Uma análise comparativa de técnicas de detecção de pontos de parada em ambientes urbanos

Edgar Oliveira[1], **Clayson Celes**[2], **Carina Oliveira**[1], **Reinaldo Braga**[1]

[1] Instituto Federal do Ceará (IFCE)

[2]Universidade Federal de Minas Gerais (UFMG) & University of Ottawa

edgar.dos.santos.oliveira@outlook.com

Resumo. *Este artigo apresenta um framework para a criação de conjuntos de dados de referência (ground-truth) destinados à detecção automatizada de pontos de parada. O framework utiliza dados do OpenStreetMap e o SUMO (Simulation of Urban MObility) como fontes de informação essenciais. Além disso, são implementados e comparados métodos amplamente discutidos na literatura para a detecção de pontos de parada, utilizando conjuntos de dados gerados por meio desse framework. Os resultados da análise confirmam a confiabilidade dos métodos estudados. O estudo também introduz novos algoritmos à análise, que demonstram ser promissores na detecção de pontos de parada, além de identificar áreas para melhorias futuras. Destacam-se a necessidade de explorar análises adicionais que considerem métodos alternativos de aquisição de dados e avaliem seus impactos na detecção de pontos de parada.*

Abstract. *This article presents a framework for creating ground-truth datasets aimed at automated stop point detection. The framework utilizes data from OpenStreetMap and SUMO (Simulation of Urban MObility) as essential sources of information. Additionally, various widely discussed methods for stop point detection are implemented and compared using datasets generated through this framework. The analysis results confirmed previous findings, consolidating the reliability of the studied methods. Notably, the study also introduces new algorithms to the analysis, which prove to be promising in stop point detection. However, it is worth noting that this work identifies areas for future improvements. These include the need to explore additional analyses considering alternative data acquisition methods and evaluating their impacts on stop point detection.*

1. Introdução

Com o avanço constante das tecnologias, como a Internet das Coisas e redes sociais baseadas em localização, o volume de dados de mobilidade disponíveis tem aumentado de forma exponencial. Essa vasta quantidade de informações oferece uma oportunidade única para adquirir *insights* valiosos sobre a mobilidade de objetos, proporcionando uma compreensão mais profunda dos padrões de deslocamento em ambientes geoespaciais [Deng et al. 2021, Nazia et al. 2022].

No entanto, extrair informações significativas desses dados pode ser um desafio complexo, envolvendo várias tarefas [Zheng 2015]. Uma das etapas mais cruciais e

fundamentais nesse processo é a detecção de pontos de parada, cujo objetivo é categorizar os momentos em que um objeto está em movimento ou em repouso.

Embora a literatura apresente inúmeros métodos propostos para solucionar essa tarefa [Li et al. 2008, Nogueira et al. 2018, Aslak and Alessandretti 2020, Custers et al. 2021, Duarte and Sakr 2023], poucos trabalhos concentram-se na comparação abrangente dessas abordagens. Essa lacuna na pesquisa representa uma oportunidade para aprimorar a análise e a compreensão dos métodos, identificando suas fraquezas e pontos fortes, bem como suas características distintas que podem favorecer um método em detrimento de outro, em diferentes cenários e tipos de dados.

Nesse contexto, o presente artigo tem como objetivo realizar uma análise detalhada dos métodos propostos em estudos anteriores, oferecendo uma visão crítica e comparativa dessas abordagens. Através dessa avaliação, busca-se fornecer valiosos *insights* para a comunidade científica e profissionais interessados em processamento de dados de mobilidade, contribuindo para o avanço nessa área e para a identificação de estratégias mais eficazes na detecção de pontos de parada em dados geoespaciais. Além disso, o conjunto de dados usado para avaliar os métodos foi gerado por meio de um *framework* implementado e disponibilizado[1], a fim de auxiliar na obtenção de *ground-truth* para a detecção de pontos de parada.

2. Trabalhos Relacionados

Como mencionado, muitos trabalhos apresentam métodos para solucionar o problema de detecção de parada. No entanto, é comum encontrar premissas sobre o tipo de dados, seu estado e a forma como foram coletados e processados. Além disso, a literatura carece de uma análise focada na comparação profunda de métodos, o que se torna evidente nos trabalhos listados a seguir.

2.1. Infostop

A detecção de pontos de parada é uma tarefa crucial na mineração de mobilidade, e o algoritmo de Hariharan e Toyama [Hariharan and Toyama 2004] destaca-se como um dos mais utilizados para esse propósito. Esse algoritmo é dividido em duas etapas distintas. Na primeira fase, ele identifica pontos de parada considerando períodos em que um indivíduo não se afasta além de uma distância máxima, r_1, por uma duração mínima, t_{min}, ambos definidos como parâmetros. Na segunda etapa, os pontos de parada são agrupados, sendo o DBSCAN [Ester et al. 1996] a versão mais empregada para essa finalidade.

No entanto, surge o Infostop [Aslak and Alessandretti 2020] como uma nova alternativa na segunda etapa, superando algumas limitações. Diferentemente do DBSCAN, o Infostop evita a mesclagem de pontos de parada muito próximos, o que possibilita uma análise mais precisa das trajetórias de múltiplos usuários simultaneamente. Além disso, destaca-se por executar o segundo passo do algoritmo de forma mais eficiente, resultando em tempos de processamento inferiores aos de outros métodos disponíveis.

Uma área de melhoria perceptível deste trabalho é a inclusão de uma comparação entre diversos métodos de detecção de paradas, indo além das variações do método apresentado por Hariharan e Toyama.

[1] https://shorturl.at/fuwCW

2.2. Detecção de pontos de parada para mineração de similaridade entre usuários

No artigo [Li et al. 2008], é apresentado um método para demonstrar a similaridade entre usuários por meio do histórico de localizações. Para isso, são executados dois passos distintos: no primeiro passo, o artigo introduz um algoritmo destinado à detecção de pontos de parada; no segundo passo, é proposto um método de agrupamento dos pontos de parada em camadas, permitindo, assim, a visualização de diferentes níveis de similaridade entre os usuários. No contexto deste estudo, o método de detecção de pontos de parada assemelha-se consideravelmente ao apresentado por Hariharan e Toyama, porém sem a segunda fase de agrupamento. Nesse sentido, apenas dois critérios, temporal e espacial, são empregados para determinar se um objeto permaneceu numa determinada região por um período de tempo.

2.3. Movement-Stop-Noise [Nogueira et al. 2018]

Dentre os estudos relacionados presentes na literatura, este trabalho propõe uma abordagem baseada em métodos probabilísticos para a detecção de pontos de parada ao longo de trajetórias de objetos em movimento. A premissa fundamental desta abordagem é que os pontos que representam as trajetórias não se apresentam na mesma frequência.

O algoritmo desenvolvido é detalhadamente apresentado, destacando suas etapas e estratégias probabilísticas adotadas para a detecção de pontos de parada. Em seguida, uma seção de validação é apresentada, dividida em comparação qualitativa e quantitativa.

Na comparação qualitativa, são listados os parâmetros utilizados para classificar as trajetórias de acordo com oito algoritmos distintos. Além de parâmetros, a comparação leva em consideração o tratamento de ruído, o suporte a filtro espacial e a independência de dados externos. Já na comparação quantitativa, o algoritmo proposto é avaliado utilizando um conjunto de dados do *dataset* Dublin Bus GPS, em conjunto com informações do OpenStreetMap. Essa abordagem de validação com dados reais proporciona uma avaliação mais precisa do desempenho do algoritmo em cenários reais. No entanto, é importante mencionar que os resultados de outros algoritmos similares não foram incluídos nessa comparação, o que pode ser considerado uma limitação do estudo.

Uma possível melhoria metodológica discutida no artigo é o emprego do GTFS (General Transit Feed Specification) como fonte adicional de informação para a validação de pontos de parada.

2.4. *Stop Go Classifier* [Spang et al. 2022a]

Este trabalho apresenta um método para a detecção de pontos de parada com base em uma análise geométrica da trajetória. Para a detecção, o sistema inclui quatro métodos independentes de classificação, que comparam diferentes aspectos das propriedades geométricas de uma trajetória. Além disso, essa abordagem pode fazer uso de informações como acelerômetro, quando disponíveis. Para validar a ferramenta proposta, o conjunto de dados reunidos no STAGA foi utilizado para realizar uma comparação de resultados com métodos implementados em duas bibliotecas, Scikit-Mobility [Pappalardo et al. 2022] e MovingPandas [Graser and Dragaschnig 2020].

Contudo, é possível realizar uma comparação mais abrangente, já que as ferramentas citadas apresentam suporte somente a métodos baseados em distância,

duração e velocidade. Além disso, novas fontes de *ground-truth* podem ser empregadas para validar o classificador proposto.

2.5. Outlier Detection and Cleaning in Trajectories: A Benchmark of Existing Tools

No trabalho [Duarte and Sakr 2023], é apresentado um método para gerar *ground-truth*, ou seja, um método para gerar um conjunto de *targets* ou *labels* que posteriormente podem ser usados para treinar e testar um ou mais modelos, utilizando validação cruzada. Especificamente, esse método serve para gerar *labels* de '*outliers*' ou 'ruído' em quatro conjuntos de dados distintos, que serão utilizados para avaliar os métodos de detecção de *outliers*, e limpeza de trajetória, implementados em sete bibliotecas: (i)Movetk [Custers et al. 2021]; (ii) Moving Pandas [Graser and Dragaschnig 2020]; (iii) Scikit-mobility [Pappalardo et al. 2022]; (iv) Ptrail [Haidri et al. 2021]; (v) Pymove [Sanches 2019, Bráz 2020]; (vi) Argosfilter [Freitas and Freitas 2022]; (vii) Stmove [Seidel et al. 2019].

Após calcular a **acurácia**, **precisão**, **Recall** e **F-1 score**, o trabalho conclui que o MoveTk, seguido pelo Moving Pandas, apresentam os melhores resultados nos experimentos realizados. Contudo, também é mencionado que outros métodos de *ground-truth* podem ser explorados, assim como outros métodos que não estão implementados nessas bibliotecas.

Um ponto de melhoria identificado pelo *benchmark* está relacionado aos métodos de detecção de pontos de parada. Como descrito no artigo, a avaliação está centrada na detecção de *outliers* e limpeza de ruído, que são etapas de pré-processamento, diferentes da detecção de pontos de parada [Zheng 2015]. Desta forma, o estado da arte ainda carece de uma análise focada em métodos para a detecção de pontos de parada.

3. Metodologia

Nesta seção são apresentados os conjuntos de dados usados nos experimentos e a técnica de *ground-truth* aplicada. Além disso, são detalhadas as técnicas que serão comparadas e, quando necessário, alguns detalhes de implementação. Por fim, estão descritas as métricas de comparação. Uma imagem da estrutura geral da metodologia implementada encontra-se na Figura 1.

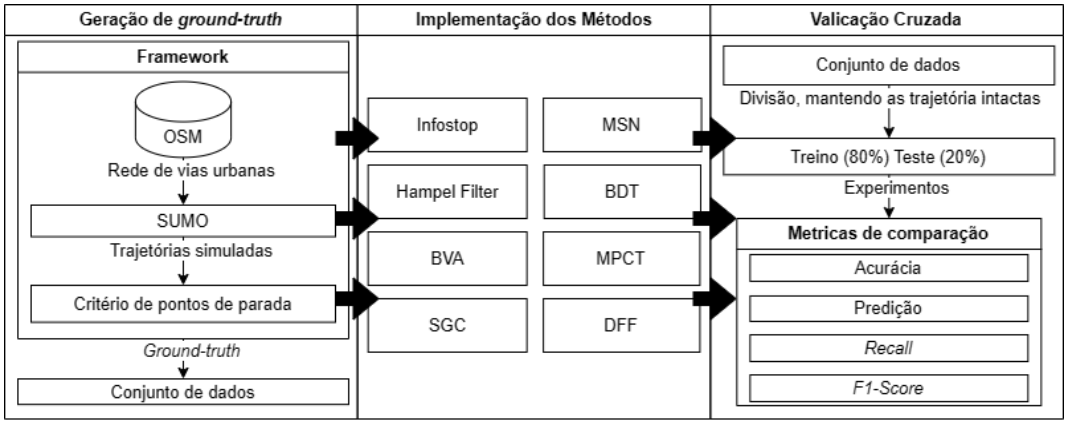

Figura 1. Estrutura geral da metodologia implementada.

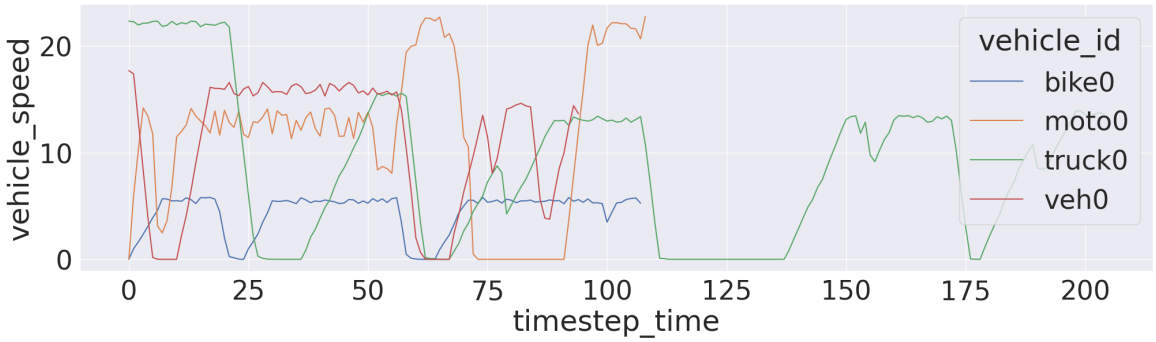

Figura 2. Amostra dos dados gerados.

3.1. Aquisição dos dados ou conjunto de dados

Nesta seção são apresentados os problemas nas opções disponíveis para *ground-truth* e o método utilizado e implementado neste artigo.

No estudo de [Spang et al. 2022b], um conjunto de dados de trajetórias GPS com anotações de pontos de parada foi registrado por meio de um aplicativo durante 126 dias. Os usuários do aplicativo, que eram voluntários com idades entre 25 e 35 anos, trabalhavam em tempo integral de forma presencial e residiam principalmente em áreas urbanas da Europa. As trajetórias coletadas foram realizadas principalmente de bicicleta, embora também tenham sido registradas viagens de carro e trem. O trabalho também incluiu pós-processamentos, como deslocamento temporal, espelhamento, rotação e deslocamento espacial, com o objetivo de proteger a privacidade dos voluntários. Todos esses dados foram coletados e disponibilizados com a finalidade de fornecer *ground-truth* para treinar e/ou testar modelos de detecção de pontos de parada. No entanto, é importante observar que os dados disponibilizados não contêm informações sobre o início e o fim das trajetórias, nem a identificação de objetos, o que pode dificultar sua usabilidade em algumas aplicações, como validação cruzada.

Para uma comparação justa entre os métodos de detecção de paradas, é necessário um conjunto de dados confiáveis. A ferramenta de código aberto *Simulation of Urban MObility* (SUMO) [Lopez et al. 2018] é amplamente utilizada para modelar e simular o tráfego urbano e sistemas de transporte. Desenvolvido pelo Instituto Alemão de Tecnologia dos Transportes (*German Aerospace Center* - DLR), o SUMO oferece uma plataforma flexível para simular o comportamento de veículos, pedestres e outros agentes de tráfego em ambientes urbanos e rodoviários. É amplamente utilizado por pesquisadores, engenheiros de tráfego, planejadores urbanos e desenvolvedores de sistemas de transporte para analisar o desempenho de infraestruturas viárias, estudar congestionamentos, testar algoritmos de controle de tráfego e avaliar o impacto de medidas de gerenciamento de tráfego e planejamento urbano.

Aproveitando essa ferramenta, juntamente com as informações de infraestrutura de mobilidade disponíveis no OpenStreetMap (OSM) [Vargas-Munoz et al. 2020], foi gerado e disponibilizado um conjunto de dados, exemplificado por uma amostra na Figura 2, para realizar os testes e comparações dos algoritmos de detecção de paradas. Por fim, também é disponibilizado o *framework* Python usado para realizar o fluxo de trabalho, incluindo a aquisição de dados do OSM, simulação usando o SUMO e a anotação dos

pontos de parada com base em um critério de velocidade definido pelo usuário. Para esta comparação, o critério de parada utilizado foi o de momentos em que os veículos simulados possuem velocidade igual a zero.

3.2. Detecção de Parada

Nesta seção, apresenta-se os métodos e detalhes de implementação, escolhidos para esta análise. A maior parte dos métodos listados, a seguir, foram implementados na linguagem de programação Python, e serão disponibilizados[2].

O algoritmo *Movement-Stop-Noise* envolve um processo de classificação de segmentos de trajetória em categorias de movimento, parada ou ruído. Inicialmente, ele realiza cálculos para determinar métricas como distância percorrida, duração, velocidade e ângulos de curva, que representam os ângulos entre três pontos vizinhos na trajetória. Essas métricas são utilizadas para categorizar os segmentos com base em limiares lineares, que são definidos como padrão em 3.5 para cada métrica. O algoritmo permite a personalização dos limiares e a inclusão de um ângulo mínimo opcional para aprimorar a detecção de ruídos. Neste trabalho, os limiares escolhidos foram zero para todas as métricas.

O Hampel Filter é uma técnica de filtragem robusta amplamente utilizada em processamento de séries temporais para detecção e suavização de *outliers*, pontos de dados que se desviam significativamente do comportamento esperado da série. Quando aplicado à análise de trajetórias, o Hampel Filter pode ser usado para identificar pontos de parada ao detectar valores discrepantes nos dados de localização. Funciona da seguinte maneira: em uma trajetória, os pontos de dados correspondem às coordenadas espaciais ou temporais da posição de um objeto. O filtro calcula uma estatística de dispersão, como a média móvel ponderada, para uma janela deslizante de pontos de dados. Quando um ponto de dados está muito afastado da estatística de dispersão calculada, é considerado um *outlier* e, portanto, pode ser identificado como um ponto de parada.

Um algoritmo eficaz para identificar pontos de parada em uma trajetória se baseia em um simples, mas poderoso conceito de *threshold*, que pode comparar diversos parâmetros, como distância percorrida, tempo decorrido, velocidade e aceleração. Utilizando uma abordagem de análise sequencial dos dados de movimento, o algoritmo examina cada ponto ao longo da trajetória, calculando a distância entre pontos consecutivos, medindo o tempo decorrido entre eles e derivando informações de velocidade e aceleração. Em seguida, estabelece limites para esses parâmetros, permitindo a detecção de pontos de parada com base em quando a velocidade cai abaixo de um certo valor mínimo, a aceleração se aproxima de zero e a distância percorrida fica relativamente constante por um período de tempo determinado. Os algoritmos que seguem esta abordagem estão entre os mais difundidos.

3.3. *Maximum Physically Consistent Trajectories*

Outra forma de identificar pontos de parada é através da verificação da consistência da trajetória sobre um modelo físico [Custers et al. 2021]. Neste trabalho uma versão deste algoritmo foi implementada usando como modelo físico um limite mínimo de velocidade. Neste método, a quantidade de parâmetros varia de acordo com o modelo físico escolhido,

[2]https://shorturl.at/fuwCW

por exemplo, na implementação desta comparação para identificar pontos de parada, apenas um limite inferior de velocidade, *threshold* igual a zero, foi usado. Desta forma, o algoritmo busca a maior sub-trajetória consistente com o modelo, para identificar os pontos de parada.

3.4. *Deep Feed Forward*

A tarefa de detecção de pontos de parada também pode ser encarada como um problema de classificação de trajetória. Desta forma, é possível aplicar uma rede neural simples, utilizando uma parte do conjunto de dados para treinar o modelo e outra parte para validação. Neste caso, o tipo de rede neural utilizado é um modelo que não faz uso dos pontos classificados anteriormente, e, portanto, pode resultar em um desempenho pior em comparação com modelos que fazem uso de resultados anteriores.

Por fim, os algoritmos avaliados nesta comparação são: *Movement-Stop-Noise* (MSN) [Nogueira et al. 2018]; Infostop [Aslak and Alessandretti 2020]; Hampel Filter (HF); Baseado em distância e tempo (BDT)[Li et al. 2008]; Baseado em velocidade e aceleração (BVC); *Maximum Physically Consistent Trajectories* (MPCT) [Custers et al. 2021]; *Stop Go Classifier* (SGC); e *Deep Feed Forward* (DFF).

3.5. Métricas de comparação

O conjunto de trajetórias geradas anteriormente foi dividido em duas partes para viabilizar a validação cruzada. A primeira, chamada de treino, que contém 80% das trajetórias e foi usada para treinar ou buscar os parâmetros mais adequados para a detecção. Uma vez treinados ou definidos os parâmetros, a segunda parte, chamada de teste, é usada para computar diversas métricas, que descrevemos a seguir. Por fim, é importante destacar que as trajetórias foram mantidas intactas no processo de divisão e, desta forma, não trazendo distorções para o desempenho dos algoritmos.

3.5.1. Métricas de avaliação

ACURÁCIA: No contexto de detecção de pontos de parada em trajetórias, a acurácia representa a proporção de trajetórias que foram corretamente classificadas como ”parada”ou ”não parada”pelo modelo de detecção. Uma alta acurácia indica que o modelo está acertando a maioria das trajetórias, identificando com precisão quando um veículo ou objeto está em um ponto de parada e quando não está.

PRECISÃO: A precisão no contexto de detecção de pontos de parada refere-se à capacidade do modelo de evitar a classificação errônea de trajetórias como pontos de parada quando, na verdade, não são. Isso é crucial para garantir que não haja alarmes falsos, o que pode ser problemático em sistemas de monitoramento de tráfego ou logística, onde a identificação incorreta de paradas pode levar a decisões inadequadas.

RECALL (REVOCAÇÃO): O *recall* é particularmente importante na detecção de pontos de parada em trajetórias, pois mede a capacidade do modelo de identificar todas as paradas reais. Em outras palavras, quantas das verdadeiras paradas foram corretamente identificadas pelo modelo. Um *recall* alto é essencial para garantir que não ocorram falsos negativos, ou seja, que nenhum ponto de parada real seja deixado de fora da detecção.

F1-SCORE: O F1-Score, no contexto de detecção de pontos de parada, avalia o equilíbrio entre a precisão e o *recall*. Um alto F1-Score indica que o modelo é capaz de encontrar a maioria das paradas reais enquanto mantém um baixo número de falsos positivos. Isso é crucial para sistemas de rastreamento de veículos, onde a identificação precisa de paradas é essencial para otimizar rotas e melhorar a eficiência operacional.

No contexto de pontos de parada, onde existem apenas dois estados para um ponto, parado ou não parado, e a quantidade de pontos de parada é frequentemente inferior à quantidade de pontos não parados, a métrica F1 tende a ter maior peso. Isso ocorre porque ela exige que o algoritmo classifique corretamente os pontos de parada e não classifique os pontos não parados de forma equivocada.

4. Resultados e Discussões

Nesta seção são apresentados os resultados obtidos a partir da realização de uma avaliação crítica das métricas coletadas através da aplicação da metodologia descrita anteriormente. Tais resultados, podem ser encontrados de forma resumida na Tabela 1 e na Figura 3.

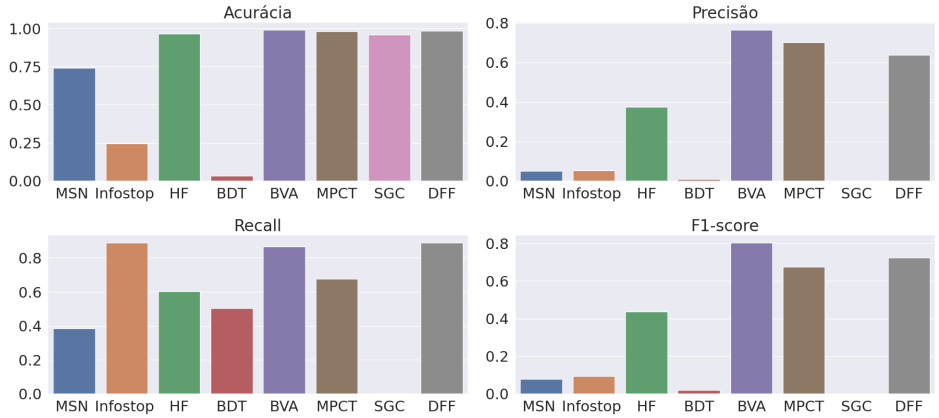

Figura 3. Comparação de Métricas por Método.

Tabela 1. Resultados

Método	MSN	Infostop	HF	BDT	BVA	MPCT	SGC	DFF
Acurácia	0.743	0.246	0.965	0.033	**0.991**	0.983	0.961	0.986
Precisão	0.051	0.053	0.376	0.010	**0.765**	0.701	0.000	0.639
Recall	0.386	**0.889**	0.603	0.504	0.865	0.677	0.003	**0.889**
F1-scores	0.079	0.093	0.438	0.020	**0.803**	0.676	0.000	0.723

Como pode-se observar, na Tabela 1, o algoritmo que obteve o melhor desempenho foi o Baseado em Velocidade e Aceleração, critério usado para a criação do *ground-truth* no conjunto de dados. No entanto, mesmo utilizando esse critério, não foram alcançadas métricas perfeitas, o que demonstra a não trivialidade da tarefa e a confiabilidade dos dados gerados pelo *framework* criado. Em relação ao BVA, os algoritmos DFF e o MPCT apresentaram resultados comparáveis. Além disso, o método DFF apresentou um *Recall* superior ao BVA, sem uma queda considerável de Precisão, indicando um resultado melhor quando há um peso maior para verdadeiros positivos.

A ferramenta Infostop apresentou um valor de *Recall* tão alto quanto DFF ao custo de uma queda considerável de Precisão. Portanto, o método considerou muitos pontos 'não parados' como 'parados' gerando um número considerável de falsos positivos.

Portanto, é desejável que trabalhos futuros aprimorem essa análise, incluindo não apenas novos métodos, mas também avaliando o impacto da combinação desses métodos com a aplicação de técnicas de pré-processamento, como o filtro de Kalman, por exemplo. Além disso, é importante realizar comparações que levem em consideração o tipo de ponto de parada, ou seja, a motivação por trás da interrupção do objeto. Essa comparação exigirá a criação de um *ground-truth* específico para esse propósito. Por fim, o desempenho razoável do uso de uma rede neural simples, DFF, torna desejável novas comparações que apliquem o uso de redes neurais, como Redes Neurais Recorrentes, e validem os modelos treinados com dados reais.

5. Conclusão

Neste artigo, foi apresentado e disponibilizado um *framework* capaz de criar *ground-truth*, utilizando dados do OpenStreetMap e do SUMO (Simulation of Urban MObility), para a detecção automática de pontos de parada, assim como o código-fonte implementado[3]. Usando um conjunto de dados gerados por meio desse *framework*, foram implementados e comparados alguns dos métodos mais utilizados descritos na literatura. Os resultados confirmaram trabalhos anteriores, como [Duarte and Sakr 2023], e introduziram novos algoritmos a comparação, SGC, MSN e DFF. Contudo, é importante destacar também que pontos de melhoria foram identificados, como a realização de novas análises que considerem métodos diferentes de aquisição de dados e seus impactos.

Referências

Aslak, U. and Alessandretti, L. (2020). Infostop: scalable stop-location detection in multi-user mobility data. *arXiv preprint arXiv:2003.14370*.

Bráz, M. C. (2020). Implementação de algoritmos para análise de similaridade de trajetória na biblioteca pymove. *Monografia. Universidade Federal do Ceará*.

Custers, B., Kerkhof, M. V. D., Meulemans, W., Speckmann, B., and Staals, F. (2021). Maximum physically consistent trajectories. *ACM Transactions on Spatial Algorithms and Systems*, 7(4):1–33.

Deng, D., Leung, C. K., Zhao, C., Wen, Y., and Zheng, H. (2021). Spatial-temporal data science of covid-19 data. In *2021 IEEE 15th International Conference on Big Data Science and Engineering (BigDataSE)*, pages 7–14. IEEE.

Duarte, M. M. and Sakr, M. (2023). Outlier detection and cleaning in trajectories: A benchmark of existing tools. In *EDBT/ICDT Workshops*.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

Freitas, C. and Freitas, M. C. (2022). Package 'argosfilter'.

[3]https://shorturl.at/fuwCW

Graser, A. and Dragaschnig, M. (2020). Exploring movement data in notebook environments. In *IEEE VIS 2020 - MoVis*.

Haidri, S., Haranwala, Y. J., Bogorny, V., Renso, C., da Fonseca, V. P., and Soares, A. (2021). Ptrail–a python package for parallel trajectory data preprocessing. *arXiv:2108.13202*.

Hariharan, R. and Toyama, K. (2004). Project lachesis: parsing and modeling location histories. In *International Conference on Geographic Information Science*, pages 106–124. Springer.

Li, Q., Zheng, Y., Xie, X., Chen, Y., Liu, W., and Ma, W.-Y. (2008). Mining user similarity based on location history. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, pages 1–10.

Lopez, P. A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y.-P., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., and Wießner, E. (2018). Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE.

Nazia, N., Butt, Z. A., Bedard, M. L., Tang, W.-C., Sehar, H., and Law, J. (2022). Methods used in the spatial and spatiotemporal analysis of covid-19 epidemiology: a systematic review. *International Journal of Environmental Research and Public Health*, 19(14):8267.

Nogueira, T. P., Celes, C., Martin, H., Loureiro, A. A., and Andrade, R. M. (2018). A statistical method for detecting move, stop, and noise: A case study with bus trajectories. *Journal of Information and Data Management*, 9(3):214–214.

Pappalardo, L., Simini, F., Barlacchi, G., and Pellungrini, R. (2022). scikit-mobility: A python library for the analysis, generation, and risk assessment of mobility data. *Journal of Statistical Software*, 103(4):1–38.

Sanches, A. d. J. A. M. (2019). Uma arquitetura e implementação do módulo de pré-processamento para biblioteca pymove.

Seidel, D. P., Dougherty, E. R., and Getz, W. M. (2019). Exploratory movement analysis and report building with r package stmove. *bioRxiv*, page 758987.

Spang, R., Pieper, K., Oesterle, B., Brauer, M., Haeger, C., Mümken, S., Gellert, P., and Voigt-Antons, J.-N. (2022a). Making sense of the noise: integrating multiple analyses for stop and trip classification. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48:435–441.

Spang, R., Pieper, K., Oesterle, B., Brauer, M., Haeger, C., Mümken, S., Gellert, P., and Voigt-Antons, J.-N. (2022b). The staga-dataset: Stop and trip annotated gps and accelerometer data of everyday life. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48:443–448.

Vargas-Munoz, J. E., Srivastava, S., Tuia, D., and Falcao, A. X. (2020). Openstreetmap: Challenges and opportunities in machine learning and remote sensing. *IEEE Geoscience and Remote Sensing Magazine*, 9(1):184–199.

Zheng, Y. (2015). Trajectory data mining: an overview. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 6(3):1–41.