Capítulo 3

Tratamento e Análise de Dados de Mobilidade Urbana: Uma metodologia teórica e prática

Edgar Oliveira, Clayson Celes, Carina Oliveira, Reinaldo Braga

Abstract

The growing availability of mobility data in the urban domain brings unprecedented opportunities for knowledge extraction and decision-making in several areas. However, obtaining valuable insights from a large volume of raw data is not a trivial task and requires a detailed process involving techniques, tools, and insights. The aim of this chapter is to present the foundations for the treatment and analysis of mobility data through a proposed methodology that encompasses such a process. Also, it is performed a comparison among Python libraries available in the literature that provide resources for mobility analysis. The libraries are categorized and associated with specific phases of the methodology. The limitations of these libraries are identified, including the lack of support for basic data processing tasks as well as the absence of solutions for trajectory classification, anomaly event detection, and anomalous trajectory detection. These limitations are opportunities for future research and the development of more comprehensive and enhanced methods for mobility analysis. Lastly, we present a real-world use case to exemplify the practical approach, providing results and corresponding source code.

Resumo

A crescente disponibilidade de dados de mobilidade no domínio urbano traz oportunidades sem precedentes para a extração de conhecimento e a tomada de decisão em diversas áreas. No entanto, obter informações valiosas a partir de um grande volume de dados brutos não é uma tarefa simples e exige um processo detalhado com técnicas, ferramentas e insights. O objetivo deste capítulo é apresentar os fundamentos para o tratamento e a análise de dados de mobilidade por meio de uma metodologia proposta que engloba tal processo. Também é realizada uma comparação entre bibliotecas Python disponíveis na literatura que oferecem recursos para a análise de mobilidade. As bibliotecas são classificadas e associadas a fases específicas da metodologia. As limitações das

bibliotecas são identificadas, incluindo a falta de suporte para tarefas básicas de tratamento de dados, bem como ausências de soluções para a classificação de trajetórias, detecção de eventos atípicos e detecção de trajetórias anômalas. Essas limitações são oportunidades para pesquisas futuras e o desenvolvimento de métodos mais abrangentes e aprimorados para a análise de mobilidade. Por fim, um caso de uso com dados reais é empregado para exemplificar a abordagem prática, fornecendo resultados e o código-fonte correspondente.

3.1. Introdução

A disciplina de Ciência de Dados tem ganhado significativa importância em diversas áreas. No contexto de mobilidade urbana, diversas iniciativas têm evidenciado como o conhecimento derivado da análise de dados pode ser altamente benéfico [7, 41, 21]. No entanto, a extração desse conhecimento nem sempre se revela uma tarefa simples e, frequentemente, demanda um conhecimento multidisciplinar. Essa complexidade é ampliada quando se trata de dados de mobilidade [44], que, na maioria das vezes, são dados não estruturados, contendo ruídos, além de serem suscetíveis a imprecisões e lacunas. Consequentemente, a comunidade científica tem dedicado consideráveis esforços no desenvolvimento de ferramentas e bibliotecas destinadas a superar os desafios inerentes ao processo de mineração de dados nesse contexto.

A linguagem de programação Python tem sido adotada no processo de aquisição de conhecimento a partir de dados, especialmente quando se trata de bibliotecas de código aberto. Essa adoção generalizada também se reflete no campo da mobilidade urbana, no qual bibliotecas voltadas para a mineração de dados de mobilidade são amplamente utilizadas [31]. Essa popularidade pode ser atribuída à extensa comunidade que cerca a linguagem, à sua facilidade de uso e ao vasto ecossistema de bibliotecas disponíveis para uma ampla gama de tarefas na ciência de dados, incluindo Pandas [26] e Numpy [13].

Na área de mineração de dados geográficos, que requer a execução de tarefas específicas, encontra-se a biblioteca GeoPandas [16], desenvolvida com base na biblioteca Pandas e desempenhando uma função semelhante para dados geoespaciais. Especificamente no campo da mobilidade, o ecossistema dessas bibliotecas oferece uma variedade de ferramentas valiosas para análise e exploração de dados de mobilidade urbana.

Table 3.1. Bibliotecas para tratamento e análise de dados de mobilidade.

Trackintel [20]	Yupi [27]
Mobilipy[25]	Mobvis [33]
PTRAIL [12]	Mobilkit [39]
Traja [32]	MovingPandas [10]
Mobipy [19]	PyMove [3]
Scikit-Mobility [24]	Tracktable [28]
SMAFramework [30]	mocha [34]
Teetool [8]	

O ecossistema Python oferece diversas bibliotecas para tratamento e a análise de dados de mobilidade urbana, conforme detalhado na Tabela 3.1. Apesar de alguns estudos

realizarem comparações entre algumas dessas bibliotecas disponíveis, a literatura carece de uma análise quantitativa abrangente do estado da arte das bibliotecas Python de código aberto para análise de mobilidade urbana. Uma outra deficiência observada na literatura é a falta de um estudo que apresente uma metodologia e a aplique em um cenário de uso prático, fazendo uso das ferramentas disponíveis no ecossistema de bibliotecas Python. Nesse sentido, o presente estudo abrange desde a teoria até a prática.

Diante desse cenário, o objetivo deste capítulo é apresentar as particularidades e técnicas envolvidas no tratamento e análise de dados de mobilidade urbana. Isso inclui abordar o estado atual da arte e realizar uma comparação quantitativa das bibliotecas disponíveis, além de oferecer uma visão geral de estudos específicos que destacam as tendências de pesquisa e os principais desafios nessa área. Adicionalmente, apresenta-se uma metodologia para a aplicação da análise de dados em projetos relacionados ao contexto urbano, permitindo que os interessados se familiarizem com a área através de um guia prático de trabalho, como ilustrado na Figura 3.1.

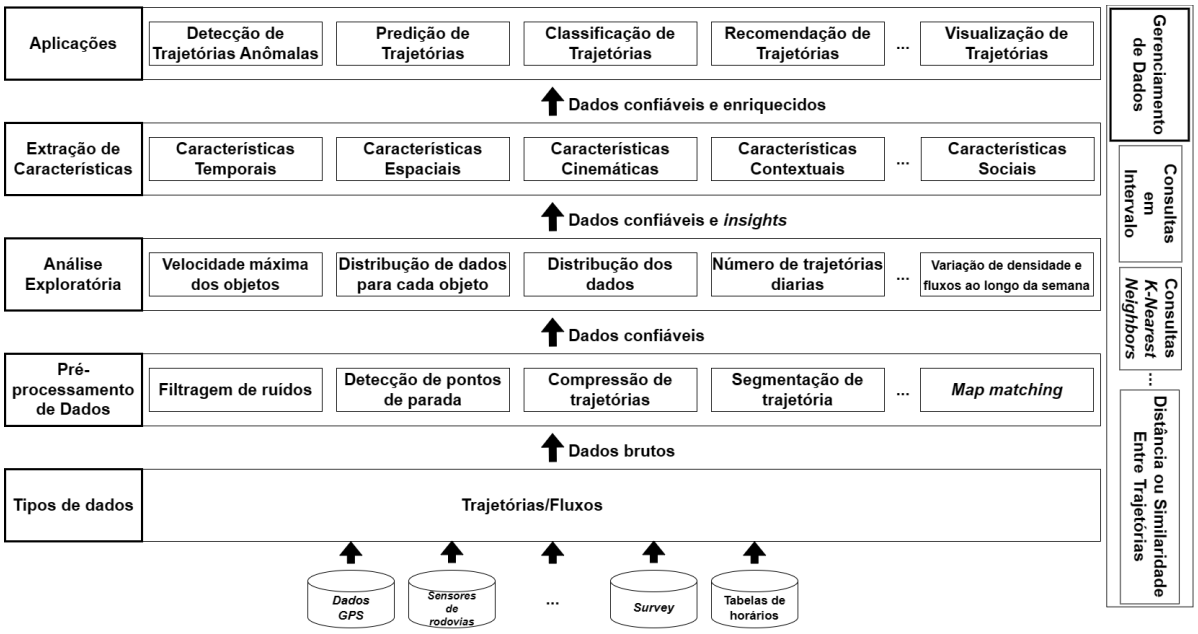

Figure 3.1. Um arcabouço para tratamento e análise de dados de mobilidade.

O capítulo está estruturado da seguinte forma: Na Seção 2 é introduzida a metodologia proposta. A Seção 3 aborda o processo de comparação quantitativa. Na Seção 4 são mostrados os resultados. Por fim, na Seção 5 é realizada uma discussão sobre o conteúdo deste capítulo.

3.2. Metodologia para Tratamento e Análise de Mobilidade

Esta seção aborda a metodologia usada, descrevendo os passos essenciais para adquirir conhecimento, derivados de um paradigma sólido [44], a serem ajustados para incluir outras entidades. A estrutura de tarefas subdivididas permanece, facilitando a busca por recursos nas bibliotecas. Além disso, seu propósito é apresentar informações significativas que vão além do código, fornecendo uma visão abrangente dos contribuintes na

manutenção das bibliotecas. Alguns tópicos relevantes para esta metodologia são: (i) Tipos de fontes geradoras de dados de mobilidade; (ii) Pré-processamento de dados; (iii) Gerenciamento de dados; (iv) Tratamento de incerteza; (v) Mineração de padrões de trajetórias; (vi) Classificação de trajetórias; (vii) Detecção de anomalias em uma trajetória; (viii) Suporte à visualização de mobilidade.

Apesar de o paradigma proposto por Yu Zheng apresentar uma delimitação consistente das categorias de tarefas relacionadas à análise de trajetórias, é importante destacar que a mobilidade pode ser representada por meio de outras entidades. Portanto, com o objetivo de abranger essas diversas entidades, como fluxos, é necessário definir as características das entidades e como as bibliotecas interagem com elas em um momento posterior.

Adicionalmente, reestruturamos o paradigma proposto, mantendo suas subdivisões, de modo a representá-lo por meio de perguntas. Essa abordagem facilita a busca por recursos específicos nas bibliotecas mencionadas, tornando o processo mais eficiente e direcionado.

3.2.1. Tipos de fontes geradoras de dados de mobilidade

De acordo com o trabalho de referência [4], os principais tipos de fontes de dados de mobilidade de veículos podem ser classificados em seis categorias principais, conforme listado a seguir:

- ***Survey***: Estes dados são coletados por meio de questionários respondidos por indivíduos específicos, abrangendo informações como origem e destino da viagem, tempo de viagem, frequência e rotas. Geralmente, são realizados em intervalos de tempo espaçados.

- ***Schedule***: Estes dados são obtidos a partir de agendas pré-determinadas de veículos, geralmente abrangendo um único tipo de transporte, como ônibus.

- ***Road Sensors***: Estes dados são adquiridos por sensores instalados nas margens ou ao longo das vias. Podem fornecer informações como contagem de veículos, fluxo de tráfego e velocidade.

- ***Closed-Circuit Television***: Estes dados consistem em imagens capturadas por câmeras em veículos ou nas vias, que podem ser usadas para extrair informações por meio de técnicas de visão computacional. Assim como os *Road Sensors*, eles têm a desvantagem de exigir um alto custo de infraestrutura para instalação e o monitoramento é restrito a estradas e áreas específicas.

- ***Global Navigation Satellite System (GNSS)***: Dispositivos equipados com GNSS, como Sistema de Posicionamento Global (*Global Positioning System* - GPS), podem registrar a posição geográfica durante o movimento. Portanto, essa fonte de dados fornece informações de posicionamento refinadas e movimento quase em tempo real de um determinado veículo. Geralmente, dados de veículos particulares não são públicos, sendo mais comuns em táxis e ônibus.

- ***Sensing-as-a-Service (SaaS)***: Estas são plataformas que fornecem informações gerais sobre dados de tráfego, geralmente usando técnicas de fusão de dados para classificar a intensidade e fornecer informações de tráfego em ruas específicas, como apresentado no Google Maps ou Waze.

3.2.2. Módulos

Com base na análise nas subdivisões de processos para analisar trajetórias, retiraremos perguntas que mantenham as mesmas tarefas e que facilitem a busca de recursos por parte do leitor. Tais tarefas, descritas em [44] são descritas a seguir e, posteriormente, apresentadas às perguntas.

Trajetórias podem ser representadas de várias maneiras, incluindo pontos geoespaciais, fluxos, imagens, vídeos e texto. Este capítulo lida exclusivamente com trajetórias de GPS e fluxos, conforme definido a seguir.

Neste capítulo, a representação de trajetória utilizada é a de uma sequência de pontos, onde cada elemento P_i pertencente a P consiste em uma tupla com latitude, longitude e tempo. Desta forma, a trajetória é representada por uma sequência de pontos geoespaciais ordenados pelo tempo em que foram coletados, como ilustrado na Figura 3.2.

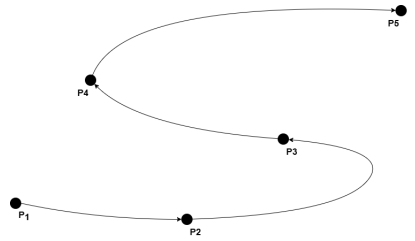

Figure 3.2. Ilustração de uma trajetória.

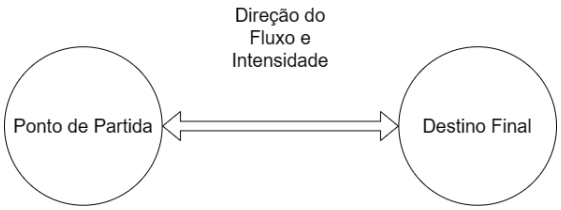

Figure 3.3. Ilustração de fluxo.

Conforme ilustrado na Figura 3.3, neste capítulo, um fluxo refere-se a pontos geoespaciais que podem ou não ter uma direção associada, acompanhados da intensidade do fluxo que ocorre entre esses dois pontos.

3.2.2.1. Classificar as fontes geradoras de trajetória

Durante esta fase da análise de trajetória, é necessária a classificação das fontes geradoras de dados em quatro grandes grupos: mobilidade de pessoas, mobilidade de veículos, mobilidade de animais e mobilidade de eventos naturais. Entretanto, atualmente, não existe nenhuma biblioteca disponível que ofereça funcionalidades para essa tarefa, o que torna impossível a extração de perguntas relacionadas. É importante ressaltar que a classificação das fontes geradoras de dados geralmente é realizada pela entidade responsável pela coleta dos dados e é disponibilizada como metadados.

Adicionalmente, é fundamental levar em consideração que a coleta de dados é uma tarefa complexa e a presença de incertezas é inevitável. Portanto, muitas bibliotecas

apresentam soluções para mitigar esse problema. É importante, assim, considerar a capacidade da biblioteca de representar diferentes tipos de mobilidade e a maneira como os dados são gerenciados.

Diante desta etapa podemos extrair as seguintes Questões de Pesquisa (QP):

QP1) Quais tipo(s) de dados de mobilidade (por exemplo, ponto(s) GPS, segmento(s)) estão(ão) disponível(is) como representação(ões) para análise nas bibliotecas?

QP2) Explora qual(is) tipo(s) de representação de mobilidade?

3.2.2.2. Pré-processamento de Dados

Este estágio da análise de trajetória envolve o tratamento dos dados coletados na etapa anterior. As tarefas aqui se dividem em cinco categorias, descritas a seguir:

FILTRAGEM DE RUÍDOS. Os dados de trajetória frequentemente apresentam erros devido a diversos fatores, tais como a qualidade dos sensores ou falhas de comunicação. Nestes casos, é necessário remover as falhas da trajetória usando algoritmos que se dividem em três grandes subcategorias: *Mean Filter*, Kalman e *Heuristics-Based Outlier Detection*.

DETECÇÃO DE PONTOS DE PARADA. Em uma trajetória, alguns pontos podem revelar informações importantes, como uma parada para abastecer o carro, chamados de pontos de parada. Em alguns casos, é necessário extrair esses pontos para melhorar a acurácia, enquanto em outros, como para estimar o tempo de viagem, é preciso removê-los e computar apenas os deslocamentos.

COMPRESSÃO DE TRAJETÓRIAS. Para tornar o envio, armazenamento e computação de trajetórias mais eficientes, é necessário reduzir o número de pontos sem perder muito de sua precisão. A literatura apresenta diversos algoritmos para realizar esta tarefa, incluindo *Distance Metric*, *Offline Compression*, *Online Data Reduction* e *Compression with Semantic Meaning*.

SEGMENTAÇÃO DE TRAJETÓRIA. Além de reduzir o poder computacional necessário para a análise, a divisão de trajetórias em partes menores permite extrair padrões de subtrajetórias e enriquecer o conhecimento. A segmentação ou clusterização de trajetória pode ser feita com base em espaços temporais, mudanças de ângulo e usando algoritmos de simplificação de linha.

***Map matching*.** O processo de *map-matching* consiste em reposicionar pontos de uma trajetória em uma estrada ou rodovia, corrigindo erros causados por imprecisão na coleta dos dados. As informações adicionais utilizadas ou o intervalo de pontos de amostragem podem ser levados em conta durante o processo.

Diante desta etapa podemos extrair a seguinte Questão de Pesquisa (QP):

QP3) Qual(is) técnica(s) de pré-processamento esta(ão) implementada(s)?

3.2.2.3. Gerenciamento de Dados

Esta etapa é especialmente importante à medida que a quantidade de dados analisados aumenta, já que o acesso a várias partes de trajetórias em um grande volume de dados pode ser oneroso. Assim, é desejável que as ferramentas utilizadas para análise possuam um gerenciamento eficiente de dados e ofereçam técnicas de busca eficazes, tais como as listadas a seguir.

CONSULTAS EM INTERVALO. Existem diversas abordagens para o gerenciamento eficiente de trajetórias. Uma delas é organizar as trajetórias em uma estrutura de árvore como *3D-Rtree*, o que permite realizar buscas espaço-temporais representadas com uma caixa 3D, onde o resultado são os nós da árvore que estiverem dentro da caixa. Outra abordagem possível é dividir um período em vários intervalos de tempo, construindo um índice espacial individual como uma *R-tree* para as trajetórias geradas em cada intervalo. Por fim, é possível particionar um espaço geográfico em grades e construir um índice para as trajetórias que caem em cada grade.

CONSULTAS *K-Nearest Neighbors* (KNN). A abordagem de encontrar k trajetórias com a menor distância agregada em relação a alguns pontos podem ser útil em diversas situações, como na identificação de todos os carros que passaram por determinada estrada ou rodovia. Essa técnica é especialmente importante em análises de dados de mobilidade urbana, permitindo uma compreensão mais precisa do tráfego em uma determinada região e fornecendo informações valiosas para o planejamento urbano.

DISTÂNCIA OU SIMILARIDADES ENTRE TRAJETÓRIAS. A similaridade entre duas trajetórias é frequentemente calculada por meio da distância agregada entre os pontos, utilizando algoritmos como *Closest-Pair Distance*, *Sum-of-Pairs Distance*, *Dynamic Time Wrapping*, *Longest Common Sub-Sequence* (LCSS), *LCSS-based Distance*, entre outros.

Diante desta etapa podemos extrair a seguinte Questão de Pesquisa (QP):

QP4) Qual(is) recurso(s) para gerenciamento de dados esta(ão) disponível(is)?

3.2.2.4. Incerteza

Como o movimento de um objeto no espaço é contínuo, enquanto seu registro é discreto, as informações sobre uma trajetória sempre terão lacunas e, portanto, incertezas. Para reduzir essa incerteza, existem diversas técnicas, bem como métodos para aumentar a incerteza e proteger a privacidade do usuário.

REDUZINDO A INCERTEZA DE DADOS DE TRAJETÓRIA. Existem duas categorias principais de métodos para gerenciar a incerteza em trajetórias. A primeira envolve modelos que recuperam trajetórias existentes usando diferentes tipos de busca na base de dados, como por meio de objetos geométricos ou funções de densidade e probabilidade independentes em cada ponto do tempo. Já a segunda categoria envolve a inferência de caminhos a partir das lacunas existentes, interpolando os pontos faltantes para construir a trajetória [5]. Em algumas situações, é possível melhorar as inferências utilizando algoritmos adicionais, como o *map-matching*.

PRIVACIDADE DE DADOS DE TRAJETÓRIA. Serviços baseados em localização em tempo real podem revelar informações sensíveis sobre a localização de um usuário. Para evitar isso, existem diversas abordagens, como o uso de técnicas de ocultação de localização, como *spatial cloaking* [6] e *mix-zones* [2]. No caso de trajetórias históricas, publicar múltiplas trajetórias de um mesmo indivíduo pode permitir a inferência de informações sensíveis, como sua residência ou local de trabalho. Algoritmos como *clustering-based* [1], *generalization-based* [23], *suppression-based* [38], *grid-based* [9] e outros podem ser aplicados para proteger a privacidade em tais cenários.

A partir desta etapa, podemos extrair as seguintes perguntas:

QP5) Qual(is) método(s) para lidar com a incerteza esta(ão) implementado(s)?

QP6) É possível gerar dados de mobilidade?

3.2.2.5. Mineração de Padrões de Trajetória

A extração de padrões, seja de uma única trajetória ou de um conjunto delas, se divide em quatro grandes categorias, que descrevemos a seguir.

***Moving Together*.** Mostrar que um grupo de objetos se movem juntos durante um intervalo de tempo pode ser feito usando algoritmos como *flock* [11], *convoy* [14, 15], *swarm* [18], *travelling companion* [37, 36] e *gathering* [42, 43].

***Trajectory Clustering*.** Ao buscar por um caminho representativo ou uma tendência compartilhada por um conjunto de trajetórias, é comum agrupá-las em *clusters*. No cenário de redes rodoviárias, existem vários estudos sobre o agrupamento de trajetórias, como o trabalho de [17]. Para solucionar esse problema, pode-se combinar *map-matching* com algoritmos de agrupamento de grafos.

MINERAÇÃO DE PADRÕES SEQUENCIAIS DE TRAJETÓRIA. Na análise de trajetórias, um padrão sequencial é identificado quando um determinado número de objetos em movimento percorre uma sequência comum de locais em um intervalo de tempo semelhante. Para que uma sequência seja considerada um padrão, geralmente é necessário que ela seja encontrada mais vezes do que um valor de limiar (*threshold*) previamente estabelecido.

MINERAÇÃO DE PADRÕES PERIÓDICOS DE TRAJETÓRIAS. É comum encontrar padrões periódicos em dados de trajetória, pois os locais que uma pessoa visita diariamente tendem a se repetir. Encontrar esses padrões podem ser útil para prever movimentos futuros de objetos, além de ajudar na compressão de suas trajetórias, reduzindo o espaço necessário para armazená-las. Diversos algoritmos têm sido desenvolvidos para a detecção de padrões periódicos, como *Fourier transform* [2] e algoritmos baseados em *clustering* [22].

Diante desta etapa podemos extrair a seguinte Questão de Pesquisa (QP):

QP7) Qual(is) técnica(s) de mineração de padrões de trajetórias esta(ão) implementada(s)?

3.2.2.6. Classificação de Trajetória

A classificação de trajetórias tem como objetivo encontrar diferenças entre trajetórias ou segmentos de trajetórias em diferentes aspectos, como movimentos, modos de transporte e atividades humanas. Tipicamente, esse processo é dividido em três etapas: primeiro, as trajetórias são divididas em segmentos usando métodos de segmentação; em seguida, características são extraídas desses segmentos; e, finalmente, um modelo é criado para classificar cada segmento. A segmentação pode ser baseada em vários critérios, como a velocidade ou a direção da trajetória. As características podem ser extraídas de várias maneiras, como a distância percorrida, a aceleração e a direção. Várias técnicas de aprendizado de máquina são usadas para criar modelos de classificação, incluindo KNN, *Support Vector Machine* (SVM) e redes neurais.

Diante desta etapa podemos extrair a seguinte Questão de Pesquisa (QP):

QP8) Qual(is) técnica(s) para predição ou classificação de trajetórias esta(ão) implementada(s)?

3.2.2.7. Detecção de Anomalias em uma Trajetória

A detecção de anomalias em trajetórias é uma tarefa importante em diversas aplicações, como monitoramento de veículos e detecção de comportamentos suspeitos. As anomalias podem ser identificadas como pontos ou segmentos que se destacam dos demais em relação a alguma métrica, como a distância espacial percorrida ou a quebra de um padrão encontrado nas demais trajetórias. Existem diversas técnicas para a detecção de anomalias, que variam desde a análise visual até o uso de modelos estatísticos e de aprendizado de máquina. A detecção de anomalias pode ser realizada tanto em tempo real quanto em trajetórias históricas, permitindo a identificação de eventos incomuns e a tomada de decisões mais eficientes.

DETECTANDO TRAJETÓRIAS ATÍPICAS. Trajetórias atípicas são aquelas que se diferenciam das demais em termos de forma ou tempo de viagem. Essas trajetórias podem ser detectadas por meio de algoritmos de *clustering* baseados em densidade ou por meio de mineração de padrões de frequência. Se uma trajetória ou segmento não se encaixa em nenhum *cluster* ou não é frequente o suficiente, é considerada uma anomalia.

DETECÇÃO DE EVENTOS ATÍPICOS EM TRAJETÓRIAS. Além da detecção de anomalias em trajetórias individuais, é possível identificar eventos anormais por meio do uso de um grande número de trajetórias. Uma abordagem para detectar anomalias no trânsito é representar o evento como um subgrafo das redes de rodovias da cidade, onde os motoristas exibem comportamentos diferentes de seus padrões históricos. É possível, então, descrever o evento por meio da mineração de termos significativos postados em redes sociais no local da ocorrência.

Diante desta etapa podemos extrair as seguintes Questões de Pesquisa (QP):

QP9) Quais técnicas para detecção de anomalias estão implementadas?

3.2.2.8. Suporte à Visualização de Dados de Trajetória

A visualização de trajetórias não se limita apenas à exibição de dados brutos, como em uma tabela na biblioteca [26]. É importante destacar que a exibição simples de dados pode não ser esclarecedora para os usuários, pois não explora as informações geoespaciais. Portanto, é fundamental que haja métodos que enriqueçam as informações geográficas, por meio de imagens, gráficos e mapas interativos.

Diante desta etapa podemos extrair a seguinte Questão de Pesquisa (QP):

QP10) Existe suporte para visualização?

3.3. Bibliotecas e uma Comparação Qualitativa

Nesta seção é realizada uma descrição e uma comparação de bibliotecas para tratamento e análise de mobilidade. Essa comparação se torna essencial para destacar os métodos presentes no estado da arte das bibliotecas do ecossistema Python, bem como para identificar as fortalezas e fraquezas de cada ferramenta ao longo das diferentes etapas da descoberta de informações. Dessa forma, diversas oportunidades de melhoria e trabalhos futuros podem ser reveladas.

3.3.1. Descrição das Bibliotecas

Nesta seção, são apresentadas as bibliotecas listadas na Tabela 3.1. Além disso, métricas comparativas com base nas informações disponíveis no GitHub podem ser encontradas juntamente com o código-fonte do capítulo[1]. Dessa forma, mais informações estão disponíveis externamente para manter o capítulo conciso.

TRACKINTEL [20]: O *Trackintel* tem como objetivo preencher a lacuna de ferramentas voltadas para a etapa de pré-processamento, fornecendo recursos para padronizar essas tarefas e aumentar a reprodutibilidade e comparabilidade de estudos científicos. Embora a mineração de trajetórias não seja o seu foco principal, a biblioteca apresenta algumas limitações em relação a outras bibliotecas em termos de métodos disponíveis para a etapa de mineração.

YUPI [27]: O *Yet Underused Path Instruments* (*Yupi*) é uma biblioteca Python para mineração de trajetórias que busca adicionar novos recursos enquanto aproveita recursos já existentes. Para isso, *Yupi* oferece APIs e métodos que permitem o uso de outras bibliotecas para análise de trajetórias, além de disponibilizar recursos próprios, como a extração de trajetórias de vídeos. Além disso, a biblioteca é altamente configurável e extensível, permitindo que os usuários personalizem o processo de análise de acordo com suas necessidades.

MOBILIPY [25]: Esta biblioteca é uma ferramenta completa para análise de mobilidade, fornecendo recursos para todos os estágios do processo de mineração. Além de permitir a segmentação de trajetórias em viagens e atividades, com detecção de locais de "casa" e "trabalho", o pacote também disponibiliza ferramentas para proteger a privacidade dos usuários, tornando a desanonimização dos dados de mobilidade mais difícil.

[1] https://github.com/edgarsoliveira1/MINICURSO-ERCEMAPI2023

MOBVIS [33]: O *MobVis* é uma biblioteca para visualização de métricas de mobilidade, com suporte para pré-processamento e extração de métricas. Embora seja uma biblioteca nova e não tenha documentação, é possível usar *docstrings* para entender melhor a ferramenta.

PTRAIL [12]: Desenvolvida para o pré-processamento de dados de trajetória, o *PTRAIL* oferece funcionalidades como filtragem de ruídos, remoção de *outliers*, interpolação e extração de *features*. O pacote também se destaca pela execução paralela e vetorizada dessas tarefas, aumentando a eficiência do processamento. Apesar de não oferecer ferramentas para análise de dados, o *PTRAIL* apresenta uma integração bem definida com outras bibliotecas, como GeoPandas[16].

MOBILKIT [39]: Diferenciando-se das outras bibliotecas livres listadas, o *Mobilkit* possui um enfoque especial na análise de mobilidade pós-desastres naturais, como terremotos e enchentes. Construído sobre o framework *Dask* [29], que tem como objetivo facilitar o processamento de grandes volumes de dados, o *Mobilkit* oferece recursos adicionais para análise de dados geo-temporais. No entanto, uma limitação desta biblioteca é a falta de recursos para proteger a privacidade dos usuários, o que é uma preocupação crescente no campo da mineração de dados.

TRAJA [32]: O *Traja* é uma biblioteca Python com foco na análise de trajetórias de animais. Além das ferramentas convencionais de mineração de trajetórias, como aquelas baseadas em redes neurais, a biblioteca oferece recursos específicos para este contexto, como a extração de trajetórias de vídeos que atendam a determinadas restrições.

MOVINGPANDAS [10]: É uma biblioteca de código aberto que oferece recursos para explorar e analisar dados de mobilidade. A biblioteca é uma extensão do Pandas, Geopandas [16] e HoloViz [35], e fornece estruturas de dados e funções para cálculo de velocidade e direção, extração de pontos de parada, divisão de trajetórias em sub-viagens, agregação, generalização e visualização estática e dinâmica dos dados de trajetória.

MOBIPY [19]: Construída em cima da biblioteca Pandas, *Mobipy* busca simplificar o cálculo de métricas e padrões de mobilidade de dados geo-temporais. Com uma sólida base de manipulação de dados na Mineração de Dados, possui recursos para competir com as outras bibliotecas aqui listadas em termos de desempenho. No entanto, é notável a limitação de métricas disponíveis, mesmo quando contamos com seus módulos de funções utilitárias internas.

PYMOVE [3]: O *PyMove* é uma biblioteca em Python para o processamento e visualização de trajetórias. Uma de suas principais características é a ênfase na flexibilidade e extensibilidade do código, permitindo que o usuário adicione funcionalidades conforme necessário. Para isso, o código segue padrões como as regras *PEP8* [40] e possibilita a alteração da estrutura de baixo nível, como a utilização de dados que não cabem em memória, utilizando a biblioteca *Dask* [29]. Essas características tornam o *PyMove* uma ferramenta interessante para a comunidade de análise de mobilidade que busca soluções personalizáveis e escaláveis para o processamento de grandes volumes de dados de trajetórias.

SCIKIT-MOBILITY [24]: É um software estatístico que fornece suporte para cientistas e profissionais em diversos aspectos da análise de mobilidade. Essa biblioteca é bas-

tante popular em comparação com outras na área de mineração de mobilidade. Além das funcionalidades básicas, o *Scikit-Mobility* oferece ferramentas para análise de riscos à privacidade dos usuários e a representação de fluxos, algo que não é encontrado em outros pacotes. Isso permite que os usuários possam lidar com questões críticas de segurança em suas análises de mobilidade. Outra característica notável do *Scikit-Mobility* é a sua capacidade de trabalhar com dados de mobilidade em grande escala, permitindo a análise de conjuntos de dados de milhões de pontos em tempo hábil.

TRACKTABLE [28]: Uma biblioteca que se destaca pela sua eficiência no processamento de grandes volumes de dados de trajetória, graças à implementação de suas principais estruturas de dados e algoritmos em C++. Além de recursos comuns em outras bibliotecas, como a visualização e análise de trajetórias, ela oferece recursos únicos, como a geração de mapas de calor e a detecção de quadrantes em trajetórias. Esses recursos podem ser úteis em diversas aplicações, desde o monitoramento de veículos em tempo real até a análise de movimentos de animais.

SMAFRAMEWORK [30]: O *SMAFramework* é uma biblioteca de análise de mobilidade urbana que foca na integração de múltiplos *datasets* heterogêneos. Possui ferramentas para gerenciamento e análise de dados, mas não possui documentação, o que torna seu uso mais desafiador em comparação com as demais bibliotecas.

MOCHA [34]: O *MOCHA* é uma ferramenta de pré-processamento de trajetórias que extrai métricas temporais, espaciais e sociais, além de classificar as trajetórias com base em suas distribuições de probabilidade. Também produz guias visuais para comparação de traços e modelos de mobilidade, útil para avaliar a qualidade de trajetórias reais ou sintéticas.

TEETOOL [8]: Com o objetivo de fornecer recursos para a análise de trajetórias em duas ou três dimensões, o *Teetool* oferece ferramentas para minerar dados e gerar predições. Além de métricas comuns, como distância e velocidade, também apresenta métricas específicas, como a região de confiança, úteis para a análise de trajetórias de objetos aéreos. Embora disponha de algumas ferramentas para outras etapas da mineração, como visualização, é necessário recorrer a outras bibliotecas para realizar todas as tarefas envolvidas na mineração de trajetórias.

3.3.2. Uma Comparação Qualitativa

Baseados nos módulos previamente apresentados e nas questões de pesquisas (QP) levantadas na seção 3.2.2, elaborou-se uma comparação qualitativa para verificar a abragência e a cobertura das bibliotecas nas tarefas de tratamento e análise de dados de mobilidade. Por meio dessa comparação, torna-se possível apresentar uma classificação para cada biblioteca, definindo qual desempenha melhor por cenário.

3.3.2.1. Quais bibliotecas escolher com base nos dados?

Todos os tipos de representação de mobilidade abordados neste estudo, GNSS, são suportados por todas as ferramentas apresentadas. No entanto, ao considerar o tipo de entidade conforme a Tabela 3.3, é perceptível que, ao lidar com fluxos, a escolha se restringe

a cinco bibliotecas. Nesse aspecto, é importante ressaltar que *MovingPandas*, *Scikit-Mobility* e *PyMove* oferecem um suporte mais abrangente para outras tarefas. Nesse sentido, com base na Tabela 3.3, as seguintes questões de pesquisas são respondidas: **QP1) Quais tipo(s) de dados de mobilidade (por exemplo, ponto(s) GPS, segmento(s)) estão(ão) disponível(is) como representação(ões) para análise nas bibliotecas?** e **QP2) Explora qual(is) tipo(s) de representação de mobilidade?**.

Table 3.3. Tipo de entidades móveis.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Trajetória	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+
Fluxo	-	-	+	-	-	-	-	+	-	+	+	-	+	-	-

3.3.2.2. Bibliotecas para tarefas de pré-processamento

Ao analisar os dados apresentados na Tabela 3.5, torna-se evidente a única biblioteca que abrange as outras cinco tarefas mencionadas é a *PyMove*. No entanto, considerando a possibilidade da utilização de várias bibliotecas, como *MovingPandas* e *Scikit-Mobility*, nas quais os dados são armazenados de maneira compatível, é factível que um desempenho semelhante seja alcançado. Portanto, é viável que a combinação dessas bibliotecas seja explorada para a obtenção de suporte quase completo para as atividades de pré-processamento. Nesse sentido, com base na Tabela 3.5, a seguinte questão de pesquisa é respondida: **QP3) Qual(is) técnica(s) de pré-processamento esta(ão) implementada(s)?**.

Table 3.5. Técnicas de pré-processamento.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Filtragem de ruídos	-	+	-	-	+	-	-	+	-	+	+	+	-	-	-
Detecção de pontos de parada	+	-	+	+	-	+	-	+	-	+	+	-	-	-	-
Compressão de trajetórias	-	-	+	-	-	+	+	-	-	+	+	-	-	-	-
Segmentação de trajetória	-	-	+	-	+	-	+	+	-	+	-	-	-	-	-
Map matching	-	-	-	-	-	-	-	-	-	+	-	-	-	-	-

3.3.2.3. Bibliotecas para gerenciar mobilidade

Uma conclusão que podemos derivar da Tabela 3.6 é que, ao lidar com um grande volume de dados, é viável empregar a ferramenta *PyMove* para a filtragem das tabelas específicas

desejadas. Posteriormente, caso seja necessário, pode-se recorrer a outras bibliotecas para realizar tarefas adicionais. Essa abordagem permite uma manipulação de dados de mobilidade eficaz e modular. Nesse sentido, com base na Tabela 3.6, a seguinte questão de pesquisa é respondida: **QP4) Qual(is) recurso(s) para gerenciamento de dados esta(ão) disponível(is)?**.

Table 3.6. Recursos para gerenciamento de dados.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Consultas em Intervalo	-	-	-	-	+	+	+	+	-	+	-	+	-	-	-
Consultas KNN	-	-	+	-	-	-	-	-	-	+	-	-	-	-	-
Similaridade de Trajetórias	-	-	-	-	-	-	-	-	+	+	-	-	-	-	-

3.3.2.4. Bibliotecas para lidar com incerteza e análise de privacidade

O exame da Tabela 3.7 revela que a escassez de métodos para lidar com incerteza é perceptível, uma vez que apenas 3 métodos oferecem ferramentas para a suavização da trajetória, a saber, *MovingPandas*, *Traja* e *PTRAIL*. Além disso, a análise de riscos para a privacidade é apoiada apenas pela biblioteca *Scikit-Mobility*, o que evidencia uma oportunidade para a implementação de novas abordagens. Nesse sentido, com base na Tabela 3.7, a seguinte questão de pesquisa é respondida: **QP5) Qual(is) método(s) para lidar com a incerteza esta(ão) implementado(s)?**.

Table 3.7. Métodos para lidar com incerteza.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Reduzindo a Incerteza	-	-	-	-	+	-	+	+	-	-	-	-	-	-	-
Análise de Privacidade	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-

3.3.2.5. Bibliotecas para gerar mobilidade

Na tarefa de geração de mobilidade sintética, conforme demonstrado na Tabela 3.8, observa-se que nenhum dos recursos disponíveis nas bibliotecas se equipara em abrangência aos oferecidos pela *Scikit-Mobility*. No entanto, em outras bibliotecas, estão disponíveis recursos de geração de trajetórias por meio de métodos mais simples, como o *Random Walk*. A ausência de métodos mais avançados revela uma carência de abordagens robustas para

a geração de mobilidade sintética. Este é um ponto que merece atenção e pode ser explorado como uma oportunidade para o desenvolvimento de novas técnicas e métodos nessa área. Nesse sentido, com base na Tabela 3.8, a seguinte questão de pesquisa é respondida: **QP6) É possível gerar dados de mobilidade?**.

Table 3.8. Métodos para gerar mobilidade.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Gerar Trajetória	-	+	-	-	-	-	+	-	-	-	+	+	-	-	+
Gerar Fluxo	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-
Mobility Diary	-	-	-	-	-	-	-	-	-	-	+	-	-	-	-

3.3.2.6. Bibliotecas para mineração de padrões de trajetórias

É possível que várias ferramentas sejam utilizadas em conjunto para obter uma gama mais completa de opções, como ilustrado na Tabela 3.9, combinando o uso do *Scikit-Mobility* com o *MovingPandas* ou *Traja*, que são compatíveis entre si. No entanto, não foi encontrada uma biblioteca, dentre as listadas, que disponha de recursos específicos para a mineração de padrões de movimentação conjunta (*Moving Together*), além da escassez de suporte para a mineração de padrões sequenciais e periódicos. Esses são aspectos que indicam possíveis áreas de desenvolvimento e aprimoramento nas bibliotecas existentes. Nesse sentido, com base na Tabela 3.9, a seguinte questão de pesquisa é respondida: **QP7) Qual(is) técnica(s) de mineração de padrões de trajetórias esta(ão) implementada(s)?**.

Table 3.9. Técnicas de minerações de padrões.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Moving Together	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Agrupamento de Trajetórias	-	-	+	-	-	+	+	+	+	+	+	+	+	-	-
Padrões Sequenciais	-	-	-	-	-	-	-	-	-	-	+	-	-	-	+
Padrões Periódicos	-	-	-	-	-	-	+	+	-	-	-	-	-	-	-

3.3.2.7. Bibliotecas para tarefa de classificação ou predição de trajetória

Conforme observado na Tabela 3.10, nenhuma das ferramentas listadas apresenta uma solução específica para a classificação. Isso indica que há uma lacuna nas bibliotecas

existentes em relação a esse aspecto particular da mineração de trajetórias. Seria interessante explorar o desenvolvimento de métodos ou a integração de outras bibliotecas para abordar essa necessidade.

Não obstante, apenas a biblioteca *Traja* oferece uma solução para o problema de predição de trajetória. Isso destaca a importância de explorar e implementar abordagens específicas para essa tarefa, considerando sua relevância na análise de dados de mobilidade. Nesse sentido, com base na Tabela 3.10, a seguinte questão de pesquisa é respondida: **QP8) Qual(is) técnica(s) para predição ou classificação de trajetórias esta(ão) implementada(s)?**.

Table 3.10. Técnicas para predição ou classificação de trajetórias

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Predição	-	-	-	-	-	-	+	-	-	-	-	-	-	-	-
Classificação	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

3.3.2.8. Bibliotecas para detecção de anomalias

Conforme pode ser observado na Tabela 3.11, outra lacuna é evidenciada no estado da arte das bibliotecas Python para mineração de dados de mobilidade. Nenhuma solução é apresentada para a detecção de eventos atípicos. Apenas a biblioteca *Tracktable* oferece uma solução para a detecção de trajetórias anômalas. Isso realça a necessidade de ferramentas adicionais e o desenvolvimento de métodos específicos para a detecção de anomalias em dados de mobilidade, com o propósito de explorar todo o potencial dessas informações e identificar padrões incomuns ou comportamentos fora do esperado. Nesse sentido, com base na Tabela 3.11, a seguinte questão de pesquisa é respondida: **QP9) Quais técnicas para detecção de anomalias estão implementadas?**.

Table 3.11. Técnicas para detecção de anomalias.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Detectando Trajetórias Atípicas	-	-	-	-	-	-	-	-	-	-	-	+	-	-	-
Detecção de Eventos Atípicos	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

3.3.2.9. Bibliotecas para visualização de mobilidade

Ao analisar os dados apresentados na Tabela 3.12, a ampla variedade de métodos disponíveis para a visualização de dados de mobilidade é evidente. É importante ressaltar que a biblioteca *MobVis* tem como foco exclusivo essa tarefa específica da mineração de dados de mobilidade. Portanto, a ausência de recursos para outras tarefas não diminui a efetividade de seu uso em conjunto com outras bibliotecas que abordam diferentes aspectos da análise de dados de mobilidade. A combinação dessas ferramentas pode oferecer uma abordagem completa e abrangente na visualização e exploração dos dados, proporcionando *insights* valiosos para a compreensão da mobilidade e a tomada de decisões informadas. Nesse sentido, com base na Tabela 3.12, a seguinte questão de pesquisa é respondida: **QP10) Existe suporte para visualização?**.

Table 3.12. Tipos de visualização disponível.

Bibliotecas	Trackintel	Yupi	Mobilipy	MobVis	PTRAIL	Mobilkit	Traja	MovingPandas	Mobipy	PyMove	Scikit-Mobility	Tracktable	SMAFramework	MOCHA	Teetool
Imagens	+	+	-	+	+	+	+	+	-	+	+	+	-	+	+
Gráficos	+	+	-	+	+	+	+	+	-	+	+	+	-	+	+
Mapas Interativos	-	-	+	+	+	-	-	+	-	+	+	-	+	-	-

É importante ressaltar que algumas bibliotecas delegam as funções de visualização para outras bibliotecas. Portanto, possibilitam o uso de todos os recursos disponíveis nestas bibliotecas de visualização, sendo necessário entendê-las para tirar total proveito de seus recursos. Isso adiciona uma camada de complexidade para o uso destes recursos.

3.4. Abordagem Prática para Tratamento e Análise de Mobilidade

Na abordagem prática, é apresentado um conjunto de dados reais, demonstrando a aplicação dos recursos já presentes nas bibliotecas comparadas. Essa aplicação deve abranger a execução de todas as tarefas relacionadas à necessidade de uso de cada biblioteca.

A metodologia proposta será aplicada da seguinte forma:

1. **Conjunto de dados**: A apresentação do conjunto de dados que será analisado revela que o *dataset* utilizado foi coletado em uma cidade urbana e contém informações de trajetórias registradas por dispositivos GPS em veículos de transporte público, táxis.

2. **Pré-processamento dos dados**: A realização do pré-processamento dos dados de trajetória será conduzida utilizando as bibliotecas *SciKit-Mobility* e *MovingPandas*. Isso incluirá a limpeza dos dados, a filtragem de ruídos, a detecção de pontos de parada, a segmentação de trajetória e a compressão.

3. **Análise exploratória**: A biblioteca *MovingPandas* será empregada para a realização de uma análise exploratória dos dados. Isso envolverá a visualização das

trajetórias, a identificação de padrões e a obtenção de *insights* iniciais sobre a mobilidade.

4. **Extração de características**: A biblioteca *PTRAIL* será utilizada para a extração de características relevantes dos dados de trajetória. Isso pode incluir informações como velocidade, distância percorrida, duração da viagem, entre outras.

5. **Predição de trajetória**: A modelagem dos dados de mobilidade será realizada por meio da biblioteca *Traja*. Isso pode envolver a aplicação de algoritmos de aprendizado de máquina ou técnicas de mineração de dados para a identificação de padrões e a predição dos dados de mobilidade.

Ao seguir essa metodologia, uma análise completa dos dados de mobilidade pode ser realizada, abrangendo desde o pré-processamento até a obtenção de conhecimentos e *insights* relevantes. Essa instância da metodologia é aplicada a um conjunto de dados real, possibilitando uma demonstração prática e aplicada dos conceitos discutidos nas seções anteriores.

3.4.1. Conjunto de dados

O uso do conjunto de dados *Taxi Service Trajectory*[2] é realizado para conduzir a análise. Este conjunto de dados descreve um ano completo das trajetórias de todos os 442 táxis que operam na cidade do Porto, em Portugal. Esses táxis operam por meio de uma central de despacho de táxis, utilizando terminais de dados móveis instalados nos veículos. As viagens são categorizadas em três categorias: (A) baseada na central de táxis, (B) baseada em ponto de táxi ou (C) não baseada na central de táxis. Para a primeira categoria, foi fornecido um ID anonimizado quando essa informação estava disponível por meio da ligação telefônica. As duas últimas categorias referem-se a serviços que foram solicitados diretamente aos motoristas de táxi em um (B) ponto de táxi ou em uma (C) rua aleatória. Este conjunto de dados oferece uma oportunidade valiosa para aplicar a metodologia discutida e obter *insights* significativos sobre a mobilidade na cidade do Porto.

3.4.2. Pré-processamento dos dados

LIMPEZA DOS DADOS

Antes de prosseguir para as etapas subsequentes, é fundamental realizar a limpeza e formatação dos dados. Para assegurar a integridade dos dados, foram excluídas todas as trajetórias em que a coluna 'MISSING_DATA' continha um valor diferente de '*False*'. Além disso, as colunas que continham valores ausentes ou apresentavam apenas um valor em todo o conjunto de dados também foram removidas. Após essa filtragem, uma amostra aleatória das trajetórias remanescentes foi selecionada, correspondendo a aproximadamente 0,05% do total. Dessa amostra, apenas as trajetórias com mais de 5 pontos foram mantidas. Por fim, houve uma modificação na representação das trajetórias: anteriormente, cada linha do conjunto de dados representava uma trajetória completa, mas agora cada linha corresponde a um ponto específico, compartilhando os mesmos valores para o restante da trajetória.

[2]https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i

FILTRAGEM DE RUÍDOS

A seguir, é apresentado um exemplo específico de uma trajetória que ilustra claramente a presença de ruídos ou valores discrepantes, nos quais dois pontos são repentinamente distanciados consideravelmente de seus vizinhos. Para lidar com esses casos, é necessário que a filtragem dos dados seja realizada, e esse problema é abordado por várias bibliotecas disponíveis. Nesta instância, a solução fornecida pela biblioteca *Scikit-Mobility* foi utilizada. Por meio dessa biblioteca, todos os pontos que excedem a velocidade máxima de 300 km/h foram removidos. O resultado, conforme ilustrado na Figura 3.4, é apresentado na comparação entre a trajetória original, representada pela linha tracejada vermelha, e a nova trajetória representada pela linha azul, na qual os pontos removidos são destacados em verde.

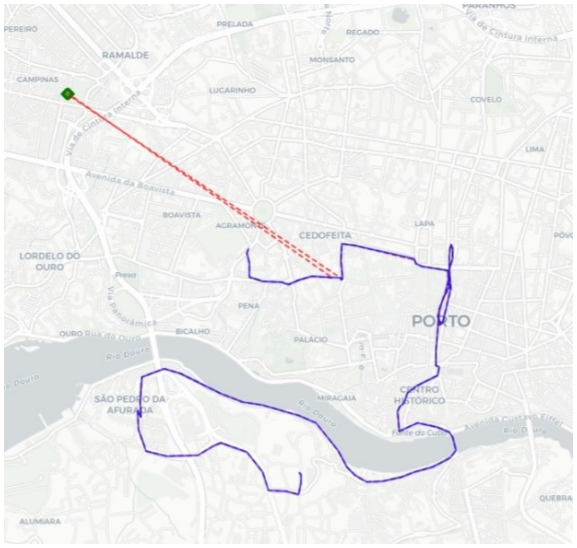

Figure 3.4. Exemplo de filtragem de ruídos

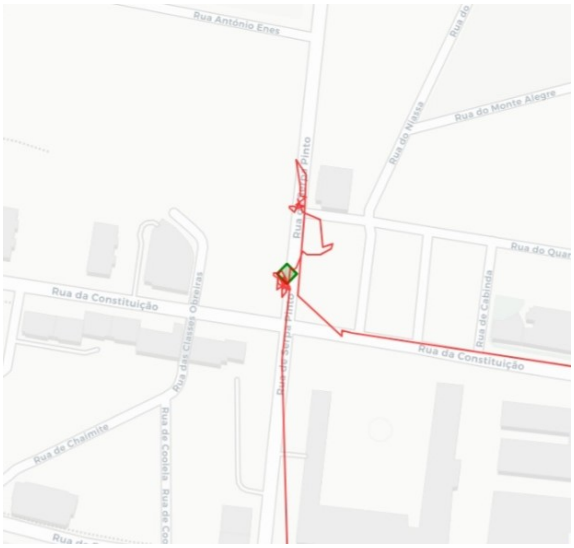

Figure 3.5. Exemplo de detecção de pontos de parada

DETECÇÃO DE PONTOS DE PARADA

Para ilustrar mais um exemplo, outra trajetória foi selecionada na qual um ponto de parada foi detectado, como pode ser visto na Figura 3.5. A detecção desse ponto de parada foi realizada por meio da funcionalidade fornecida pelo *Scikit-Mobility*. Nesse caso, os critérios utilizados para a classificação de um ponto como parada foram aqueles em que o objeto permaneceu por mais de 5 minutos a uma distância de 0.2 quilômetros. O ponto de parada detectado está destacado em verde na Figura 3.5.

COMPRESSÃO DE TRAJETÓRIA

Para exemplificar a compressão de uma trajetória, uma instância do *dataset* foi selecionada e o método de compressão fornecido pelo *Scikit-Mobility* foi aplicado. Esse método substitui conjuntos de pontos que estejam a uma distância de 0.2 quilômetros por um único ponto médio. A Figura 3.6 apresenta a trajetória original em azul, a nova trajetória comprimida em vermelho tracejado e destaca em verde os pontos que foram removidos durante o processo de compressão.

SEGMENTAÇÃO DE TRAJETÓRIA

Para realizar essa tarefa, os recursos da biblioteca *MovingPandas* são utilizados para seg-

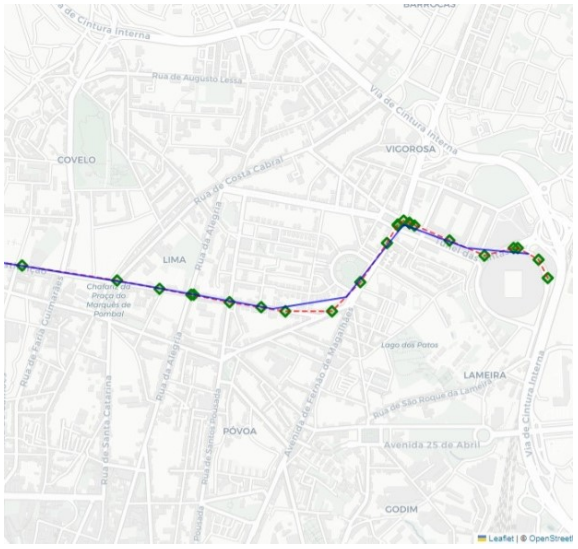

Figure 3.6. Exemplo de compressão de trajetória

Figure 3.7. Exemplo de segmentação de trajetória

mentar uma trajetória, exemplificando critérios que podem ser empregados para essa divisão. Na Figura 3.7, a trajetória original é dividida em três novos grupos de subtrajetórias. O primeiro grupo é criado com base em lacunas temporais, o segundo é formado pela detecção de pontos de parada, e o terceiro é resultado da análise das diferenças de velocidade.

3.4.3. Análise exploratória

Esta etapa busca aumentar o conhecimento sobre os dados disponíveis. Para exemplificá-la, serão respondidas algumas perguntas, listadas a seguir.

1. ***Qual foi a maior velocidade registrada?*** A maior velocidade registrada foi de 292.46 km/h, possivelmente originada de um *outlier* não filtrado.

2. ***Onde estão localizados os pontos mais significativos, os agrupamentos com maior densidade e qual é o fluxo entre eles?*** A Figura 3.11 torna visível que o centro da cidade tende a conter uma maior densidade de veículos.

3. ***Existe uma lacuna temporal nos dados de mobilidade?*** Os dados apresentam uma forma semelhante a uma distribuição sinusoidal ao visualizar a quantidade de pontos para cada dia, começando em '2013-07-01 05:55:41' e terminando em '2014-06-30 09:22:04'.

4. ***Qual a distribuição de viajem em cada mês e dia da semana?***

   Nas Figuras 3.8 e 3.9, é perceptível que há uma maior quantidade de viagens nas segundas e sextas-feiras. De forma semelhando, é notável o aumento de viagens nos meses próximos a junho e outubro.

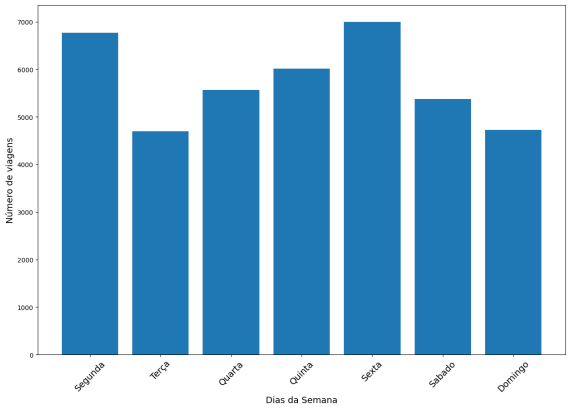

Figure 3.8. Número de viagens em cada dia da semana.

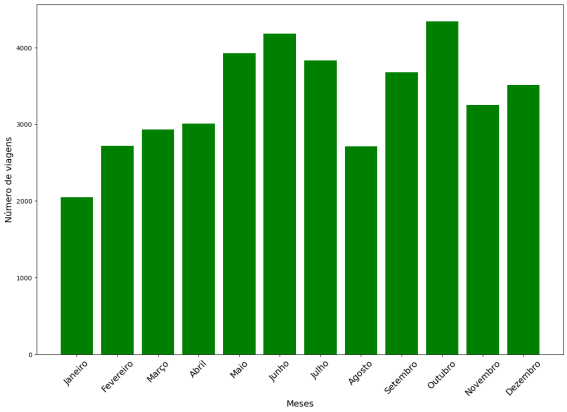

Figure 3.9. Número de viagens em cada mês.

3.4.4. Extração de características

Nesta fase, métodos de extração de características presentes na biblioteca *PTRAIL* foram empregados para enriquecer o conjunto de dados, adicionando informações temporais, como dia da semana e hora do dia, além de informações derivadas dos movimentos, como distância, velocidade e aceleração.

Por fim, modelos disponíveis na biblioteca *scikit-learn*[3] foram utilizados para exemplificar o impacto que a extração de características pode ter em tarefas relacionadas à análise de mobilidade, como a melhoria na acurácia da classificação dos tipos de chamadas (call_type). Essa abordagem demonstra como a utilização adequada das características extraídas pode contribuir significativamente para o aprimoramento dos resultados em aplicações de análise de mobilidade.

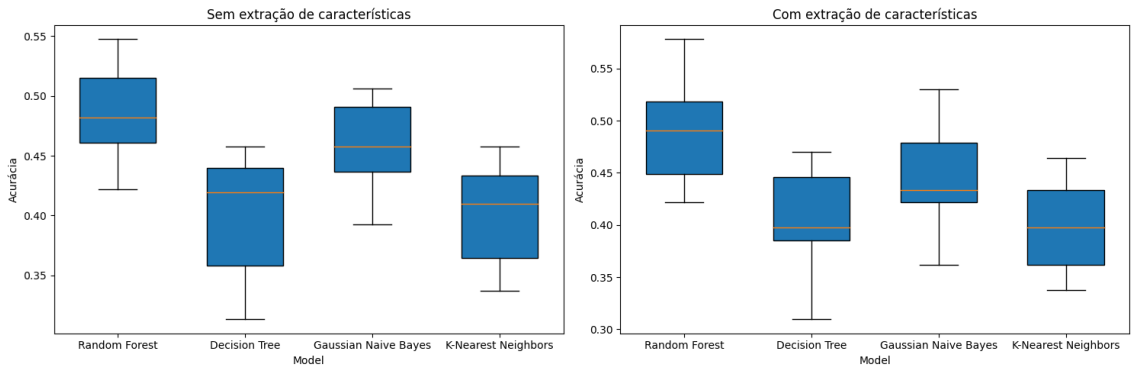

Figure 3.10. Impacto da extração de caraterísticas, na classificação de trajetórias.

3.4.5. Predição de trajetória

Com base na biblioteca Python *Traja*, realizou-se um experimento para prever os três próximos pontos de uma trajetória. Para isso, utilizamos o modelo *Long Short-Term Memory* (LSTM) disponibilizado pela biblioteca, o qual foi treinado e validado em sub-

[3]https://scikit-learn.org/stable/index.html

conjuntos distintos da amostra de dados. Após o processo de treinamento e validação, a predição com o menor Erro Quadrático Médio foi selecionada. Os resultados obtidos foram apresentados na Figura 3.12, demonstrando a acurácia da predição.

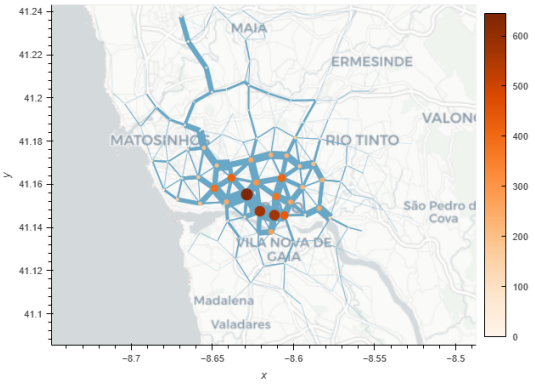

Figure 3.11. Número de pontos de densidade e fluxos.

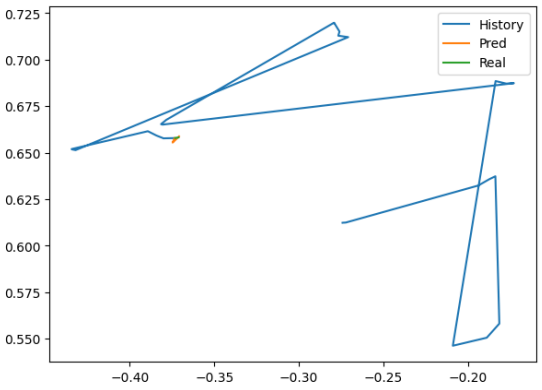

Figure 3.12. Predição de trajetória.

3.5. Conclusão

O objetivo principal deste estudo foi realizar uma comparação quantitativa entre bibliotecas Python que disponibilizam recursos para a análise de mobilidade. Posteriormente, as bibliotecas foram classificadas e associadas a fases específicas da aquisição de conhecimento de dados de trajetórias, seguindo um paradigma similar ao apresentado por [44].

A combinação dessas fases em uma metodologia ou *pipeline*, juntamente com os resultados obtidos na comparação, foi exemplificada por meio de um caso de uso. Esse caso de uso ilustrou resultados extraídos de uma base de dados real e disponibilizou o código-fonte[4] utilizado, demonstrando a relevância desses métodos na aquisição de conhecimento em dados de mobilidade.

Os *insights* gerados pela comparação das bibliotecas revelaram várias informações importantes. Foi identificada a quase ausência de suporte para tarefas de *map matching*, poucos métodos de geração de trajetória baseados em modelagem de conjuntos de dados e a inexistência de bibliotecas que permitam a mineração de movimentação conjunta (Moving Together). Além disso, observou-se a falta de suporte para a classificação de trajetórias e detecção de eventos atípicos dentro das trajetórias. Também foi notada uma escassez de soluções para análise de privacidade, predição e detecção de trajetórias atípicas.

Em resumo, a comparação, a metodologia e o caso de uso apresentados neste artigo proporcionam auxílio na tomada de decisões relacionadas ao uso dos recursos disponíveis nas bibliotecas Python para análise de mobilidade. Além disso, as lacunas identificadas revelam oportunidades para o avanço da pesquisa nessa área, incentivando o desenvolvimento de métodos mais abrangentes e aprimorados.

[4]https://github.com/edgarsoliveira1/MINICURSO-ERCEMAPI2023

References

[1] Osman Abul, Francesco Bonchi, and Mirco Nanni. “Never walk alone: Uncertainty for anonymity in moving objects databases”. In: *2008 IEEE 24th international conference on data engineering*. Ieee. 2008, pp. 376–385.

[2] Alastair R Beresford and Frank Stajano. “Location privacy in pervasive computing”. In: *IEEE Pervasive computing* 2.1 (2003), pp. 46–55.

[3] Maurício Cavalcante Bráz. “Implementação de algoritmos para análise de similaridade de trajetória na biblioteca PyMove”. In: *Monografia. Universidade Federal do Ceará* (2020).

[4] Clayson Celes, Azzedine Boukerche, and Antonio AF Loureiro. “From mobility traces to knowledge: Design guidance for intelligent vehicular networks”. In: *IEEE Network* 34.4 (2020), pp. 227–233.

[5] Clayson Celes et al. “Improving vanet simulation with calibrated vehicular mobility traces”. In: *IEEE Transactions on Mobile Computing* 16.12 (2017), pp. 3376–3389.

[6] Chi-Yin Chow, Mohamed F Mokbel, and Walid G Aref. “Casper* Query processing for location services without compromising privacy”. In: *ACM Transactions on Database Systems (TODS)* 34.4 (2009), pp. 1–48.

[7] Deyu Deng et al. “Spatial-temporal data science of COVID-19 data”. In: *2021 IEEE 15th International Conference on Big Data Science and Engineering (BigDataSE)*. IEEE. 2021, pp. 7–14.

[8] Willem Eerland et al. “Teetool–a probabilistic trajectory analysis tool”. In: *Journal of Open Research Software* 5.1 (2017).

[9] Gyozo Gidofalvi, Xuegang Huang, and Torben Bach Pedersen. “Privacy-preserving data mining on moving object trajectories”. In: *2007 International Conference on Mobile Data Management*. IEEE. 2007, pp. 60–68.

[10] Anita Graser and Melitta Dragaschnig. “Exploring movement data in notebook environments”. In: *IEEE VIS 2020 - MoVis*. 2020.

[11] Joachim Gudmundsson and Marc Van Kreveld. “Computing longest duration flocks in trajectory data”. In: *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*. 2006, pp. 35–42.

[12] Salman Haidri et al. “PTRAIL—A python package for parallel trajectory data preprocessing”. In: *SoftwareX* 19 (2022), p. 101176.

[13] Charles R. Harris et al. “Array programming with NumPy”. In: *Nature* 585 (2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

[14] Hoyoung Jeung, Heng Tao Shen, and Xiaofang Zhou. “Convoy queries in spatio-temporal databases”. In: *2008 IEEE 24th International Conference on Data Engineering*. IEEE. 2008, pp. 1457–1459.

[15] Hoyoung Jeung et al. “Discovery of convoys in trajectory databases”. In: *arXiv preprint arXiv:1002.0963* (2010).

[16] Kelsey Jordahl et al. “geopandas/geopandas: v0. 6.0”. In: *Zenodo* (2019).

[17] Ahmed Kharrat et al. “Clustering algorithm for network constraint trajectories”. In: *Headway in Spatial Data Handling*. Springer, 2008, pp. 631–647.

[18] Zhenhui Li et al. *Swarm: Mining relaxed temporal moving object clusters*. Tech. rep. ILLINOIS UNIV AT URBANA-CHAMPAIGN DEPT OF COMPUTER SCIENCE, 2010.

[19] Pedro HC Maia, Claudio EC Campelo, and Campina Grande–PB–Brazil. “Mobipy-A Python Library for Analyzing Mobility Patterns.” In: *GeoInfo*. 2020, pp. 141–152.

[20] Henry Martin et al. “Trackintel: An open-source Python library for human mobility analysis”. In: *Computers, Environment and Urban Systems* 101 (2023), p. 101938. DOI: 10.1016/j.compenvurbsys.2023.101938.

[21] Rui Moreno and Carina Oliveira. “Avaliação de Desempenho de Redes em Malha Sem Fio em Cenários de Cidades Inteligentes”. In: *Anais da X Escola Regional de Computação do Ceará, Maranhão e Piauí*. São Luís: SBC, 2022, pp. 109–118. DOI: 10.5753/ercemapi.2022.226045. URL: https://sol.sbc.org.br/index.php/ercemapi/article/view/21965.

[22] Brendan Morris and Mohan Trivedi. “Learning trajectory patterns by clustering: Experimental studies and comparative evaluation”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 312–319.

[23] Mehmet Ercan Nergiz, Maurizio Atzori, and Yucel Saygin. “Towards trajectory anonymization: a generalization-based approach”. In: *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*. 2008, pp. 52–61.

[24] Luca Pappalardo et al. “scikit-mobility: A Python Library for the Analysis, Generation, and Risk Assessment of Mobility Data”. In: *Journal of Statistical Software* 103.4 (2022), pp. 1–38. DOI: 10.18637/jss.v103.i04. URL: https://www.jstatsoft.org/index.php/jss/article/view/v103i04.

[25] Michal Pleskowicz. *Mobilipy-from raw GPS data to mobility data*. Tech. rep. 2022.

[26] Jeff Reback et al. “pandas-dev/pandas: Pandas 1.0. 5”. In: *Zenodo* (2020).

[27] A Reyes et al. “yupi: Generation, Tracking and Analysis of Trajectory data in Python”. In: *Environmental Modelling & Software* 163 (2023), p. 105679.

[28] Mark Daniel Rintoul. *ML for Trajectories: Bridging the Gap from Computer to Analyst with Tracktable.* Tech. rep. Sandia National Lab.(SNL-NM), Albuquerque, NM (USA), 2020.

[29] Matthew Rocklin et al. “Dask: Parallel computation with blocked algorithms and task scheduling”. In: *Proceedings of the 14th python in science conference*. Vol. 130. SciPy Austin, TX. 2015, p. 136.

[30] Diego O Rodrigues and Leandro Villas. “SMAFramework: Arcabouço para Integração de Dados Urbanos Cientes da Correlação Espaço-Temporal”. In: *Anais Estendidos do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. SBC. 2019, pp. 113–120.

[31] AL Sayeth Saabith, MMM Fareez, and T Vinothraj. “Python current trend applications-an overview”. In: *International Journal of Advance Engineering and Research Development* 6.10 (2019).

[32] Justin Shenk et al. “Traja: A Python toolbox for animal trajectory analysis”. In: *Journal of Open Source Software* 6.63 (2021), p. 3202.

[33] Lucas N Silva et al. “MobVis: A Framework for Analysis and Visualization of Mobility Traces”. In: *2022 IEEE Symposium on Computers and Communications (ISCC)*. IEEE. 2022, pp. 1–6.

[34] Fabrício R de Souza et al. “Mocha: A tool for mobility characterization”. In: *Proceedings of the 21st ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. 2018, pp. 281–288.

[35] Jean-Luc R Stevens, Philipp Rudiger, and James A Bednar. “HoloViews: Building complex visualizations easily for reproducible science”. In: *Proceedings of the 14th Python in Science Conference*. SciPy. 2015, pp. 61–69.

[36] Lu-An Tang et al. “A framework of traveling companion discovery on trajectory data streams”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 5.1 (2014), pp. 1–34.

[37] ZhengYu TangLu’an et al. “Discover of Traveling COmpanions from Streaming Trajectoris”. In: *ICDE, WashingtonDC, USA* (2012).

[38] Manolis Terrovitis and Nikos Mamoulis. “Privacy preservation in the publication of trajectories”. In: *The Ninth international conference on mobile data management (mdm 2008)*. IEEE. 2008, pp. 65–72.

[39] Enrico Ubaldi et al. “Mobilkit: A Python Toolkit for Urban Resilience and Disaster Risk Management Analytics using High Frequency Human Mobility Data”. In: *arXiv preprint arXiv:2107.14297* (2021).

[40] Guido Van Rossum, Barry Warsaw, and Nick Coghlan. “PEP 8–style guide for python code”. In: *Python. org* 1565 (2001), p. 28.

[41] Haitao Yuan and Guoliang Li. “A survey of traffic prediction: from spatio-temporal data to intelligent transportation”. In: *Data Science and Engineering* 6 (2021), pp. 63–85.

[42] Kai Zheng et al. “On discovery of gathering patterns from trajectories”. In: *2013 IEEE 29th international conference on data engineering (ICDE)*. IEEE. 2013, pp. 242–253.

[43] Kai Zheng et al. “Online discovery of gathering patterns over trajectories”. In: *IEEE Transactions on Knowledge and Data Engineering* 26.8 (2013), pp. 1974–1988.

[44] Yu Zheng. “Trajectory data mining: an overview”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 6.3 (2015), pp. 1–41.