidos no procedimento de teste utilizando o dataset HIGHWAY, dados não vistos pelos modelos. Os resultados finais estão apresentados na Tabela 2, onde é possível observar que apenas a U-Net superou o desempenho da rede ENet com diferença de 1.34 p.p. Outrora, o modelo PSPNet teve novamente um desempenho inferior com diferença de 7.95 p.p. A Figura 3 proporciona uma representação visual mais clara das diferenças de desempenho, onde (a) corresponde à saída esperada, (b) à saída gerada pela U-Net, e (c) à saída produzida pela PSPNet.

Figura 3. Ground-Truth (a), saída do modelo U-Net (b) e saída do modelo PSPNet (c) extraído do teste com o conjunto HIGHWAY.



(a) (b) (c)

Tabela 2. Resultado dos testes com o dataset HIGHWAY

Modelos	Acurácia Média
ENet	64.10%
U-Net	65.44%
PSPNet	56.15%
FPN	63.02%
LinkNet	63.24%

5. Conclusão

Neste estudo, foram comparados diferentes modelos de segmentação de estradas utilizando Redes Neurais Convolucionais Profundas (DCNNs), replicando os experimentos conduzidos por [Carneiro et al. 2018] com a rede ENet. Os resultados demonstraram que os modelos testados não atingiram o mesmo nível de desempenho que a ENet no dataset da UFES. Entretanto, em um contexto distinto, os desempenhos dos modelos foram bastante similares, com exceção do PSPNet, sendo que a U-Net superou a ENet.

Para pesquisas futuras, é proposto investigar a segmentação de estradas em mapas de remissão, utilizando outras técnicas e arquiteturas de redes neurais, com o objetivo de aprimorar os resultados obtidos neste domínio.

Referências

Badue, C., Guidolini, R., Carneiro, R. V., Azevedo, P., Cardoso, V. B., Forechi, A., Jesus, L., Berriel, R., Paixão, T. M., Mutz, F., de Paula Veronese, L., Oliveira-Santos, T., and De Souza, A. F. (2021). Self-driving cars: A survey. Expert Systems with Applications, 165:113816.

Bolimera, A., Muthalagu, R., Kalaichelvi, V., and Singh, A. (2023). Ego vehicle lane detection and key point determination using deep convolutional neural networks and inverse projection mapping. Transport and Telecommunication Journal, 24(2):110–119.

Carneiro, R. V., Nascimento, R. C., Guidolini, R., Cardoso, V. B., Oliveira-Santos, T., Badue, C., and De Souza, A. F. (2018). Mapping road lanes using laser remission and deep neural networks. In 2018 International Joint Conference on Neural Networks (IJCNN), pages 1–8. IEEE.

Chaurasia, A. and Culurciello, E. (2017). Linknet: Exploiting encoder representations for efficient semantic segmentation. CoRR, abs/1707.03718.

Fagnant, D. J. and Kockelman, K. (2015). Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations. Transportation Research Part A: Policy and Practice, 77:167–181.

Hao, S., Zhou, Y., and Guo, Y. (2020). A brief survey on semantic segmentation with deep learning. Neurocomputing, 406:302–321.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization.

Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B., and Belongie, S. J. (2016). Feature pyramid networks for object detection. CoRR, abs/1612.03144.

Lin, T., Goyal, P., Girshick, R. B., He, K., and Dollár, P. (2017). Focal loss for dense object detection. CoRR, abs/1708.02002.

Martínez-Díaz, M. and Soriguera, F. (2018). Autonomous vehicles: theoretical and practical challenges. Transportation Research Procedia, 33:275–282.

Organização Pan-Americana da Saúde (2021). Oms lança década de ação pela segurança no trânsito 2021-2030. OPAS.

Paden, B., Cap, M., Yong, S. Z., Yershov, D., and Frazzoli, E. (2016). A survey of motion planning and control techniques for self-driving urban vehicles.

Paszke, A., Chaurasia, A., Kim, S., and Culurciello, E. (2016). Enet: A deep neural network architecture for real-time semantic segmentation. arXiv preprint arXiv:1606.02147.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. CoRR, abs/1505.04597.

Schrank, D., Eisele, B., Lomax, T., and Bak, J. (2019). Urban mobility report.

Singh, R. and Rani, R. (2020). Semantic segmentation using deep convolutional neural network: A review. SSRN Electronic Journal, pages 1–8.

Wong, K., Gu, Y., and Kamijo, S. (2021). Mapping for autonomous driving: Opportunities and challenges. IEEE Intelligent Transportation Systems Magazine, 13(1):91–106.

Zhao, H., Shi, J., Qi, X., Wang, X., and Jia, J. (2016). Pyramid scene parsing network. CoRR, abs/1612.01105.