INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO CEARÁ
IFCE CAMPUS ARACATI
COORDENADORIA DE CIÊNCIA DA COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

LEONARDO FREITAS DA SILVA

ARCA - Aplicativo de Recomendação de Caronas baseado em Algoritmos de similaridade de mobilidade de usuário

ARACATI-CE
2020

LEONARDO FREITAS DA SILVA

ARCA - APLICATIVO DE RECOMENDAÇÃO DE CARONAS BASEADO EM ALGORITMOS DE SIMILARIDADE DE MOBILIDADE DE USUÁRIOS

Trabalho de Conclusão de Curso (TCC) apresentado ao curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Aracati, como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Orientador (a): Prof. Dr. Reinaldo Bezerra Braga
Coorientador: Prof. Msc. Clayson Sandro Francisco de Sousa Celes

Aracati-CE
2020

Dados Internacionais de Catalogação na Publicação
Instituto Federal do Ceará - IFCE
Sistema de Bibliotecas - SIBI
Ficha catalográfica elaborada pelo SIBI/IFCE, com os dados fornecidos pelo(a) autor(a)

S586a Silva, Leonardo Freitas da Silva.
ARCA - Aplicativo de Recomendação de Caronas baseado em Algoritmos de similaridade de mobilidade de usuário / Leonardo Freitas da Silva Silva. - 2020.
62 f. : il. color.

Trabalho de Conclusão de Curso (graduação) - Instituto Federal do Ceará, Bacharelado em Ciência da Computação, Campus Aracati, 2020.
Orientação: Prof. Dr. Reinaldo Bezerra Braga.
Coorientação: Prof. Me. Clayson Sandro Francisco de Sousa Celes.

1. Compartilhamento de carona. 2. Algoritmos de similaridade. 3. Mobilidade urbana. I. Titulo.

LEONARDO FREITAS DA SILVA

ARCA - APLICATIVO DE RECOMENDAÇÃO DE CARONAS BASEADO EM ALGORITMOS DE SIMILARIDADE DE MOBILIDADE DE USUÁRIOS

Trabalho de Conclusão de Curso (TCC) apresentado ao curso de Bacharelado em Ciência da Computação do Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE - Campus Aracati, como requisito parcial para obtenção do Título de Bacharel em Ciência da Computação.

Aprovada em 11/03/2020

BANCA EXAMINADORA

Prof. Dr. Reinaldo Bezerra Braga (Orientador)
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE-Aracati

---

Prof. M.Sc. Clayson Sandro Francisco de Sousa Celes (Co-orientador)
Universidade Federal de Minas Gerais e Universidade de Ottawa

---

Profa. Dra. Carina Teixeira de Oliveira
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE-Aracati

---

Prof. M.Sc. Silas Santiago Lopes Pereira
Instituto Federal de Educação, Ciência e Tecnologia do Ceará - IFCE-Aracati

DEDICATÓRIA

Aos meus pais, Raimundo Edilson Maia da Silva e Maria de Lourdes de Freitas.
A todos os meus professores.
A todos os pesquisadores e estudantes que possam usufruir deste trabalho.

AGRADECIMENTOS

Agradeço a Deus, por ter me sustentado e por sempre estar comigo durante toda minha caminhada.

Aos meus pais, Raimundo Edilson e Maria de Lourdes, por todo amor, apoio, dedicação e pelo grande incentivo à conquista da minha formação. Também à minha irmã, Larissa Freitas, por todo o apoio e carinho que tem dado a mim. Eles foram o alicerce para que eu chegasse onde cheguei.

À minha namorada, Ingrid Emilly, por todo amor, carinho e compreensão durante toda essa caminhada. Além de sempre estar comigo, me ajudando e apoiando durante todo este trabalho e toda minha formação.

Ao meu professor e orientador, Reinaldo Braga, por todo auxílio e orientação ao longo deste trabalho, além de todos os ensinamentos e conselhos dados durante toda minha formação.

Ao meu coorientador, Clayson Celes, por todo auxílio e coorientação essenciais para a realização deste trabalho.

A todos os meus amigos, em especial, Rômulo Henrique, Igor Galdino, Roberta Alencar, Vinícius Nunes, Ruan Gondim, Renato Alves, Guilherme Oliveira e Edgar dos Santos, que me acompanharam durante todo o curso e que sempre estiveram juntos comigo, me ajudando durante toda minha formação. Amigos que levarei para a vida toda.

Aos meus professores, por todo o conhecimento e ensinamentos que me passaram, não só técnicos, mas também de vida.

A todos que me ajudaram com este trabalho, em especial Fernando Éricles e Lucas Sena.

À banca avaliadora.

RESUMO

Desde o surgimento do automóvel em 1885, nunca houve um aumento tão significativo de veículos automobilísticos como observado nos últimos anos. Esse aumento tem gerado diversos problemas nos grandes centros urbanos relacionados à mobilidade, tais como, congestionamentos e aumento da emissão de poluentes na atmosfera, além de estar diretamente relacionado ao crescimento do número de acidentes de trânsito. Consequentemente, estes problemas geram outros problemas, ligados à saúde pública, bem como problemas econômicos para as cidades, pois gera-se um alto custo para o tratamento e reabilitação de pacientes vítimas de acidentes de trânsito. Diante dessa realidade, várias estratégias vêm sendo propostas com a finalidade de resolver ou, pelo menos, amenizar tais problemáticas por meio da redução de veículos nos grandes centros urbanos. Dentre muitas estratégias utilizadas, o compartilhamento de caronas tem sido uma das alternativas mais apropriadas. Nesse contexto, este trabalho propõe, por meio da tecnologia da informação, o aplicativo ARCA, uma solução que utiliza estratégias de criação de grupos de caronas, baseada em algoritmos de similaridade de mobilidade de usuários. O ARCA é dividido em três partes: (*i*) o Módulo de Análise, responsável pela análise de mobilidade e pelo agrupamento dos usuários com trajetórias similares, utilizando algoritmos de aprendizado de máquina; (*ii*) a API, onde é feita a comunicação entre os dados gerados no Módulo de Análise e o Aplicativo Móvel; (*iii*) o Aplicativo Móvel ARCA, onde é realizada a interação com o usuário e a recomendação dos grupos de caronas, gerados de forma automática e transparente para o usuário. A principal contribuição deste trabalho está relacionada à construção destes três componentes do ARCA, que objetiva oferecer uma estratégia de redução do número de veículos circulantes nas ruas. Além disso, uma contribuição secundária está ligada à análise do comportamento de alguns dos algoritmos de similaridade e agrupamento disponíveis na literatura. Por fim, os resultados obtidos demonstraram que o ARCA é uma estratégia viável e eficiente.

Palavras-chaves: Compartilhamento de carona. Algoritmos de similaridade. Mobilidade urbana.

ABSTRACT

Since the appearance of the automobile in 1885, there has never been a more significant increase in motor vehicles than observed in recent years. This increase has generated several problems in large urban centers related to mobility, such as congestion and increased emission of pollutants into the atmosphere, in addition to being directly related to the growth in the number of traffic accidents. Consequently, these problems generate other problems, linked to public health, as well as economic problems for cities, as it generates a high cost for the treatment and rehabilitation of patients who are victims of traffic accidents. In view of this reality, several strategies have been proposed in order to resolve or, at least, alleviate such problems by reducing vehicles in large urban centers. Among many strategies used, sharing rides has been one of the most appropriate alternatives. In this context, this work proposes, through information technology, the ARCA application, a solution that uses strategies to create groups of hitchhikers, based on user mobility similarity algorithms. ARCA is divided into three parts: (*i*) the Analysis Module, responsible for the mobility analysis and for grouping users with similar trajectories, using machine learning algorithms; (*ii*) the API, where communication is made between the data generated in the Analysis Module and the Mobile Application; (*iii*) the ARCA Mobile Application, where interaction with the user and the recommendation of groups of hitchhikers are carried out, generated automatically and transparently for the user. The main contribution of this work is related to the construction of these three components of ARCA, which aims to offer a strategy to reduce the number of vehicles circulating on the streets. In addition, a secondary contribution is linked to the analysis of the behavior of some of the similarity and clustering algorithms available in the literature. Finally, the results obtained demonstrated that ARCA is a viable and efficient strategy.

Keywords: Sharing a ride. Similarity algorithms. Urban mobility.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
ACEA	*European Automobile Manufacturers Association*
AM	Aprendizado de Máquina
DSRPAI	*Dartmouth Summer Research Project on Artificial Intelligence*
DBSCAN	*Density-based Spatial Clustering of Applications with Noise*
IA	Inteligência Artificial
IPEA	Instituto de Pesquisa e Econômica Aplicada
MD	Mineração de Dados
OPTICS	*Ordering Points To Identify the Clustering Structure*
OMS	Organização Mundial da Saúde
PLN	Processamento de Linguagem Natural
UTC	*Coordinated Universal Time*

SUMÁRIO

1 INTRODUÇÃO

Desde o surgimento do automóvel em 1885, nunca houve um aumento tão significativo de veículos automobilísticos como observado nos últimos anos. Somente na união europeia, segundo dados publicados em junho de 2018 pela *European Automobile Manufacturers Association* (ACEA), a frota de veículos cresceu de 273,4 milhões de veículos em 2008 para 308,2 milhões em 2017, sendo 268 milhões de carros de passageiros e 40,2 milhões de carros comerciais (ACEA, 2017). Nos Estados Unidos, os dados não são diferentes. Em 2017, foram registrados aproximadamente 272 milhões de veículos, totalizando um aumento de cerca de 22 milhões de veículos, comparado ao ano de 2010, que registrou 250 milhões de veículos (STATISTA, 2019b). No Brasil, a frota circulante de carros foi de 29,4 milhões em 2009 para 44,8 milhões em 2018 (SINDIPEçAS, 2019). Situações semelhantes a essas são perceptíveis nos demais países no mundo.

Partindo de uma observação capitalista, econômica e industrial, o aumento na frota de veículos nas ruas e rodovias do mundo é um marco positivo. Esse crescimento tende a continuar, visto que, em 2017, a quantidade de veículos vendidos no mundo todo foi de 79 milhões, trazendo estimativas ainda maiores de crescimento para os anos posteriores (STATISTA, 2019a).

Entretanto, algumas problemáticas graves surgem com esse aumento excessivo de veículos nas ruas. Dentre estas problemáticas destacam-se: a emissão extrema de gases poluentes na atmosfera; aumento de fluxo nas ruas e estradas, trazendo, consequentemente, um aumento de engarrafamentos no trânsito das cidades; e a redução da qualidade de vida da população nos grandes centros urbanos. Além disso, o crescente aumento da frota mundial de veículos tem sido diretamente relacionado com o alto número de acidentes de trânsito com vítimas (SOUSA, 2018).

Segundo dados da Organização Mundial da Saúde (OMS), disponibilizados em seu Relatório Global de Situação da Segurança no Trânsito, lançado em dezembro de 2018, 1,35 milhão de pessoas morrem, por ano, em todo o mundo devido aos acidentes de trânsito. Para se ter um comparativo, o número de mortos na guerra do Vietnã foi de aproximadamente 1,2 milhão[1]. Ainda segundo a OMS, entre 20 e 50 milhões de pessoas sofrem acidentes não fatais, porém, parte destes acidentes acaba provocando algum tipo de sequela e, muitas vezes, incapacita o indivíduo. Outro problema que surge com o número de acidentes de trânsito está relacionado aos custos com o tratamento desses pacientes, tais como: custos com a investigação da causa;

[1] https://www.bbc.com/portuguese/noticias/2015/04/150430_vietna_guerra_fatos_pai

redução ou perda da produtividade, devido a inatividade do trabalhador, gerando assim um déficit na economia; e o custo para reabilitar o paciente. A OMS ainda afirma que 3% do PIB da maioria dos países são designados a estes custos (OPAS/OMS, 2019).

Como citado anteriormente, o grande número de veículos trafegando nas ruas causa o aumento no fluxo de trânsito das cidades, provocando um aumento de congestionamentos nas vias. Uma notícia publicada em Janeiro de 2019, na página oficial da revista Forbes, mostrou uma pesquisa feita por uma empresa holandesa, realizada a partir de uma análise de congestionamentos em 403 cidades de 56 países. Tal pesquisa afirmou que 75% dessas cidades relataram níveis de congestionamento maiores ou estáveis entre os anos de 2017 e 2018 (FORBES; MCCARTHY, 2019). Os congestionamentos são tão graves, que afetam até mesmo a economia de um centro urbano. Uma pesquisa realizada pelo economista Guilherme Vianna, da empresa Quanta Consultoria, em 2018, e publicada no site oficial de notícias G1, mostrou que em média, no Brasil, R$ 267 bilhões de reais são perdidos devido ao tempo consumido em congestionamentos, apenas durante o trajeto de casa para o trabalho. Esse valor representa 4% do Produto Interno Bruto (PIB) nacional (G1; RAMALHO, 2018).

Outro ponto importante a ser tratado neste contexto é o de sustentabilidade. Segundo (SOUZA, 2017), nos grandes centros urbanos, as emissões de gases poluentes por veículos automotores rodoviários se destacam. O autor de (SOUZA, 2017) ainda cita que, mesmo havendo uma diminuição de emissão de poluentes nos carros atuais e havendo ganhos devido ao avanço da tecnologia, variáveis como o aumento da frota de veículos e, consequentemente, os congestionamentos das vias, ainda causam problemas sociais, econômicos e ambientais expressivos. Um trabalho publicado pelo Instituto de Pesquisa e Econômica Aplicada (IPEA), em 2011, afirma que um ônibus transportando 70 pessoas equivale a cerca de 50 automóveis transportando um valor médio de 1,5 pessoas por veículo. Em outras palavras, apenas o fato de adotar transportes coletivos já reduz, consideravelmente, o número de veículos nas ruas e, por consequência, a poluição e o número de engarrafamentos (CARVALHO, 2011).

Diante dos dados levantados, observa-se uma gama de fatores prejudiciais e adversos nas cidades, relacionados ao número excessivo de veículos nas ruas, principalmente nos grandes centros urbanos, onde a concentração de veículos é muito maior que em cidades do interior. Todas as problemáticas expostas estão ligadas diretamente ao conceito de Mobilidade Urbana. Tal conceito vem sendo explorado nos últimos anos, pois é de suma importância a existência de planos governamentais e planos de ação e controle afim de melhorar os problemas que surgem dentro deste escopo. Os problemas relacionados a esse assunto são relevantes e um artigo publicado na página oficial da comissão europeia afirma que:

> “As cidades europeias enfrentam cada vez mais problemas causados pelo transporte e pelo trânsito. A questão de como melhorar a mobilidade e, ao mesmo tempo, reduzir o congestionamento, os acidentes e a poluição é um desafio comum a todas as grandes cidades da Europa" (COMMISSION, 2019).

Com todas essas informações expostas e diante das reais dificuldades encontradas nas cidades em sua mobilidade, se faz necessário definir estratégias de melhorias desses problemas. Uma das estratégias propostas por Sousa (2018) é a de adotar as facilidades trazidas pelas Tecnologias da Informação e Comunicação (TICs), para prover o compartilhamento de veículos:

> “É possível gerir a demanda por viagens de maneira eficaz, por exemplo, por meio do compartilhamento do veículo “carona solidária", que graças à inovação tecnológica na área da informação, das redes e da energia, é possível conectar pessoas a itinerários parecidos, viabilizando o uso compartilhado do automóvel, reduzindo o número de veículos nas vias e reproduzindo uma mobilidade mais eficiente e limpa, com mais pessoas/veículos transportada"(SOUSA, 2018).

Diversas aplicações comerciais já existem atualmente com o intuito de fornecer o compartilhamento de caronas entre pessoas. Dentre elas podemos citar a Uber (UBER, 2018), BlaBlaCar (BLABLACAR, 2019), AmigoExpress (AMIGOEXPRESS, 2019) e Waze Carpool (CARPOOL, 2019). Todas estas aplicações funcionam de forma semelhante, onde os usuários realizam de forma manual o cadastro e as buscas de caronas. Tais aplicações serão descritas com mais detalhes no Capítulo 3.

Desta forma, este trabalho visa, por meio da tecnologia da informação, propor o ARCA, um aplicativo que utiliza estratégias de criação de grupos de caronas baseados em algoritmos de similaridade de trajetórias. O objetivo principal do ARCA é o de reduzir o fluxo de veículos nas cidades, promovendo a redução dos problemas de mobilidade urbana já apresentados.

Resumidamente, o funcionamento do ARCA é dividido em etapas. Primeiramente, o algoritmo do ARCA, de forma autônoma e transparente, realiza a coleta e análise de similaridade dos dados dos usuários. Em seguida, o ARCA sugere grupos de caronas para os usuários que possuem perfis semelhantes de mobilidade e de locais de partida e chegada. Isso é possível devido a utilização das informações geradas pela análise de similaridade do ARCA. Portanto, o ARCA surge como uma solução de apoio para reduzir o número de carros nas ruas. Isso é possível porque os usuários podem compartilhar seus veículos em escalas diferentes com os membros de seus grupos e, consequentemente, contribuir para a diminuição de congestionamentos e a emissão de poluentes, gerando um ambiente mais agradável nas cidades.

1.1 Objetivos

1.1.1 Objetivo geral

Desenvolver um aplicativo de caronas, no qual os usuários poderão, de forma transparente, receber recomendações de grupos de carona de forma automática, criados através de algoritmos de agrupamento, com base na similaridade de suas rotas veiculares diárias nos grandes centros urbanos.

1.1.2 Objetivos específicos

- Investigar algoritmos de similaridade de mobilidade e agrupamento para análise de rotas de usuários;
- Analisar a aplicabilidade dos algoritmos de similaridade selecionados considerando um contexto de compartilhamento de caronas;
- Projetar e implementar um módulo de análise de mobilidade e recomendação automática;
- Desenvolver uma API para integrar módulo de análise de mobilidade e recomendação automática com o aplicativo móvel ARCA;
- Planejar e implementar o aplicativo móvel ARCA que mostrará e recomendará os grupos formados de acordo com a mobilidade dos usuários.

1.2 Organização do Trabalho

Os próximos capítulos deste trabalho estão organizados da seguinte forma. No Capítulo 2, está exposto todo o contexto fundamental para a compreensão deste trabalho, contendo desde a compreensão dos conceitos de mobilidade urbana e grupos de carona, bem como o estado da arte da Inteligência Artificial, desde o seu contexto histórico até os conceitos técnicos relacionados. No Capítulo de 3, estão expostos os trabalhos relacionados ao ARCA, além do comparativo destes trabalhos em relação ao ARCA. No Capítulo 4 é apresentado o ARCA de forma detalhada, demonstrando todo o processo de construção e suas funcionalidades. A Avaliação Experimental, disposta do Capítulo 5, demonstra os resultados obtidos da avaliação experimental realizada, buscando avaliar os algoritmos de agrupamento utilizados nesta proposta. O Capítulo 6 expõe as conclusões deste trabalho assim como os trabalhos futuros a serem aplicados ao ARCA.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, os conceitos relacionados à proposta apresentada são descritos. O objetivo principal desta seção é expor e facilitar o entendimento teórico dos temas relevantes ao trabalho.

2.1 Mobilidade Urbana

No Brasil, o parágrafo II, do artigo 4, da Lei 12.587 de 2012 define mobilidade urbana como sendo a condição em que se realizam os deslocamentos de pessoas e cargas no espaço urbano. Já de acordo com (BARCELOS; SILVA, 2018), mobilidade urbana é a condição para a realização de deslocamentos para transportes e carga. Outra definição exposta por (SANTOS, 2019), é de que “mobilidade urbana é um atributo das cidades, relativo ao deslocamento de pessoas e bens no espaço urbano, utilizando, para isto, veículos, vias e toda a infraestrutura urbana".

Segundo (CRUZ, 2016), nos últimos anos, a mobilidade urbana tem chamado bastante atenção de pesquisadores e entidades governamentais, principalmente por estar relacionada aos muitos problemas cotidianos, tais como congestionamento, danos ao meio ambiente, prejuízos econômicos e comprometimento da saúde da população. Nesse sentido, diversas estratégias têm sido tomadas para contornar esses problemas. Por exemplo, o governo brasileiro, por meio de um projeto de lei, está buscando estimular o uso de transporte público a fim de diminuir a quantidade de carros nas ruas. Além disso, como discutido em (CRUZ, 2016), uma outra estratégia de melhorar a mobilidade urbana é por meio do compartilhamento de veículos particulares (caronas), visto que a maioria dos veículos são ocupados somente pelos condutores.

2.2 Grupos de Carona

A utilização de grupos de carona é algo que já vem sendo estudado há algum tempo e tem se mostrado uma ótima estratégia de redução de congestionamentos nas vias (CRUZ, 2016). Existe um conceito já implantado, semelhante ao compartilhamento de caronas, que é o chamado *ridesharing*. Esse conceito consiste de um modo de transporte onde os passageiros utilizam o mesmo veículo de forma compartilhada para realizar uma determinada viagem, de forma que as pessoas dividem os custos dessa viagem (FURUHATA et al., 2013). O autor de (FURUHATA et al., 2013), afirma que a coordenação de viagens compartilhadas ainda é feita de forma informal

e desorganizada. Por exemplo, por meio de redes sociais ou grupos de usuários em aplicativos de mensagem instantânea. Com base nisso, existe a necessidade de construir aplicativos que automatizem esse processo. Ao mesmo tempo, esses aplicativos devem fornecer segurança e confiabilidade às pessoas envolvidas.

Existem diversos aplicativos que realizam a integração de pessoas em grupos de caronas. Os autores de (GHOSEIRI; HAGHAN; HAMEDI, 2011) citam a existência de 33 plataformas relevantes com esta finalidade. O autor de (CRUZ, 2016) cita 9 aplicativos que fornecem serviços de *ridesharing* e realiza uma breve comparação entre elas. Desta forma, observa-se a viabilidade tecnológica da criação desses aplicativos de carona, bem como uma tendência na aceitação desses aplicativos por parte de usuários.

2.3 Inteligência Artificial

Inteligência Artificial (IA), segundo (NING; YAN, 2019), é uma ciência tecnológica que busca simular, expandir e estender a teoria, técnicas e aplicações da inteligência humana. Entretanto, a IA não chega a ser de fato a inteligência humana nem tampouco superior a ela. Desta forma, a IA possui o intuito de compreender como a inteligência funciona e assim poder aplicar e reproduzir esse conhecimento em máquinas. Outra definição dada por (HOSEA; HARIKRISHNAN; RAJKUMAR, 2011) é a de que IA é a capacidade de um dispositivo executar funções similares à inteligência humana, como a capacidade de raciocinar sobre determinada situação e manipulação de conhecimentos sobre fatos e heurísticas.

Apesar de a Inteligência Artificial parecer ser algo totalmente novo, o conceito de IA já existe desde a década de 50, mais especificamente no ano de 1956, quando dois cientistas da computação da universidade de Stanford, Marvin Minsky e John McCarthy, organizaram o *Dartmouth Summer Research Project on Artificial Intelligence (DSRPAI)*, em português: Projeto de Pesquisa de Verão de Dartmouth sobre Inteligência Artificial, realizado no Dartmouth College em Nova Hampshire. Esse evento foi considerado o marco inicial da Inteligência Artificial (HAENLEIN; KAPLAN, 2019).

A partir daí, grandes investimentos foram feitos na área, o que possibilitou um grande triunfo da IA. Um dos primeiros exemplos de sucesso envolvendo Inteligência Artificial foi o ELIZA, um *software* desenvolvido entre 1964 e 1966, por Joseph Weizenbaum. O *software* consistia em uma ferramenta de Processamento de Linguagem Natural (PLN) capaz de simular uma conversa humana (HAENLEIN; KAPLAN, 2019).

Nos últimos anos, são observadas diversas aplicações de Inteligência Artificial, nas mais diversas áreas de conhecimento. Segundo (LU et al., 2017), isso se deu

principalmente graças ao aumento da capacidade de processamento dos computadores e do acúmulo de grandes volumes de dados.

Dentre suas diversas aplicações, (HOSEA; HARIKRISHNAN; RAJKUMAR, 2011) cita algumas das áreas onde a IA está presente:

- **Jogos eletrônicos** - o uso de IA nos jogos eletrônicos possibilita que o computador seja um jogador nos jogos de xadrez ou damas, afim de competir com humanos.
- **Sistemas especialistas** - nesse tipo de sistema, a IA possibilita a tomada de decisão em situações reais, baseada em dados previamente coletados e analisados pelos algoritmos. Por exemplo, o uso de sistemas especialistas para diagnosticar doenças através dos sintomas.
- **Linguagem natural** - possibilita que o computador compreenda linguagens humanas.
- **Redes neurais** - a IA, nesse tipo de sistema, tenta simular conexões físicas presentes no cérebro de animais.
- **Robótica** - possibilita computadores a reagir a estímulos sensoriais, visuais e auditivos.

2.4 Aprendizado de máquina (AM)

Uma das áreas de estudo da inteligência artificial é a de Aprendizado de Máquina (AM). Conforme (GOLDSCHMIDT, 2010), tal técnica tem o objetivo de criar e desenvolver algoritmos e técnicas computacionais, que tornem computadores capazes de aprender automaticamente a partir de dados existentes prévios, tornando-se capazes de aperfeiçoar seu desempenho em tarefas a eles destinadas. Ainda segundo (GOLDSCHMIDT, 2010), a grande maioria dos algoritmos de aprendizado de máquina, utiliza como base métodos indutivos de aprendizado, onde a indução é a capacidade de inferir logicamente sobre um conjunto de dados existente e retirar conclusões genéricas desses dados.

De acordo com os autores de (RASCHKA; MIRJALILI, 2019), o aprendizado de máquina é dividido em duas técnicas: aprendizado supervisionado e aprendizado não supervisionado. Cada método possui objetivos distintos em aplicações diversas.

No aprendizado supervisionado, tem-se como objetivo, realizar previsões sobre dados futuros ainda desconhecidos, com base em dados prévios adquiridos. Os

dados utilizados nesse mecanismo de aprendizado são conhecidos como dados rotulados, ou seja, dados onde o conjunto de amostras são conhecidas. Nessa técnica, o algoritmo de AM é treinado previamente e, a partir do aprendizado construído deste treinamento, pode inferir sobre novos dados desconhecidos. Dois subconjuntos dessa técnica de aprendizado são:

- **Classificação** - técnica onde o algoritmo classifica um novo dado de entrada atribuindo um valor discreto a esse novo dado baseado amostras observadas anteriormente.
- **Regressão** - técnica onde o algoritmo atribui como resultado de saída, um valor contínuo para um dado de entrada desconhecido.

Ainda segundo (RASCHKA; MIRJALILI, 2019), aprendizado não supervisionado tem como principal característica a utilização de dados não rotulados ou com estruturas não conhecidas. Assim, através dessa técnica é possível extrair informações sem o conhecimento do dado prévio. Uma das principais técnicas usadas nesse tipo de AM é o *Clustering* ou Agrupamento. A técnica de *Clustering* utiliza uma análise baseada em similaridade de dados, ou seja, dados que compartilham determinado grau de similaridade são inseridos em um mesmo grupo *(cluster)*, onde cada grupo possui características distintas dos demais.

2.5 Mineração de dados (MD)

Mineração de dados (MD), segundo (AMARAL, 2016), é uma ciência de dados que tem como principal objetivo obter informações e conhecimento de grandes conjuntos de dados, utilizando técnicas de aprendizagem de máquina. Ainda segundo (AMARAL, 2016), onde houver dado pode haver processo de mineração. Esse processo tem diversos campos de aplicação como, medicina, educação, marketing, reconhecimento de fala, reconhecimento de fraudes, sistemas de recomendação, dentre outros.

Como a mineração de dados está diretamente ligada ao aprendizado de máquina, esta também é dividida em duas categorias, semelhantes às de AM, a descoberta supervisionada e não supervisionada, como afirma os autores (GOMES; PIMENTA; SCHNEIDER, 2019). Assim como no aprendizado de máquina, a descoberta supervisionada cria modelos computacionais de dados, baseado em objetos pré-treinados. Esse tipo de abordagem geralmente tem como objetivo a predição de dados. Ainda segundo (GOMES; PIMENTA; SCHNEIDER, 2019), descoberta não supervisionada consiste na técnica de categorizar objetos sem o prévio conhecimento de quantas classes de dados existem no conjunto de dados.

2.6 Algoritmos de Similaridade e Agrupamento *(Clustering)*

Algoritmos de similaridade são técnicas capazes de analisar e obter informações contidas em uma base de dados. Essas técnicas realizam agrupamentos através da análise de similaridade, onde todos os objetos contidos em um grupo possuem características semelhantes (MARATHE et al., 2017). A análise de similaridade é usada por algoritmos não supervisionados de agrupamento como citado por (RASCHKA; MIRJALILI, 2019).

De forma geral, os algoritmos de agrupamento, segundo (SAXENA et al., 2017), podem ser divididos em dois grandes grupos: hierárquicos e de particionamento. Os algoritmos hierárquicos são subdivididos em mais duas categorias, sendo elas: aglomerativos e divisivos. Nos algoritmos hierárquicos aglomerativos utiliza-se uma abordagem conhecida como *bottom-up* ou seja, de baixo para cima, onde a construção dos grupos começa a partir de um único objeto e depois mescla esse grupo em grupos maiores, finalizando o processo quando todos os objetos estiverem cada um em um único grupo, ou atenderem as especificidades determinadas de parada. Os algoritmos hierárquicos divisivos seguem uma abordagem *top-down*, ou seja, de cima para baixo. Nessa abordagem um grupo contendo todos os objetos é dividido em grupos menores, até que cada objeto forme um único grupo ou atenda as condições de parada determinadas. Um exemplo de algoritmo dessa categoria é o HDBSCAN, utilizado nesse trabalho.

Ainda segundo (SAXENA et al., 2017), o segundo grupo de algoritmos de agrupamento são os algoritmos de particionamento. Esse grupo é dividido em outros subgrupos. Dentre eles, estão os algoritmos baseados em distância e os algoritmos baseados em densidade. Essas categorias de agrupamento particionado funcionam de forma oposta aos hierárquicos. Nesse tipo de abordagem, cada objeto é distribuído em um número K de grupos, e tal distribuição é feita na maioria das vezes utilizando distância euclidiana como função para definir em qual grupo um objeto irá pertencer. Entretanto, existem outras métricas de cálculo de distância que também são utilizadas, como, distância de Manhattan, distância de Minkowski, dentre outras[1]. Dentre os diversos algoritmos que fazem parte dessa categoria, podemos citar o K-means, DBSCAN e o OPTICS, que foram utilizados neste trabalho.

2.6.1 K-means

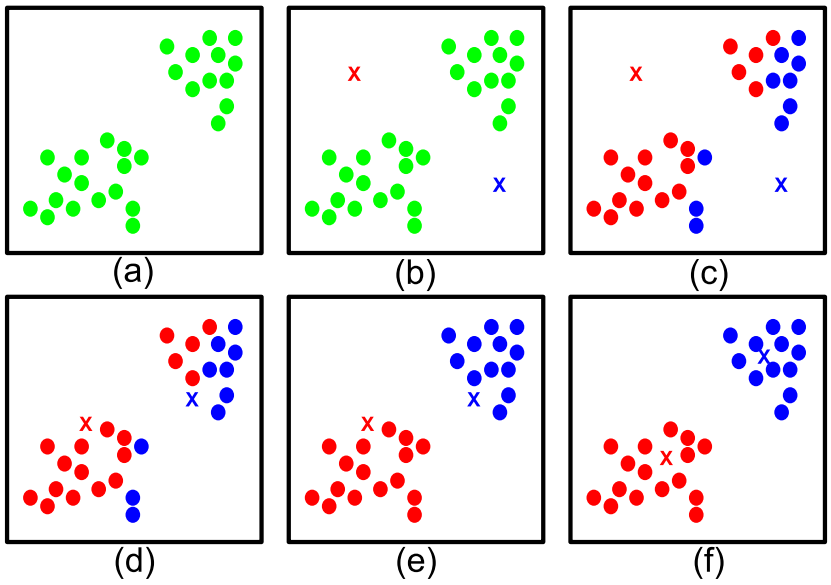
K-means é um dos algoritmos de agrupamento mais conhecidos. Sua metodologia se baseia na distância euclidiana entre os pontos. O funcionamento do K-means

[1] https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.DistanceMetric.html

consiste em um número *k* de grupos de entrada que são distribuídos aleatoriamente no conjunto de dados. Esses *k* pontos iniciais são denominados **centroides**, que são pontos centrais de cada grupo. A partir daí, utilizando-se dos cálculos de distância entre os pontos em relação aos centroides, cada ponto será atribuído a um grupo. Em seguida, o algoritmo recalcula as posições dos centroides. A operação se repete até que não haja mais movimentação dos centroides (SAXENA et al., 2017), (ARORA; VARSHNEY et al., 2016).

A Figura 1, demonstra os passos descritos acima das etapas do K-means, onde: *(a)* mostra o conjunto de dados original; *(b)* representa a atribuição dos centroides iniciais, representados com um **X**; *(c), (d)* e *(e)* demonstram a redistribuição dos centroides; e *(f)* mostra o estado final do agrupamento, quando não houver mais alteração no posicionamento dos centroides. [2]

Figura 1 – Funcionamento do algoritmo K-means



Fonte: Adaptada de (STANFORD, 2013)

Como citado, o K-means utiliza distância euclidiana para calcular a similaridade entre os objetos. Tal função encontra a distância mínima entre dois pontos e, a partir dessa distância, é determinado em qual grupo um objeto irá se encaixar. A Equação 2.1 apresenta a fórmula matemática para a obtenção da distância euclidiana, onde *q* e *p* são os pontos a serem calculados.

$$DE = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}, \tag{2.1}$$

[2] https://stanford.edu/cpiech/cs221/handouts/kmeans.html

Assim, quanto mais próximo de zero for a distância entre os pontos, mais similares são os objetos analisados (SEIDEL et al., 2008).

2.6.2 DBSCAN

DBSCAN, acrônimo de *Density-based Spatial Clustering of Applications with Noise*, traduzido para o português: Agrupamento Espacial Baseado em Densidade de Aplicações com Ruído, é um algoritmo de agrupamento baseado em densidade de dados (RASCHKA; MIRJALILI, 2019). O conceito por trás do DBSCAN é o de agrupamento de dados, onde a atribuição dos objetos em um grupo, se baseia na densidade em que esses objetos se encontram, a partir de parâmetros pré-estabelecidos. Diferentemente do K-means, o DBSCAN possui a capacidade de agrupar os objetos arbitrariamente. Em outras palavras, não existe a necessidade de definir um número de grupos a serem gerados, como afirma (HOU; GAO; LI, 2016). Segundo (WANG et al., 2015), isso leva o DBSCAN a possuir uma vantagem em relação ao K-means, além disso, o DBSCAN tem a capacidade de detectar ruídos nos objetos de análise, ou seja, objetos que não fazem parte de nenhum grupo.

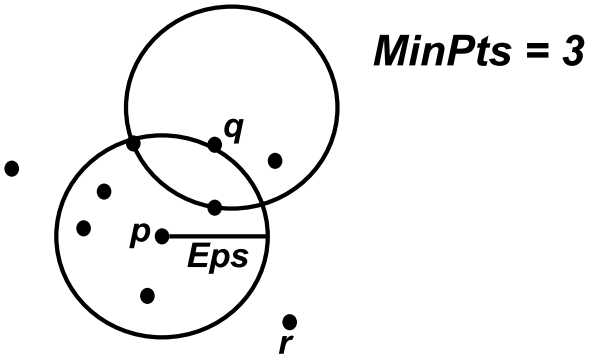
Como entrada, o algoritmo DBSCAN possui dois parâmetros, o raio *(Eps)* e a concentração (ou quantidade mínima de pontos) *(Min_pts)*. O raio determina a distância em que determinado objeto deve estar do ponto central de um grupo. Assim, um ponto que estiver a uma distância menor que o raio estabelecido é considerado um vizinho. Já a concentração é a quantidade mínima de pontos dentro do raio. Ou seja, um grupo de raio *Eps* deve possuir um número mínimo de vizinhos *Min_pts* para considerá-los do mesmo grupo (OLIVEIRA; SILVA; SILVA, 2019), (LADEIRA et al., 2019). Um objeto que não é um ponto central, mas está acessível a um ponto central, é denominado ponto de borda. Caso este objeto não esteja acessível a nenhum ponto central, é considerado um ponto de desvio, ou seja, um ponto ruidoso (HOU; GAO; LI, 2016).

A Figura 2 demonstra o método de funcionamento do DBSCAN descrito, onde *Eps* é o raio, *p* é o ponto central, *q* é o ponto de borda, *r* é um ponto de ruído.

2.6.3 OPTICS

OPTICS (*Ordering Points To Identify the Clustering Structure*), assim como o DBSCAN, é um algoritmo de agrupamento baseado em densidade. O algoritmo é considerado um aprimoramento do DBSCAN (GAO; YU, 2017). O OPTICS utiliza as mesmas definições de análise do DBSCAN, recebendo como parâmetro um valor mínimo de pontos, chamado de concentração de pontos em um grupo *(Min_pts)*. No

Figura 2 – Representação do método agrupamento do DBSCAN



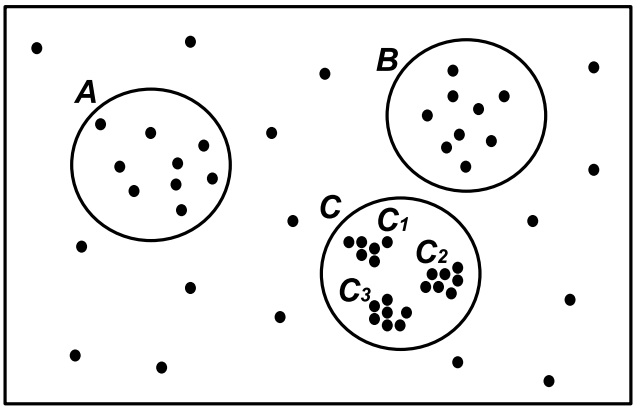
Fonte: Elaborado pelo autor

entanto, o algoritmo trabalha com um valor máximo de distância *(Max_eps)*, o que faz com que o algoritmo analise números de raios *(Eps)* infinitos até o limiar máximo de raio determinado. Essa forma de trabalhar permite que o OPTICS realize o chamado "aninhamento" de grupos, que são grupos de alta densidade a serem relacionados e se estão contidos em grupos de menor densidade (SANTOS, 2018).

A forma com que o OPTICS trabalha na correlação dos objetos permite que bases de dados com densidades variadas possam ser agrupadas, o que não era possível utilizando o DBSCAN, já que o DBSCAN utiliza um valor fixo padrão de raio para todo o conjunto de dados, como explica (MACEDO, 2018). A Figura 3 demonstra um conjunto de dados com densidades variadas, agrupados pelo OPTICS, onde, através do DBSCAN, não é possível identificar os grupos A, B, C_1, C_2 *e* C_3.

Além das vantagens sobre o algoritmo DBSCAN, os autores de (SANTOS, 2018) citam outra vantagem do algoritmo OPTICS em relação aos algoritmos hierárquicos de agrupamento, em bases de dados com densidades variadas. Os algoritmos hierárquicos possuem algumas limitações como sensibilidade à ruídos, o que é comum em base de dados reais. Além disso, a interpretação dos dendogramas gerados pelas hierarquias desse tipo de algoritmo, em base dados com muitos objetos, é de difícil compreensão.

Figura 3 – Conjunto de dados com densidades variadas identificadas pelo algoritmo OPTICS



Fonte: Adaptada de (ANKERST et al., 1999)

2.6.4 HDBSCAN

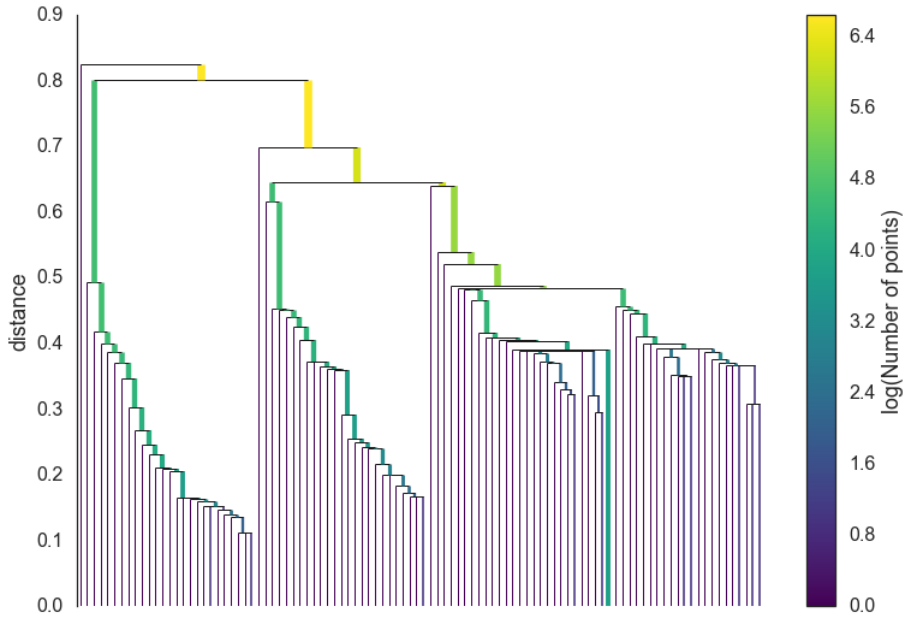
Density-Based Clustering Based on Hierarchical Density (HDBSCAN) é um algoritmo de agrupamento que mescla conceitos de algoritmos hierárquicos com de densidade, desenvolvido por (CAMPELLO; MOULAVI; SANDER, 2013). O HDBSCAN é uma extensão do algoritmo DBSCAN, porém com a possibilidade de criar hierarquias de grupos em função de suas densidades (CAMPELLO; MOULAVI; SANDER, 2013).

O algoritmo HDBSCAN utiliza apenas um parâmetro de entrada, que é o número mínimo de pontos *(Min_pts)* para determinar os grupos. Em seguida, ele monta hierarquias desses grupos, que podem ser verificadas através de dendogramas. Essa característica de criação de hierarquias permite que se possam ser consultados grupos em diversos níveis de densidade. A Figura 4 mostra um exemplo de um denograma gerado pelo HDBSCAN. Devido a essa característica, o HDBSCAN é muitas vezes descrito como um aperfeiçoamento do OPTICS (SIMÕES, 2019).

2.7 Avaliação de algoritmos de agrupamento

Avaliar os resultados de um algoritmo de agrupamento é uma forma de se obter uma análise sobre sua eficiência em determinado problema, ou avaliar diferentes parâmetros para a resolução de um problema específico de acordo com (RODRÍGUEZ et al., 2018). Uma vez que as técnicas de agrupamento são métodos não supervisionados de análise, os autores de (KRONBAUER; FONTOURA; WINCK, 2016) afirmam que análises de agrupamento não podem utilizar medidas de erro de estimativas em seus dados, como acontece com técnicas supervisionadas de aprendizado de máquina.

Figura 4 – Exemplo de dendograma gerado pelo HDBSCAN



Fonte: (MCINNES; HEALY; ASTELS, 2016)

Segundo (ANTONESCU; MOAYYED; BASAGNI, 2019), as métricas de avaliação de algoritmos de agrupamento são divididas em três categorias: avaliação externa, avaliação interna e avaliação relativa. Elas são determinadas de acordo com as informações disponíveis para o processo de validação. Ainda segundo (ANTONESCU; MOAYYED; BASAGNI, 2019), métricas de avaliação externas são utilizadas quando existe um conjunto de partições conhecidas, ou seja, quando existe um agrupamento definido como verdadeiro. Assim, essa forma de validação compara um agrupamento feito pelo algoritmo e o agrupamento verdadeiro. Por outro lado, os métodos internos de avaliação não necessitam de um conjunto de grupos verdadeiro para comparação. A análise feita nesse tipo de validação utiliza apenas do conjunto de dados, através de medições de compactação e separação dos grupos. Por fim, as técnicas relativas de avaliação fazem comparações entre diferentes parâmetros ou entre conjuntos de dados diferentes, em um mesmo algoritmo de agrupamento. Desta forma, para esta solução, técnicas internas de avaliação são mais adequadas, pois apenas o conjunto de dados é conhecido para realização das avaliações.

Dentre os métodos de avaliação de agrupamento, foi utilizado neste trabalho o Índice de Davies-Bouldin (DAVIES; BOULDIN, 1979). A análise feita por essa técnica utiliza a soma de dispersão dos grupos e a distância entre esses grupos para determinar um valor de referência de avaliação. Tal valor é um número que tem valor mínimo igual a 0. Desta forma, quanto menor for o valor retornado pelo índice, melhor é o resultado do agrupamento gerado por determinado algoritmo (CAMARGO; SANTOS, 2019).

As Equações 2.2 e 2.3 demonstram as fórmulas de análise do Índice de Davies-Bouldin, onde em 2.2, R_{ij} é a medida de separação entre os grupos que determina a similaridade entre eles, *s* é a distância média entre cada ponto i e j. O denominador d_{ij} é a distância entre os centroides de cada grupo. Por fim, o valor do índice é definido pela Equação 2.3 onde R_{ij} é o valor calculado pelo Equação 2.2 (DAVIES; BOULDIN, 1979).

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{2.2}$$

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij} \tag{2.3}$$

O método de execução do cálculo do Índice de Davies-Bouldin está disponível através da biblioteca *scikit-learn*[3]. Como parâmetros de entrada o método requer o conjunto de dados a ser analisado e o conjunto de grupos resultantes do algoritmo de agrupamento. Ao final da execução do método é retornado o valor do índice.

[3] https://scikit-learn.org/stable/modules/clustering.html#davies-bouldin-index

3 TRABALHOS RELACIONADOS

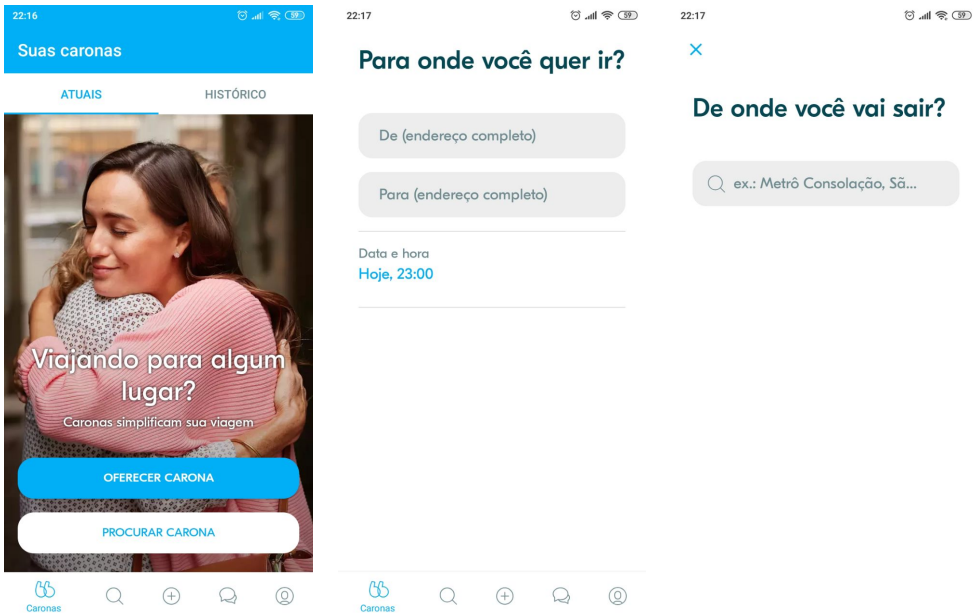
Como é possível observar, o conceito de mobilidade já vem sendo estudado e trabalhado nos últimos anos (EKMAN et al., 2008). Com isso, diversas aplicações vêm sendo criadas com o intuito de promover caronas e facilitar viagens. O autor de (CRUZ, 2016) cita diversos aplicativos de compartilhamento de viagens, dentre eles os principais citados são o Uber, lyft, Carticipate, Tripda, BlaBlacar, e Carma. O autor de (CRUZ, 2016) ainda afirma que estas aplicações são consideradas aplicações de caronas em tempo real, pois funcionam como um táxi, não havendo a necessidade de relação de amizade entre os usuários nem que os usuários queiram dar caronas, mesmo que elas possam ser usadas para este fim.

Uber é um aplicativo de táxi no qual os usuários podem solicitar a outros usuários motoristas corridas de táxi para seus respectivos destinos. No entanto, em 2018, a empresa disponibilizou uma nova opção para seus usuários, chamada de Uber Juntos, posteriormente substituída por UberPool. Essa funcionalidade permite que seus usuários possam compartilhar viagens com outros usuários que estejam indo em direções semelhantes. Assim, os passageiros dividem o veículo solicitado na aplicação e, desta forma, diminuem os custos com a viagem, já que os passageiros compartilham os mesmos trajetos (UBER, 2018).

BlaBlaCar é uma plataforma online de caronas de longas distâncias, no qual seus usuários compartilham seus destinos e veículos, facilitando viagens para pessoas que não tem a possibilidade de viajar em veículos próprios ou transportes públicos. Além da versão *Web*, o BlaBlaCar possui também a versão *mobile*. Dessa forma, um usuário que necessita de uma viagem realiza uma busca ou solicitação de ida a um destino na plataforma. Em paralelo, outro usuário fornece vagas em seus veículos para um destino. Caso os destinos destes usuários sejam iguais ou tenham trajetos iguais, estes usuários poderão trocar uma carona (BLABLACAR, 2019). A Figura 5 demonstra algumas telas do BlaBlaCar, sendo elas a tela inicial do aplicativo (a), a tela de busca de uma carona (b) e a tela de oferecimento de uma carona (c).

Outra plataforma semelhante ao BlaBlaCar é o AmigoExpress, uma ferramenta criada pela empresa canadense Kangaride, que tem como finalidade fornecer serviços de caronas para motoristas e passageiros que desejam dividir custos com combustível em viagens de média ou longa distâncias. Seu funcionamento consiste no usuário motorista postar uma viagem que deseja realizar, informando os custos com o combustível. Desta forma os usuários passageiros que se interessarem pelo anúncio irão solicitar uma ou mais vagas com um determinado valor, combinando um ponto de encontro com o motorista do anúncio. Por fim, os passageiros pagam o va-

Figura 5 – Telas do aplicativo BlaBlaCar

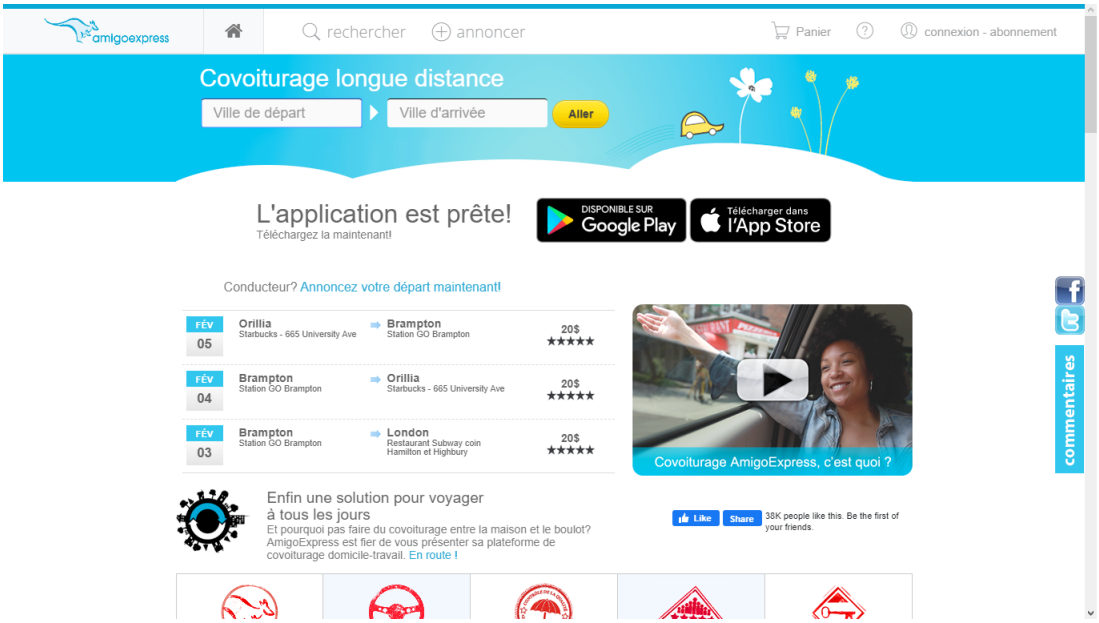


(a) Tela inicial BlaBlaCar (b) Busca por uma carona (c) Oferecer uma carona

Fonte: Adaptada de (BLABLACAR, 2019)

lor combinado dos custos do anúncio na viagem. O AmigoExpress está disponível apenas no Quebec, Canadá e Estados Unidos (AMIGOEXPRESS, 2019). A Figura 6 mostra a tela da versão *Web* do AmigoExpress.

Figura 6 – Tela inicial AmigoExpress

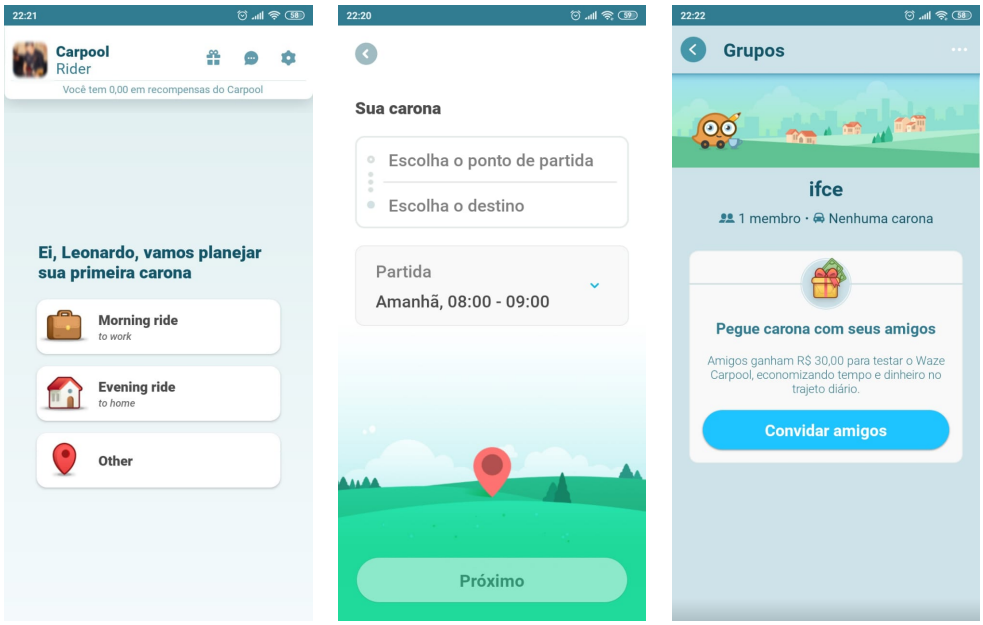


Fonte: Adaptada de (AMIGOEXPRESS, 2019)

A empresa de otimização de trânsito Waze também disponibiliza um serviço de caronas, nomeado de Waze Carpool. Ele funciona da mesma maneira dos apli-

cativos citados anteriormente, no qual motoristas e passageiros dividem custos compartilhados de seus itinerários. O Waze Carpool está disponível atualmente no Brasil, Israel, México e EUA (CARPOOL, 2019). Na Figura 7 podemos ver algumas das tela do aplicativo. Em (a) temos a tela inicial do aplicativo, em (b) a tela de busca por uma carona e em (c) a tela de criação de grupo de forma manual.

Figura 7 – Telas do aplicativo Waze Carpool



(a) Tela inicial do Waze Carpool (b) Busca por carona (c) Criar um grupo

Fonte: Adaptada de (CARPOOL, 2019)

Além dos aplicativos comerciais citados anteriormente, alguns trabalhos focados no desenvolvimento de análises inteligentes de rotas também foram desenvolvidos. Uma proposta desenvolvida por (CRUZ, 2016) é o GO!Caronas, um aplicativo de caronas que é uma extensão de um sistema chamado GO!. Tal sistema consiste em uma rede social que permite o compartilhamento de caronas. Entretanto, diferente das aplicações citadas anteriormente, GO!Caronas implementa um algoritmo *ridematching*, com o intuito de fornecer compartilhamentos em tempo real. Além disso, ele permite a possibilidade de criação de grupos, onde os membros compartilham os mesmos locais de trabalho ou estudo, porém, de forma manual, pelo usuário.

Os autores de (LI et al., 2008) criaram um algoritmo de similaridade baseado em mineração de usuários, através de históricos de localização. Para isso, eles utilizaram um arcabouço chamado *Hierarchical Similarity Measurement Graph* (HGSM) em português, Gráfico Hierárquico de Medição de Similaridade, para prover a criação dos modelos dos históricos de localização de pessoas, com isso os dados coletados eram agrupados através da similaridade encontrada entre eles.

Além desses, são encontrados outros aplicativos semelhantes, como, por exemplo, o Zumpy (ZUMPY, 2019) ou Wunder Carpool (WUNDER, 2019), dentre outros. Uma característica forte e comum da maioria dessas aplicações é a forma ativa de participação dos usuários, pois as buscas e os registros são realizados de forma manual. Ou seja, os usuários precisam realizar as buscas por veículos compartilhados. Ao mesmo tempo, os usuários também devem compartilhar as suas rotas de forma manual.

Observando detalhadamente todos esses aplicativos citados anteriormente, pode-se notar que todos eles necessitam da intervenção dos usuários para buscar, planejar e combinar uma carona. Portanto, devido a forma manual de realizar o fornecimento de caronas desses aplicativos, muitas caronas podem ser perdidas. Isso se dá ao fato de que, mesmo havendo várias opções de caronas, os usuários podem não ter tempo de buscar e analisar detalhadamente por uma viagem, para ver se tal carona se encaixa na sua necessidade. Como consequência disso, uma carona, por mais que esteja disponível, usuários não conseguirão encontrá-la. Desta forma, um aplicativo que faça recomendações de caronas de forma automática e transparente se faz necessário para a resolução desta limitação das aplicações descritas.

A Tabela 1 demonstra o comparativo entre as ferramentas descritas anteriormente. Nela podemos observar que o ARCA se destaca em relação às demais ferramentas em relação as funcionalidades observadas e propostas nesse trabalho: a transparência para o usuário, que é necessidade de intervenção direta do usuário para a realização das buscas por carona; a realização de análise inteligente de rotas; a criação de grupos de carona de forma automática; e a realização de recomendação automática de caronas.

Tabela 1 – Comparativo dos Trabalhos Relacionados

	Transparente para o usuário	Análise inteligente de rotas	Grupos de carona automáticos	Recomendação automática
Uber	-	-	-	-
BlaBlaCar	-	-	-	-
AmigoExpress	-	-	-	-
Waze Carpool	-	-	-	-
Cruz, 2016	-	Sim	-	-
LI et al., 2008	-	Sim	-	-
ARCA	Sim	Sim	Sim	Sim

Na primeira coluna da Tabela 1 está o comparativo em relação à transparência para o usuário. Vê-se que todos os trabalhos relacionado necessitam da intervenção

direta dos usuários para que possam encontrar rotas semelhantes. Já o ARCA essa intervenção direta não é necessária, visto que o ARCA realiza suas funcionalidades de forma transparente ao usuário.

A segunda comparação é em relação à análise inteligente de rotas. Alguns dos trabalhos relacionados funcionam como um buscador de rotas, ou seja, o usuário informa ao aplicativo um destino que queira ir e o aplicativo retorna todas as rotas de caronas com aquele destino. Os trabalhos, ARCA, GO!Caronas e Li et al., 2008, realizam de forma inteligente as análises de rotas aos usuários através da utilização de algoritmos de similaridade.

A terceira comparação realizada é em relação aos grupos de carona automáticos. Todos os trabalhos, com exceção do ARCA, realizam essa funcionalidade de forma manual, no qual o usuário cria seus grupos de carona manualmente. Já o ARCA realiza a criação de grupos de carona de forma automático, sem haver a necessidade do usuário realizar esta ação. Por fim, é analisado a funcionalidade recomendação automática. Observa-se que nenhum dos trabalhos realizam essa funcionalidade, com exceção do ARCA, que, de forma automática, faz a recomendação de grupos de carona logo após o usuário acessar o aplicativo.

4 ARCA

Diante dos diversos problemas de mobilidade urbana citados no Capítulo 1 e os conceitos expostos nos capítulos anteriores, foi realizado o desenvolvimento do ARCA, **A**plicativo de **R**ecomendação de **C**aronas baseado em **A**lgoritmos de similaridade de mobilidade de usuário. O ARCA, através de técnicas de aprendizado de máquina aplicadas em mineração de dados, realiza, de forma automática, a detecção de similaridade de trajetórias diárias de usuários. Através desses dados de similaridade, executa o agrupamento dos usuários e, por fim, recomenda os grupos de carona gerados de forma transparente e automática, sem a necessidade de intervenção do usuário.

4.1 Base de Dados

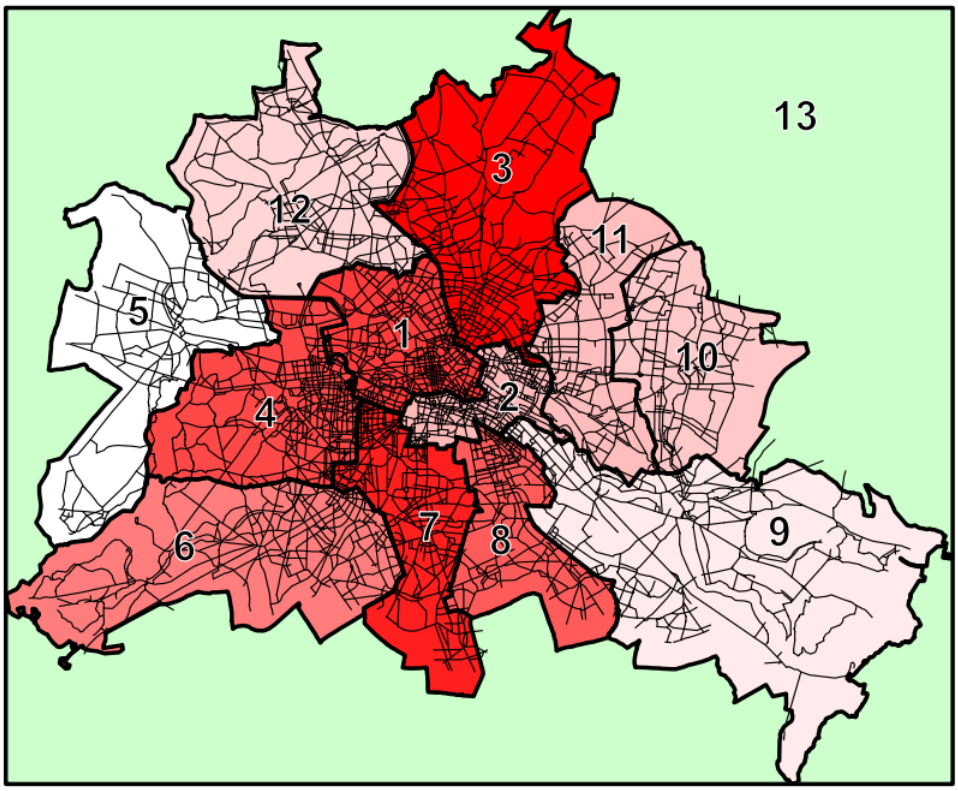
Para a realização dos experimentos propostos pelo ARCA, utiliza-se, como ambiente real, o BerlinMOD [1], uma base de dados criada com a finalidade de ser uma referência para gerenciamento de banco de dados espaço-temporais. Bancos de dados espaço-temporais, são conjunto de dados modelados para o armazenamento de informações geográficas no decorrer do tempo (NORONHA; FERREIRA; QUEIROZ, 2017). O BerlinMOD possui dados de rotas baseados em condições reais de deslocamento de veículos, dispostos nas ruas da cidade de Berlim, na Alemanha, no ano de 2007.

A Figura 8 demonstra o mapa das regiões de Berlim onde foram gerados os trajetos. Os dados do BerlinMOD são gerados a partir de um *script* executado no Secondo [2], que é um sistema de gerenciamento de banco de dados voltados para a criação de diversos modelos de dados, desenvolvido na FernUniversität em Hagen, na Alemanha. Vale ressaltar que todas as rotas dos veículos feitas pelo BerlinMOD seguem os limites de velocidade estabelecidos das vias reais das ruas de Berlim.

A base de dados do BerlinMOD está disponibilizada em diversos arquivos com tamanhos e quantidade de dados diferentes, adaptados para qualquer tipo de aplicação. O conjunto de dados escolhido para compor a proposta do ARCA consiste em um arquivo CSV com dados espaço-temporais, referentes a 2 dias de viagens (entre os dias 28 e 29 de maio de 2007) de 141 veículos.

[1] http://dna.fernuni-hagen.de/secondo/BerlinMOD/BerlinMOD.html
[2] http://dna.fernuni-hagen.de/Secondo.html/

Figura 8 – Regiões onde foram gerados os dados do BerlinMOD



Fonte: (DÜNTGEN; BEHR; GÜTING, 2008)

4.1.1 Estrutura da Base de Dados (BerlinMOD)

O conjunto de dados utilizado para a realização dos experimentos com o ARCA está disponível no arquivo **trips.csv**, que contém, para cada objeto, um total de 8 parâmetros, sendo eles:

- **Moid** - Tipo (*int*) - Identificador do automóvel do usuário referente a viagem.
- **Tripid** - Tipo (*int*) - Identificador da viagem.
- **Tstart** - Tipo (*date*) - Data e horário de início de cada viagem.
- **Tend** - Tipo (*date*) - Data e horário de término de cada viagem.
- **Xstart** - Tipo (*float*) - Valor da coordenada geográfica X de início da viagem.
- **Ystart** - Tipo (*float*) - Valor da coordenada geográfica Y de início da viagem.
- **Xend** - Tipo (*float*) - Valor da coordenada geográfica X de fim da viagem.
- **Yend** - Tipo (*float*) - Valor da coordenada geográfica Y de fim da viagem.

É importante ressaltar que cada viagem é dividida em micro viagens dentro do *dataset*. Isso é positivo para a análise de similaridade, pois permite que as viagens sejam analisadas por segmentos.

4.2 Limpeza e normalização dos dados

Após a concessão da base de dados, uma análise feita no conteúdo desses dados possibilitou a detecção de algumas inconsistências. Essas inconsistências são objetos com campos vazios ou incompletos, ou com padrões de dados diferentes. Tais inconsistências foram eliminadas do conjunto de dados, sem haver perdas significativas. Além disso, durante os experimentos, houve a necessidade de transformar os valores das colunas **Tstart** e **Tend** em outro padrão de representação. Essa transformação foi necessária devido aos dados no formato *date*, de tais colunas, não serem aceitos como parâmetro pelos algoritmos de agrupamento, utilizados no experimento. Para a resolução desse problema, foi realizada a conversão dos dados dessas colunas para o formato *Unix Timestamp*. Em seguida esses dados foram adicionados em duas colunas dentro do conjunto de dados original, denominadas **TstartTimestamp** e **TendTimestamp**, que correspondem aos valores convertidos do tempo de início da viagem e tempo de fim da viagem respectivamente. Alguns trabalhos como em (MONTEIRO; SILVA; MURTA, 2017), (SILVA, 2018), (FERREIRA et al., 2017) apresentam o formato *Unix Timestamp* como representação de dados temporal.

O *Unix Timestamp*, também conhecido como *Unix time*, é uma forma de representação de dados temporais, em que cada valor de data e hora é representado na forma de um número único. Basicamente, o *Unix Timestamp* conta os segundos passados desde o dia 01/01/1970 às 00:00:00 do Tempo Universal Coordenado, em inglês, *Coordinated Universal Time* (UTC) (PANDEY; MCBRIDE; EUSTICE, 2011). Esta data foi a data inicial de contagem do *Unix Timestamp*. Desta forma, a cada hora que passa o valor aumenta em 3600 unidades de tempo do *Unix Timestamp*, que equivale a quantidade de segundos contidos em uma hora. Por exemplo, a data 01/01/2019 às 12:00:00 UTC convertida para *Unix Timestamp* corresponde a 1546344000 e a mesma data 1 segundo à frente corresponde a 1546344001.

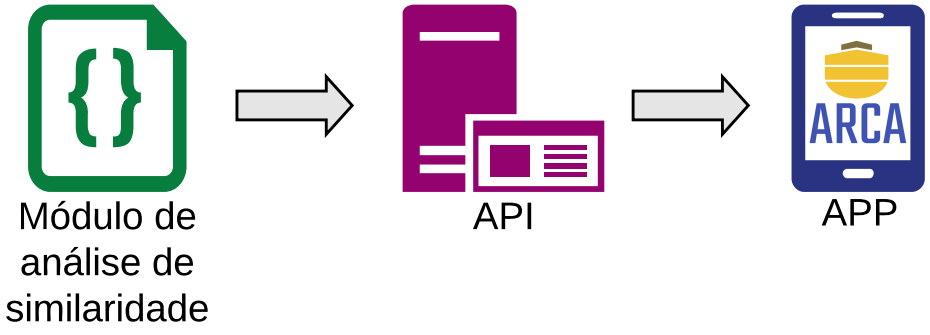
Todo o processo de limpeza e normalização foi realizado utilizando a biblioteca Pandas[3], uma biblioteca *open source* de alto nível, voltada para análise e manipulação de dados. A conversão dos valores de data para o formato *Unix Timestamp* foi feita utilizando métodos disponíveis nas bibliotecas **time**[4] e **datetime**[5] da linguagem Python.

[3] https://pandas.pydata.org/about/index.html
[4] https://docs.python.org/3/library/time.html
[5] https://docs.python.org/3/library/datetime.html

4.3 Estrutura de funcionamento do ARCA

A estrutura do ARCA está dividida em três partes, sendo elas, o módulo de análise de similaridade, a API e o aplicativo móvel. No módulo de análise de similaridade ocorre o processamento da base de dados em busca de usuários com trajetos de viagens similares. Nesta etapa, os dados coletados, já normalizados e limpos, são analisados e classificados em grupos os usuários contendo características similares. Após a etapa de agrupamento, realizada no módulo de análise de similaridade, os grupos formados são consumidos pela API. A API tem a função de direcionar a cada usuário os respectivos grupos de carona gerados pelo módulo de análise de similaridade. Por fim, o aplicativo móvel recebe os dados coletados pela API e recomenda ao usuário de forma automática. A Figura 9 demonstra a organização da estrutura do ARCA.

Figura 9 – Estrutura de funcionamento do ARCA.



Fonte: Elaborada pelo autor

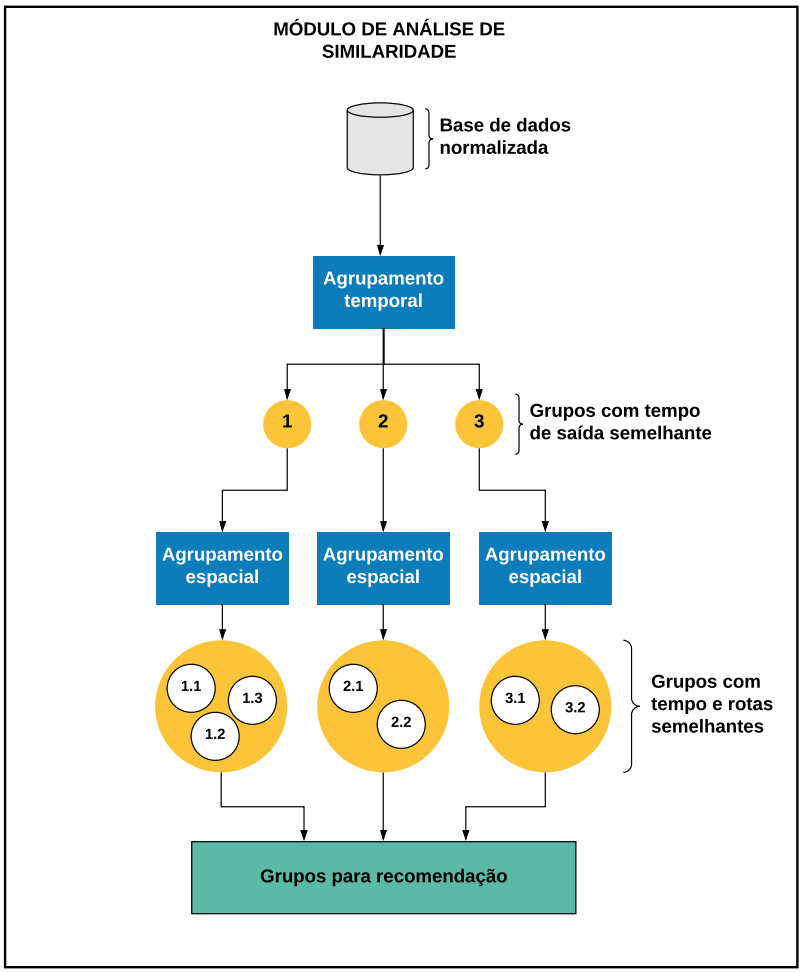
4.4 Módulo de análise de similaridade

O módulo de análise de similaridade, demonstrado na Figura 9, é responsável pela realização da identificação de similaridade dos dados de mobilidade. Após feita a análise, o módulo realiza o agrupamento dos usuários, deixando-os prontos para a recomendação.

4.4.1 Arquitetura do módulo de análise de similaridade

O módulo de análise possui algumas etapas definidas, iniciando com a captura dos dados normalizados. Logo após a captura dos dados, são realizadas as análises de similaridade e agrupamento e, por fim, a recomendação, conforme é exposto na Figura 10.

Figura 10 – Arquitetura do módulo de análise de similaridade



Fonte: Elaborada pelo autor

Inicialmente, os dados são colhidos da base de dados já normalizada, prontos para a realização das análises de similaridade. A etapa de agrupamento é feita através de duas fases, definidas como agrupamento temporal e agrupamento espacial, respectivamente. No agrupamento temporal, são gerados os primeiros grupos contendo usuários com tempos similares de saída em suas viagens. Em seguida, na segunda fase, dentro de cada grupo gerado na fase de agrupamento temporal, é realizado um novo agrupamento, porém, dessa vez, utilizando as rotas de cada viagem como análise. Com isso, são gerados os grupos que possuem viagens com tempo de saída e rotas semelhantes. Por fim, após o agrupamento ser concluído, os grupos gerados estão prontos para a recomendação.

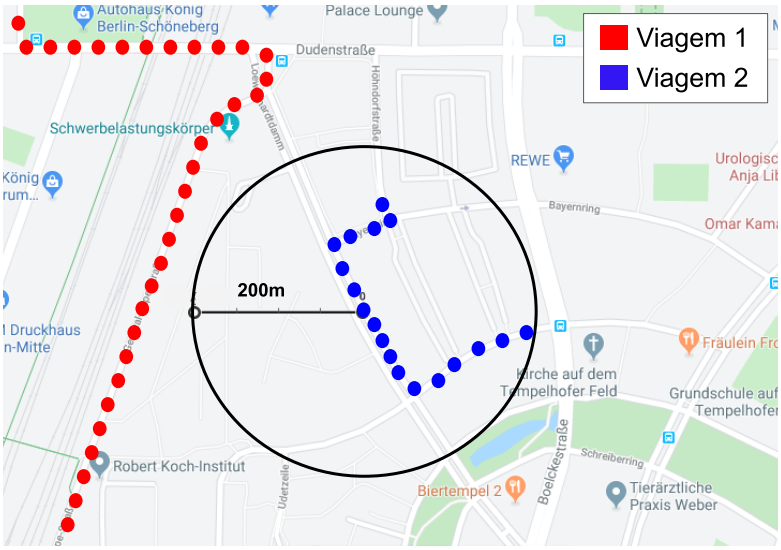
Uma arquitetura semelhante de análise de similaridade é proposta por (CRUZ; MACEDO; GUIMARAES, 2015), que possui duas abordagens de análise, uma temporal e outra espacial, da mesma forma que a arquitetura do ARCA. Entretanto a arquitetura proposta por (CRUZ; MACEDO; GUIMARAES, 2015) possui uma terceira etapa

que antecede as etapas de análise, sendo esta uma etapa de discretização da base dados, o que não acontece com a arquitetura utilizada no ARCA. Outra distinção entre as duas arquiteturas ocorre pelo fato de que a análise temporal do ARCA é feita através de algoritmos de agrupamento, observando o tempo de saída de cada viagem e gerando como resultado, grupos com tempo de saída semelhantes. Já a arquitetura proposta em (CRUZ; MACEDO; GUIMARAES, 2015) utiliza a análise temporal como um filtro para a redução do número de viagens a ser analisada considerando o tempo de saída e chegada das viagens.

Antes do início do agrupamento, algumas condições são propostas a fim de estabelecer os requisitos para se definir quando duas ou mais viagens são consideradas semelhantes. Primeiramente, é definido o intervalo de tempo máximo entre viagens. Para essa condição assume-se o intervalo máximo de 5 minutos entre duas ou mais viagens, o equivalente a 300 segundos, que corresponde a 300 unidades de tempo do *Unix Timestamp*, padrão temporal de tempo usado para a análise.

A segunda condição a ser considerada entre duas viagens, para determinar o grau de similaridade entre elas, é a distância. Para que duas ou mais viagens sejam consideradas similares em relação à distância entre elas, assume-se um valor de distância máximo de 200 metros. Ou seja, para que as viagens possam ser consideradas do mesmo grupo, cada ponto geográfico das microviagens deve necessariamente estar em um raio máximo de 200m entre si. A Figura 11 mostra uma visualização demonstrativa do raio de distância de verificação.

Figura 11 – Representação do raio de distância de verificação.



Fonte: Elaborada pelo autor

Como o conjunto de dados utiliza o padrão de coordenadas decimais para armazenar os dados espaciais das viagens, faz-se necessário calcular a distância equi-

valente ao intervalo assumido para a condição de distância descrita anteriormente. Para isso, adota-se o valor calculado proposto por (SIMÕES, 2019), onde, levando em consideração o perímetro da Terra, afirmou que a distância de 200 metros é equivalente a aproximados 0.0024 graus decimais.

4.4.2 Agrupamento temporal

Após a determinação das condições de agrupamento de tempo e de rotas definidos, inicia-se a fase de testes dos algoritmos. No primeiro agrupamento realizado, definem-se os grupos de rotas com seus respectivos usuários, observando o tempo de saída das viagens. É assumido que o tempo de chegada de uma viagem não é relevante para o experimento, tendo em consideração que empiricamente, um usuário, ao realizar a procura de uma carona, não se preocupa com o horário de chegada da viagem (RESENDE; CUZZUOL,), (CORDEIRO et al., 2017), (MACEDO; MACEDO, 2019).

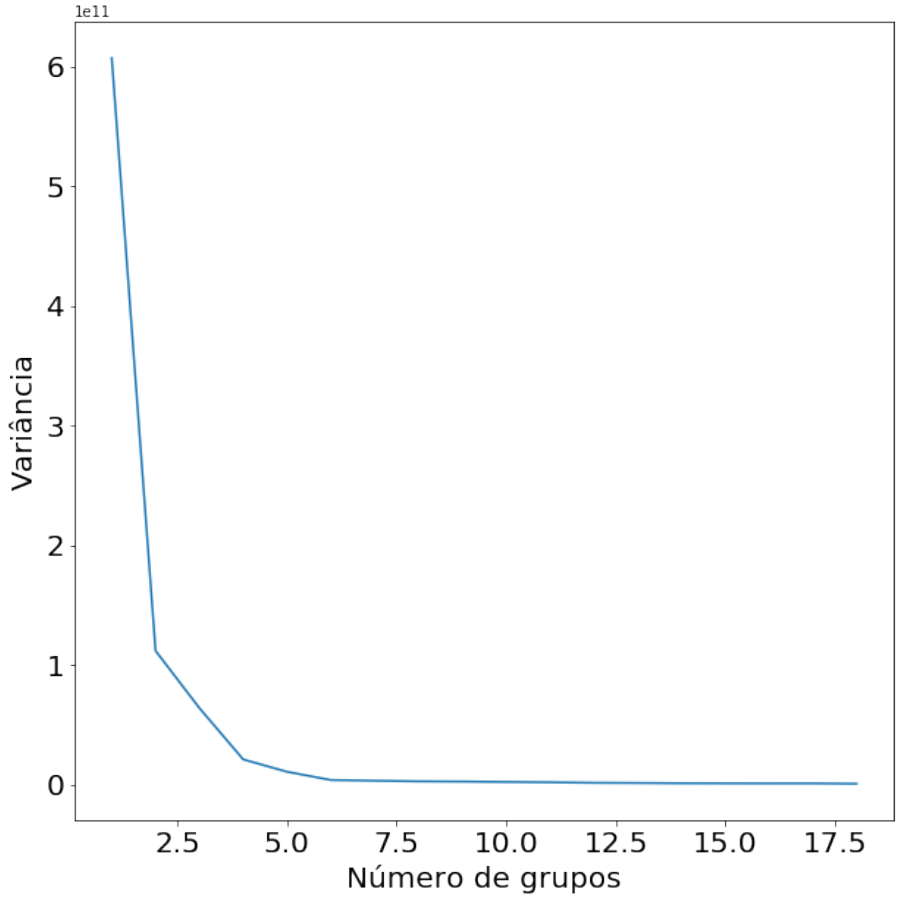
Para os algoritmos DBSCAN e OPTICS, é utilizado como parâmetro de verificação (*Eps* e *Max_eps*, respectivamente) o valor de 300 unidades do *Unix timestamp*, correspondentes aos 5 minutos de diferença, definido como distância máxima de tempo entre as viagens para que haja similaridade entre elas. Já como segundo parâmetro (Min_pts), que é um parâmetro em comum aos dois algoritmos, é definido o valor 2 como valor ideal, visto que um grupo de carona é composto por mais de um usuário. O algoritmo HDBSCAN, apesar de necessitar apenas do parâmetro *Min_pts* como valor de entrada, permite a utilização do parâmetro de entrada *Eps*. Desta forma, para este algoritmo, são utilizados os mesmos parâmetros *Eps* e *Min_pts* dos algoritmos DBSCAN E OPTICS.

Como o algoritmo K-means trabalha com o número de grupos (*n_clusters*) como parâmetro de entrada, encontrar o valor ideal para o algoritmo é um desafio. Como a base de dados é um conjunto baseado em parâmetros de rotas reais, não supervisionado, isso faz com que a quantidade de grupos seja desconhecida. Assim, utiliza-se o método Elbow para determinar o parâmetro de entrada para o K-means. O Método Elbow consiste em analisar a variação dos dados em relação ao número de grupos e, assim, determinar o melhor valor através dessa análise.

Para se determinar o número ideal de grupos, o método realiza vários agrupamentos analisando a variância desses grupos. Quando essa variância começa a se estabilizar, o ponto onde ocorre essa redução da variância é o número ideal de grupos para determinado conjunto de dados. Para se obter esse número deve-se plotar um gráfico dos resultados do método e analisar onde ocorre essa redução da variância. A Figura 12 mostra um exemplo de gráfico resultante da execução do método Elbow.

Percebe-se que a variância começa a se estabilizar quando o número de grupos é igual a 4. Então, nesse exemplo, o número ideal de grupos para ser usado como parâmetro é 4.

Figura 12 – Gráfico Método Elbow



Fonte: Elaborada pelo autor

4.4.3 Agrupamento espacial

O segunda etapa para se determinar os grupos de caronas com similaridade entre seus usuários e suas respectivas viagens é o agrupamento espacial. Essa etapa é feita através da análise dos dados geográficos das rotas de cada usuário. A análise é realizada dentro de cada grupo resultante do agrupamento temporal. Desta forma, os grupos de rotas similares gerados nesta etapa são subgrupos da etapa de agrupamento temporal. Assim, os grupos finais para a recomendação são formados.

Os valores de entrada para os parâmetros *Eps* dos algoritmos DBSCAN e HDBSCAN, bem como o *Max_eps*, do algoritmo OPTICS, seguem o critério de distância determinado na Seção 4.4, equivalente a 0.0024. Já o valor para *Min_pts* para os três algoritmos é igual a 2. Tal valor é determinado levando em consideração que um

grupo deve possuir no mínimo 2 viagens para ser considerado um grupo. Para o K-means, o valor de *n_clusters* usado, é a média dos números de grupos determinados pelo método Elbow em cada grupo resultante do agrupamento temporal.

Ao final desta etapa têm-se os grupos de usuários prontos para a recomendação, com similaridades entre suas rotas. É necessário ressaltar que, nas duas etapas de agrupamento, não buscam-se os melhores valores de parâmetros para se agrupar todos ou a maioria dos usuários. Por outro lado, buscam-se os parâmetros definidos que determinam quando dois usuários devem fazer parte de um grupo. Desta forma, não é garantido que todos os pontos são agrupados, visto que, em um ambiente real, as rotas dos usuários podem não ter nenhuma correlação de similaridade entre elas.

4.5 A API do ARCA

A Interface de Programação de Aplicação (API) do ARCA, demonstrada na Figura 9, é responsável pela integração entre o módulo de análise com o aplicativo móvel ARCA. Essa integração é feita através de um novo arquivo CSV resultante do módulo de análise contendo a atribuição dos grupos para cada usuário. Através da API, os dados analisados e agrupados no módulo de análise são transferidos para os usuários, de modo que cada usuário recebe automaticamente as recomendações dos grupos de carona. Os dados referentes à esses grupos são enviados através da API para o aplicativo móvel no formato JSON.

A tecnologia utilizada para o desenvolvimento do ARCA API foi o Flask. O Flask é um *framework* para desenvolvimento *Web* baseado na biblioteca *Web Server Gateway Interface* (WSGI)[6]. Esse *framework* foi criado com o intuito de facilitar e acelerar o desenvolvimento de aplicações *Web* [7]. O Flask permite a criação de aplicações de forma rápida e com fácil codificação, o que acelera e otimiza o tempo de desenvolvimento.

A escolha desta tecnologia foi pensada por conta da facilidade e rapidez para o desenvolvimento e também devido à linguagem utilizada nesse *framework*, o Python. A utilização da linguagem Python como linguagem base do Flask, permite uma melhor integração e utilização dos dados, já que as bibliotecas utilizadas no módulo de análise utilizam a mesma linguagem.

[6] https://wsgi.readthedocs.io/en/latest/index.html
[7] https://palletsprojects.com/p/flask/

4.6 Aplicativo Móvel do ARCA

Como visto na Seção 4.3, a etapa final do ARCA consiste no aplicativo móvel, que é o ambiente onde se faz a interação com o usuário. Por meio do aplicativo móvel (app), os usuários do ARCA podem receber as recomendações de forma automática vindas da API, sem a necessidade de intermédio do usuário. Assim, o app funciona como um consumidor da API, enviando e recebendo as requisições e retornando-as aos usuários.

Para o desenvolvimento do aplicativo móvel do ARCA foi utilizado o Flutter[8], um kit de desenvolvimento de aplicativos móveis, criado pela Google. Com o Flutter é possível criar aplicativos compilados nativamente para Android e também para IOS, através de uma única base de código. Além disso, o Flutter ainda permite o desenvolvimento *Web* e *Desktop*. A escolha do Flutter foi feita devido à praticidade no desenvolvimento e pela possibilidade de obter um aplicativo multiplataforma no final do desenvolvimento.

O desenvolvimento do aplicativo móvel do ARCA foi realizado com base no fluxograma apresentado na Figura 13. A sequência de funcionamento do aplicativo é simples. Inicialmente o usuário realiza o *login* no aplicativo, através do número de autenticação. Devido à base de dados não possuir informações de usuário, utilizou-se o atributo **Moid**, que corresponde ao identificador do veículo, como parâmetro para definir o número de autenticação do usuário. Portanto, a premissa é que cada veículo está relacionado a apenas um usuário.

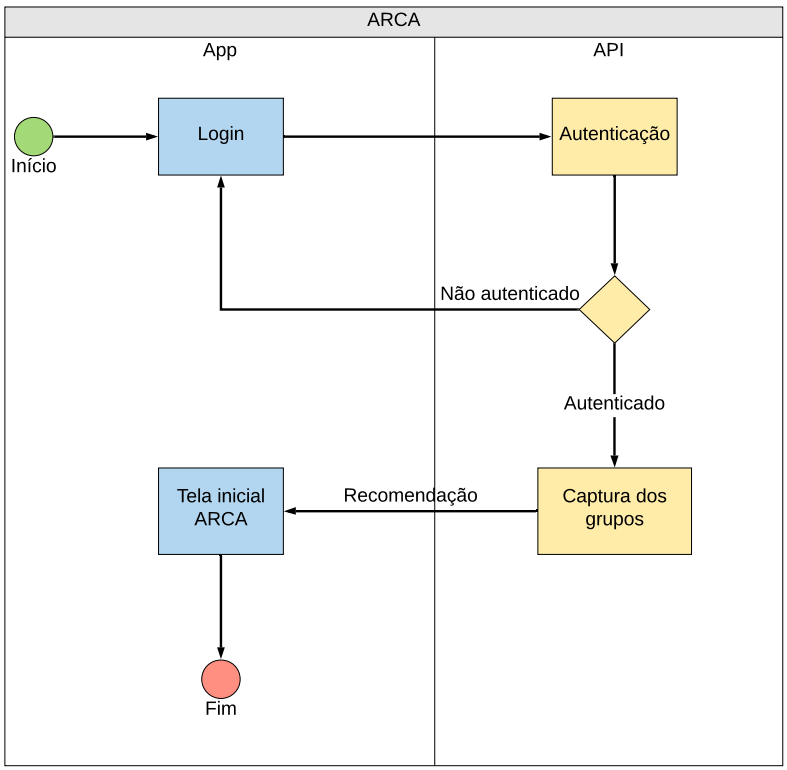
Após o usuário inserir seu identificador, o aplicativo realiza a conexão com a API para a realização da autenticação do usuário. Caso o usuário não esteja cadastrado no ARCA, ele é redirecionado à tela de login do aplicativo. Estando o usuário cadastrado, ele é autenticado e, automaticamente, a API realiza a captura de todos os grupos de carona criados no módulo de análise, como explicado na Seção 4.4.1, em que o usuário faz parte.

Após a captura de todos os grupos relacionados ao respectivo usuário, estes grupos são enviados para o aplicativo móvel como recomendação de carona. O aplicativo móvel recebe esses grupos e disponibiliza-os para que o usuário possa visualizá-los. Nesses grupos, ele visualiza os usuários que possuem rotas semelhantes a ele.

A representação dos usuários no aplicativo é feita através do conjunto de dados, também disponibilizado pelo BelinMOD, que contém os dados relacionados a cada veículo. Desta forma, cada veículo foi atribuído como representação de um usuá-

[8] https://flutter.dev

Figura 13 – Fluxograma de atividades do aplicativo móvel ARCA



Fonte: Elaborada pelo autor

rio, visto que cada veículo só poderá possuir um único usuário como proprietário. O conjunto de dados referente possui os campos a seguir:

- **Moid** - Valor referente ao identificador de cada veículo na base de dados;
- **License** - Representa o conjunto de caracteres que correspondem ao licenciamento do veículo (Número da placa);
- **Type** - Valor correspondente ao tipo de veículo;
- **Model** - Valor correspondente ao modelo do veículo.

4.6.1 Telas do Aplicativo Móvel do ARCA

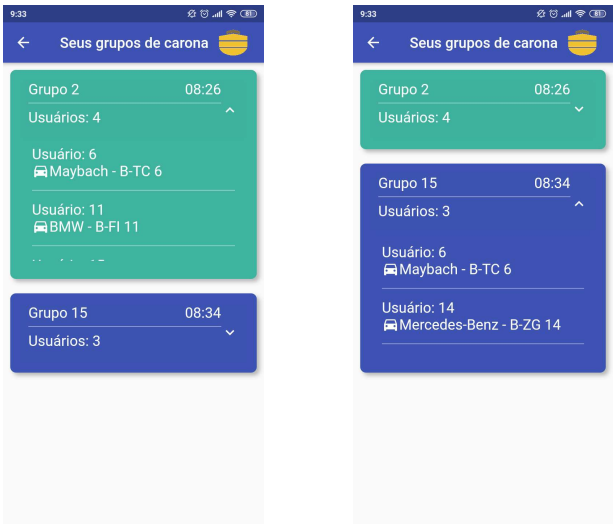
O ARCA App é o resultado final de todo o processo realizado nesta proposta. Neste app todos os dados gerados através do módulo de análise são disponibilizados ao usuário de forma transparente e automática através da API. Assim o usuário pode visualizar os grupos de caronas com usuários que possuem similaridade em suas rotas.

A primeira tela do ARCA é a tela de login. Nesta tela, é disponibilizado ao usuário um campo de digitação, no qual o usuário fornece suas credenciais para a autenticação e por fim obtém acesso à tela inicial, contendo os grupos de caronas referentes a este usuário.

A segunda tela do aplicativo ARCA é a tela individual do usuário, após a realização da autenticação. Nesta tela são disponibilizados todos os grupos de carona pertencentes ao determinado usuário autenticado no aplicativo. O usuário pode visualizar todos os grupos de carona, onde cada grupo possui: o número identificador do grupo; o horário de saída da respectiva viagem que o usuário possui similaridade com os demais usuários do grupo; e a quantidade de usuários daquele grupo.

O usuário pode clicar nos grupos e ao realizar esta ação, ele tem disponível a visualização das informações de cada grupo, como mostra a Figura 14. Tais informações apresentadas são, os identificadores dos usuários do grupo, o modelo do veículo de cada usuário e a numeração da placa do veículo.

Figura 14 – Grupos de carona expandidos



(a) (b)

Fonte: Elaborada pelo autor

4.7 Configurações do experimento

Durante o processo de construção do ARCA foi realizado o experimento para a avaliação dos algoritmos. Após o tratamento dos dados 4.2, inicia-se a etapa dos testes dos algoritmos de agrupamento para determinar qual algoritmo, dentre os selecionados para o experimento, é mais adequado para a utilização no ARCA. Utiliza-se neste experimento os algoritmos K-means 2.6.1, DBSCAN 2.6.2, OPTICS 2.6.3 e

HDBSCAN 2.6.4, todos disponíveis através da biblioteca *scikit-learn*[9], com exceção do HDBSCAN, que está disponível em sua biblioteca própria, *hdbscan*[10]. Foram escolhidos esses algoritmos devido as diferentes abordagens com que cada um realiza a análise de similaridade. Todos os experimentos foram realizados usando a linguagem Python, através do *Google Colaboratory*, uma plataforma *Web* que permite a execução de códigos Python *online*[11].

A determinação do algoritmo mais adequado para esta proposta se dá através da análise de um conjunto fatores considerados, dentre eles: (1) a sensibilidade aos pontos ruidosos *(outliers)*; (2) a quantidade e os tipos de parâmetros de entrada; (3) a avaliação de agrupamento através do Índice de Davies-Bouldin; (4) a verificação manual de grupos; (5) a quantidade de grupos gerados por cada algoritmo em relação à base de dados; (6) e o tempo de execução de cada algoritmo. Todos os resultados destas análises estão dispostas no Capítulo 5. Para o conjunto de dados do experimento, utiliza-se uma fração da base dados do BerlinMOD 4.1, contendo 60 mil pontos, o que equivale a aproximadamente 317 viagens, visto que o grande volume de dados requer maiores recursos de processamento. Assim, essa redução facilitou a realização dos experimentos.

[9] https://scikit-learn.org/stable/modules/classes.html#module-sklearn.cluster
[10] https://hdbscan.readthedocs.io/en/latest/index.html
[11] https://colab.research.google.com/notebooks/intro.ipynb#scrollTo=5fCEDCU_qrC0

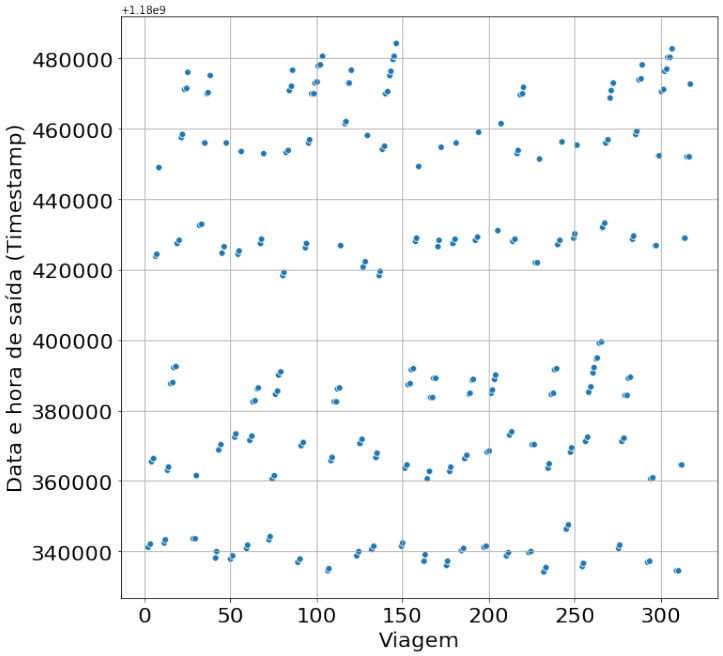
5 AVALIAÇÃO EXPERIMENTAL

Após todo o processo de construção do ARCA, apresentado no Capítulo 4, o presente capítulo apresenta as análises feitas durante os experimentos e os resultados obtidos ao final desta proposta. Para avaliar qual algoritmo obteve melhor aplicabilidade para o ARCA, alguns pontos foram instituídos tanto para o agrupamento temporal como o agrupamento espacial, sendo eles: análise dos Índices de Davies-Bouldin, Tempo de execução, Número de grupos gerados, parâmetros de entrada. Para o agrupamento temporal, ainda foi realizada quais algoritmos realizam a detecção de ruídos além de uma verificação manual de cada algoritmo.

5.1 Avaliação do agrupamento temporal

A Figura 15 demonstra a relação entre cada viagem e sua data e hora de saída (em *Unix Timestamp*) do conjunto de dados sem nenhum agrupamento. Cada ponto do gráfico representa o número de uma viagem. No eixo X estão os valores correspondentes ao número identificador das viagem e no eixo Y os valores de tempo de saída de cada viagem. Nota-se que é um conjunto com uma quantidade de dados significativa para os experimentos realizados.

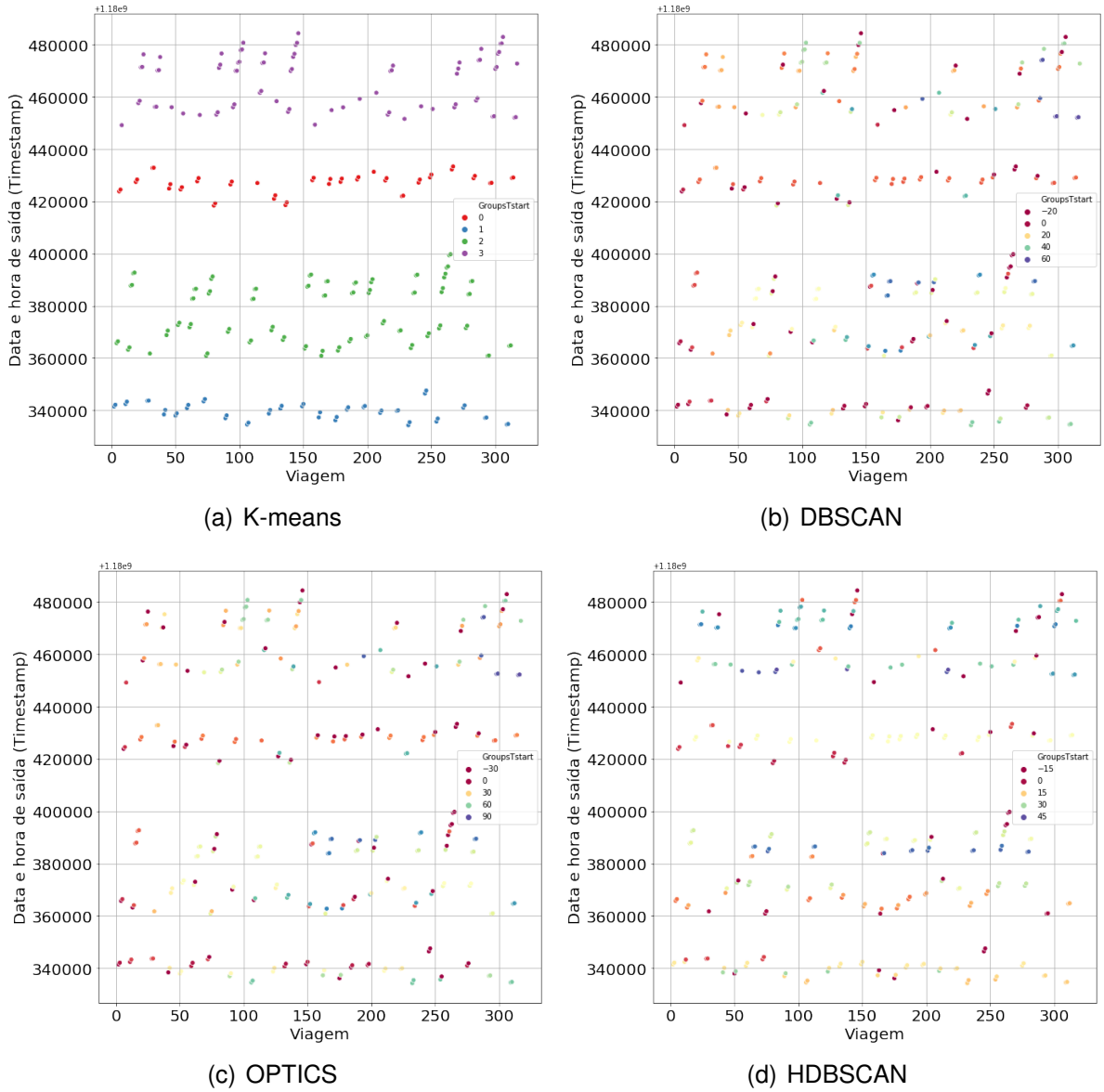
Figura 15 – Dados desagrupados



Fonte: Elaborada pelo autor

Após a passagem do mesmo conjunto de dados da Figura 15 pelos algoritmos, todos foram capazes de realizar o agrupamento temporal. A Figura 16 mostra visualmente o agrupamento realizado por cada algoritmo, e como cada algoritmo trabalha. Pode-se notar, inicialmente, que o K-means não realiza tratamento de pontos de ruído (*outliers*), ou seja, todos os pontos são agrupados, mesmo havendo distinção entre alguns. Isso não acontece com os demais algoritmos, no qual os grupos com índices de valor negativo são grupos de pontos ruidosos.

Figura 16 – Detecção de pontos de ruído do agrupamento temporal



(a) K-means (b) DBSCAN

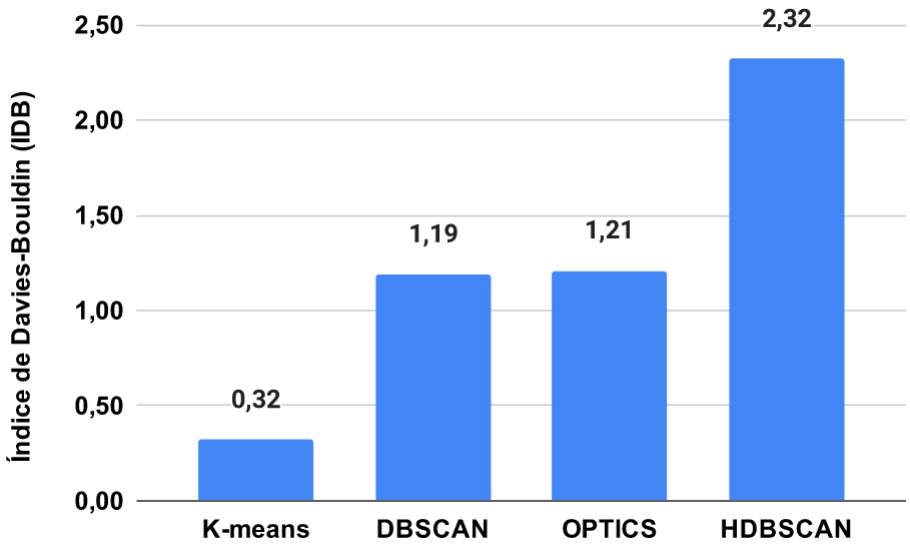
(c) OPTICS (d) HDBSCAN

Fonte: Elaborada pelo autor

5.1.1 Índice de Davies-Bouldin para o agrupamento temporal

Avaliar a eficiência de um algoritmo de agrupamento não é uma tarefa tão simples, dado que esse tipo de técnica trabalha sob perspectivas de dados não supervisionados. A análise através do Índice de Davies-Bouldin (IDB) permite avaliar o desempenho em relação ao agrupamento dos dados. Nos experimentos, ao final do agrupamento temporal, em cada algoritmo era calculado o IDB. A Figura 17 mostra o gráfico contendo os valores obtidos de IDB de cada algoritmo e, como citado na Seção 2.7, quanto mais próximo de zero o valor, melhor foi a eficiência do agrupamento.

Figura 17 – Comparativo dos Índices de Davies-Bouldin para o agrupamento temporal



É possível notar que o algoritmo K-means se sobressai sobre os demais algoritmos, o que o faz ser mais eficiente. Entretanto, o IDB tende a efetuar resultados melhores para algoritmos de agrupamentos cujo resultado são grupos convexos, como os gerados pelo K-means, diferente dos algoritmos que geram grupos não-convexos, como os algoritmos baseados em densidade [1]. Tendo em vista esse fato, os algoritmos DBSCAN e OPTICS obtiveram um desempenho consideravelmente melhor quando submetidos ao agrupamento temporal em relação ao IDB.

5.1.2 Tempo de execução do agrupamento temporal

O tempo de execução tem grande influência sobre a escolha do algoritmo, pois algoritmos rápidos e eficientes demonstram uma melhor dinâmica para se trabalhar com grandes volumes de dados em aplicações de tempo real. Na Tabela 2 estão dispostos os tempos de execução de cada algoritmo.

[1] https://scikit-learn.org/stable/modules/clustering.thmlclustering-performance-evaluation

Tabela 2 – Tabela comparativa entre os tempos de execução

	Tempo de execução
K-means	0.69 s
DBSCAN	0.614 s
OPTICS	0.641 s
HDBSCAN	0.927 s

A diferença no tempo de execução entre os algoritmos não foi tão significativa, visto que não houve grandes variações entre eles, com exceção do HDBSCAN, que obteve uma diferença um pouco mais acentuada em relação aos demais. Entretanto, mesmo o HDBSCAN possuindo uma variação maior com relação aos demais algoritmos, todos obtiveram resultados satisfatórios para a utilização nesta proposta em relação ao tempo de execução.

5.1.3 Número de grupos gerados no agrupamento temporal

Em um ambiente real, é esperado que as viagens possuam diversos intervalos de tempo de saída. Em um agrupamento, a escolha do valor do intervalo de verificação, pode acarretar em grandes quantidades de grupos. É demonstrada, na Tabela 3, a quantidade de grupos, por tempo de saída, gerados pelos algoritmos.

Tabela 3 – Tabela comparativa entre as quantidades de grupos gerados

	Número de grupos gerados
K-means	4
DBSCAN	54
OPTICS	82
HDBSCAN	44

Levando em consideração o intervalo de tempo de 5 minutos, estabelecido nos experimentos e, um dia contendo 24 horas, é esperado que haja variados grupos. Na Tabela 3, pode-se observar que os algoritmos DBSCAN, OPTICS e HDBSCAN, possuem um melhor rendimento nessa categoria em relação ao K-means. O OPTICS, em específico, se sobressaiu em relação aos outros, confirmando as expectativas.

5.1.4 Parâmetros de entrada do agrupamento temporal

Aqui, foram analisados os parâmetros de entrada dos algoritmos para verificar qual algoritmo possui uma melhor forma de se trabalhar, visto as necessidades da

proposta. Logo, se observa que o K-means possui uma grande dificuldade para determinar o parâmetro de entrada do algoritmo. Mesmo através do método Elbow, ainda é difícil determinar o valor, já que para isso deve-se observar o gráfico resultante do método. Além disso, essa observação não é algo preciso, o que pode comprometer os resultados. Outro ponto negativo do K-means, em relação aos parâmetros de entrada para a proposta, é o fato de que o conjunto de dados possui um número desconhecido de grupos. Desta forma, estipular um valor não é algo viável para esse tipo de situação.

Os algoritmos DBSCAN, OPTICS e HDBSCAN possuem algumas vantagens em relação aos parâmetros de entrada. Tais algoritmos permitem que sejam estabelecidas as condições necessárias para que duas ou mais viagens possam ser consideradas do mesmo grupo através dos parâmetros *Min_pts* e *Eps* (Utilizados no DBSCAN e HDBSCAN) e *Max_eps* (Utilizado no OPTICS). Outra vantagem observada desses algoritmos, é que eles são capazes de determinar a quantidade de grupos sem a necessidade de um valor fixo como entrada, o que facilita e agiliza o processo de agrupamento.

5.1.5 Verificação manual

Nessa etapa foram realizadas análises dos grupos gerados por cada algoritmo, com a finalidade de se obter uma visualização manual do comportamento de alguns dos grupos desses algoritmos e comparar esses grupos.

A Tabela 4 mostra um dos grupos, formado pelo agrupamento temporal gerados pelo K-means. A coluna **Tripid** representa o identificador da viagem, e a coluna **Tstart** é o tempo de saída da respectiva viagem. Pode-se observar que o algoritmo não obtém um resultado satisfatório com relação ao intervalo de tempo estabelecido para a similaridade das viagens, havendo grandes diferenças no intervalo de tempo de saída das viagens agrupadas. Esse comportamento torna-se insatisfatório para os objetivos desta proposta.

Na Tabela 5 estão dispostas as viagens de um dos grupos, gerado pelo DBSCAN em relação ao tempo de saída das viagens. Em relação ao algoritmo K-means, o DBSCAN obteve melhores resultados nesse tipo de agrupamento, obtendo viagens agrupadas em intervalos de tempo mais próximos do ideal estabelecido para a proposta. Entretanto, observa-se que ainda há algumas viagens com intervalos de tempo de, aproximadamente, 20 minutos entre elas.

Um dos grupos gerados pelo algoritmo OPTICS está disposto na Tabela 6, como exemplo da análise feita. O OPTICS demonstrou resultados consistentes em relação ao agrupamento temporal, pois pode-se observar que as viagens estão agru-

Tabela 4 – Grupo por tempo de saída gerado pelo K-means

Tripid	Tstart
8	2007-05-29 14:33:03.967
21	2007-05-29 16:53:53.009
22	2007-05-29 17:08:51.657
23	2007-05-29 20:42:34.482
24	2007-05-29 20:44:54.550
305	2007-05-29 23:14:41.287
306	2007-05-29 23:56:10.631
315	2007-05-29 15:21:03.793
316	2007-05-29 15:24:10.029
317	2007-05-29 21:06:26.636

Tabela 5 – Grupo por tempo de saída gerado pelo DBSCAN

Tripid	Tstart
2	2007-05-28 08:36:47.846
3	2007-05-28 08:47:55.920
11	2007-05-28 08:52:53.673
59	2007-05-28 08:26:50.970
60	2007-05-28 08:46:56.034
132	2007-05-28 08:23:33.832
133	2007-05-28 08:41:16.442
149	2007-05-28 08:38:24.995
150	2007-05-28 08:53:09.862
185	2007-05-28 08:31:32.882
197	2007-05-28 08:32:51.629
198	2007-05-28 08:40:07.934
275	2007-05-28 08:29:21.611
276	2007-05-28 08:44:10.611

padas mais próximas do intervalo de tempo de saída estabelecido, o que é bastante positivo. Isso coloca o OPTICS em vantagem com relação aos algoritmos K-means e DBSCAN, no agrupamento temporal.

A Tabela 7 mostra um dos grupos formados pelo algoritmo HDBSCAN no agrupamento por tempo. Nesta tabela pode-se observar que os intervalos de tempo de saída entre a viagens possuem um espaço de tempo dentro do intervalo estabelecido, porém, com algumas pequenas exceções, o que relativamente não compromete o rendimento desse algoritmo no agrupamento por tempo de saída. Os resultados

Tabela 6 – Grupo por tempo de saída gerado pelo OPTICS

Tripid	Tstart
59	2007-05-28 08:26:50.970
132	2007-05-28 08:23:33.832
185	2007-05-28 08:31:32.882
197	2007-05-28 08:32:51.629
275	2007-05-28 08:29:21.611

observados pelo HDBSCAN estão relativamente próximos aos gerados pelo OPTICS.

Tabela 7 – Grupo por tempo de saída gerado pelo HDBSCAN

Tripid	Tstart
6	2007-05-29 07:30:41.966
7	2007-05-29 07:41:56.995
45	2007-05-29 07:48:12.870
54	2007-05-29 07:42:41.026
55	2007-05-29 07:56:50.303

De forma geral, o algoritmo OPTICS obteve melhores resultados no processo de agrupamento temporal, quando comparado com os demais algoritmos utilizados neste experimento. Como demonstrado na Tabela 8, os algoritmos DBSCAN e OPTICS obtiveram resultados relativamente equivalentes, bem como em relação aos parâmetros de entrada de verificação 5.1.4. Entretanto, quando feita a análise manual dos grupos 5.1.5, verificou-se que o OPTICS obteve grupos com similaridade mais próximas do limiar estabelecido de tempo. Assim, o OPTICS se mostrou eficiente em todas as categorias analisadas, o que o colocou como escolha para ser utilizado no agrupamento temporal do ARCA.

Tabela 8 – Tabela geral dos resultados do agrupamento temporal

	Número de Grupos	IDB	Tempo de execução(s)	Detecção de ruídos
K-means	4	0.32	0.69	Não
DBSCAN	54	1.19	0.614	Sim
OPTICS	82	1.21	0.641	Sim
HDBSCAN	44	2.32	0.927	Sim

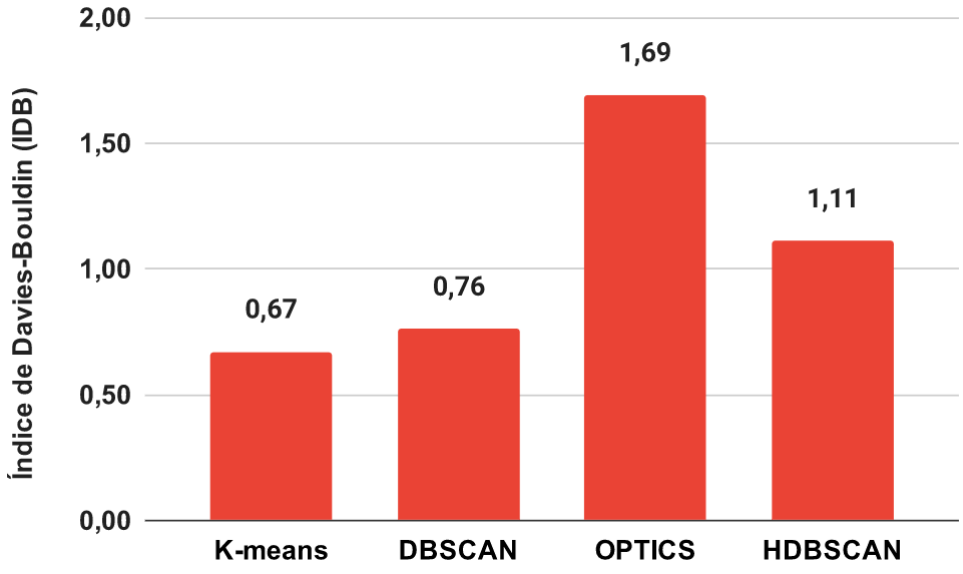
5.2 Avaliação do agrupamento espacial

A segunda fase da avaliação foi a análise dos resultados dos agrupamentos dos algoritmos em relação às rotas das viagens. As análises levaram em consideração o IDB, o tempo de execução, a quantidade de grupos formados e os parâmetros de entrada.

5.2.1 Índice de Davies-Bouldin do agrupamento espacial

A Figura 18 mostra o gráfico com todos os valores do IDB resultantes do agrupamento por rotas de cada algoritmo de agrupamento utilizado. Nesta tabela é possível observar que novamente o algoritmo K-means possui um IDB melhor, se comparado com os demais algoritmos. Entretanto, o algoritmo DBSCAN obteve um IDB favorável em comparação com os algoritmos OPTICS e HDBSCAN, que trabalham de forma semelhante ao DBSCAN. Também pode-se observar que os algoritmos OPTICS e HDBSCAN não possuem grandes diferenças entre eles.

Figura 18 – Comparativo dos Índices de Davies-Bouldin para o agrupamento temporal



5.2.2 Tempo de execução do agrupamento espacial

Em relação ao tempo de execução para o agrupamento de rotas o algoritmo DBSCAN demonstrou o melhor resultado nessa fase de agrupamento. O DBSCAN obteve um tempo de execução de 1,357 segundos, seguido do algoritmo K-means com

1,593 segundos. Na sequência o algoritmo HDBSCAN com 3,32 segundos, demonstrou um tempo considerável em comparação com os anteriores. O OPTICS obteve o valor mais extremo, atingindo 44,776 segundos, valor este, em comparação com os demais, bastante expressivo.

5.2.3 Número de grupos gerados no agrupamento espacial

A Tabela 9 mostra a quantidade de grupos gerados por agrupamento espacial dos algoritmos. Podemos observar que o algoritmo OPTICS gerou um número expressivo de grupos, se comparado com os demais algoritmos, equivalente a 9812 grupos. Os algoritmos DBSCAN e HDBSCAN demonstram valores relativamente equivalentes, o que pode indicar uma quantidade mais próxima do real ao número de grupos esperados.

Tabela 9 – Tabela comparativa entre as quantidades de grupos gerados

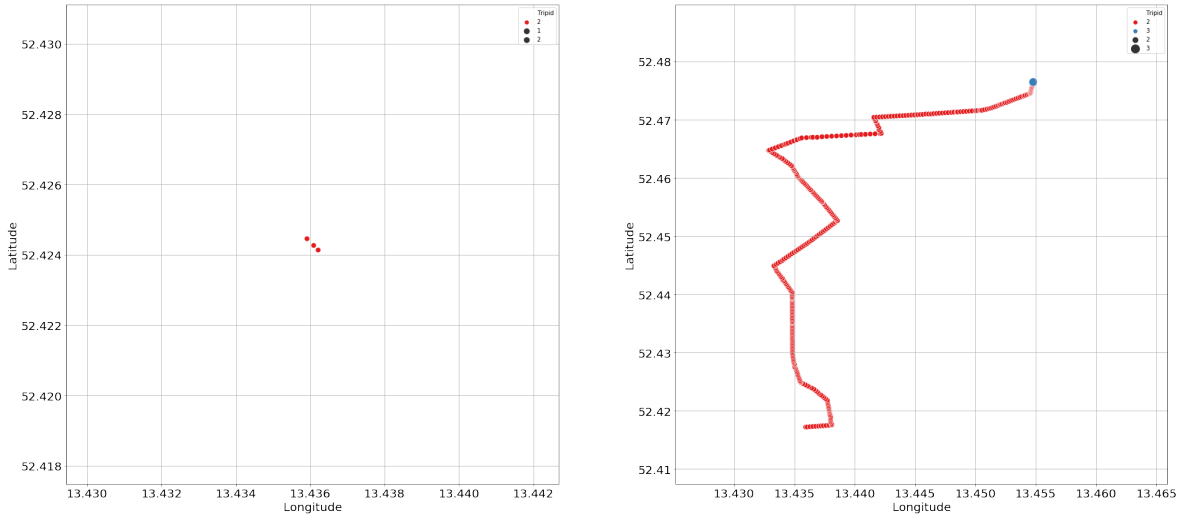
	Número de grupos gerados
K-means	12
DBSCAN	96
OPTICS	9812
HDBSCAN	129

O número expressivo de grupos gerados pelo OPTICS ocorreu devido ao fato de que o algoritmo realizou agrupamentos muito pequenos em relação as rotas. A Figura 19 mostra o agrupamento de uma das viagens feita pelos algoritmos OPTICS, em *(a)* e pelo DBSCAN, em *(b)*. Vemos que o OPTICS agrupou apenas uma pequena parte de uma viagem. Já o DBSCAN agrupou os pontos da mesma viagem em um mesmo grupo de forma completa.

5.2.4 Parâmetros de entrada do agrupamento espacial

Da mesma forma que o agrupamento temporal, no agrupamento espacial, também foi verificado quais os parâmetros utilizados pelos algoritmos para se determinar qual algoritmo teria melhor relevância para se utilizar no ARCA. Como no agrupamento temporal, o K-means demonstra novamente ser limitante devido a utilização do número de grupos a serem gerados como parâmetro de entrada, tornando sua utilização inviável para esse tipo de aplicação. Os demais algoritmos, por utilizarem parâmetros equivalentes de entrada, sendo eles, *Eps ou Max_eps* e *Min_pts*, tornam-se mais aptos para o ARCA. Esse fato torna possível determinar as condições

Figura 19 – Agrupamento das viagens pelo OPTICS e DBSCAN



(a) Agrupamento OPTICS (b) Agrupamento DBSCAN

Fonte: Elaborada pelo autor

de verificação estabelecidas no agrupamento espacial para que duas viagens sejam similares.

A Tabela 10 demonstra de forma resumida, os dados obtidos dos resultados colhidos da análise espacial. Com base nos dados analisados da avaliação do agrupamento espacial, nota-se que o algoritmo DBSCAN apresentou os melhores resultados em todas as categorias, incluindo os parâmetros de entrada, em relação aos demais algoritmos. Desta forma, para a realização o agrupamento espacial realizado pelo ARCA, foi escolhido o algoritmo DBSCAN.

Tabela 10 – Tabela geral dos resultados do agrupamento espacial

	Número de Grupos	IDB	Tempo de execução(s)
K-means	12	0.67	1.593
DBSCAN	96	0.76	1.357
OPTICS	9812	1.69	44.776
HDBSCAN	129	1.11	3.32

6 CONCLUSÕES E TRABALHOS FUTUROS

Com este trabalho pode-se observar a importância e a relevância de se propor aplicações com o objetivo de melhorar as problemáticas relacionadas à mobilidade urbana, através de funcionalidades que auxiliam na redução da quantidade de veículos transitando nas ruas. Desta forma, o ARCA cumpre seu papel com uma estratégia viável de promoção de caronas de forma automática e transparente ao usuário, visando a redução do número de veículo nas ruas e, consequentemente, diminuindo os problemas de saúde pública, mobilidade, problemas econômicos e ambientais.

Além dessas contribuições, o ARCA mostra que as tecnologias atuais existentes são eficientes para o desenvolvimento de aplicações inteligentes, voltadas para a promoção de caronas, através do compartilhamento de veículos. Além disso, este trabalho analisa as principais vantagens e desvantagens das tecnologias atuais utilizadas para este tipo de aplicação. Por fim, o ARCA demonstra um potencial elevado como uma solução viável no compartilhamento de caronas baseado em algoritmos de similaridade de mobilidade de veículos.

Como trabalhos futuros, pretende-se evoluir a arquitetura do ARCA para uma versão comercial, visando adaptar o comportamento do ARCA à um ambiente dinâmico de dados. Também pretende-se realizar a adição e implementação de funcionalidades de coleta de dados de viagens dos usuários, utilizando o GPS de cada usuário cadastrado na plataforma, com a finalidade de tornar o ARCA uma aplicação completa, com coleta, análise de rotas e recomendação de caronas. Outra funcionalidade a ser implementada é a de gerenciamento dos grupos de carona recomendados, permitindo ao usuário interagir com os demais membros do grupo. Além disso, pretende-se a implementação de uma versão *Web* para o ARCA, com a objetivo de promover ao usuário outra alternativa para gerenciar os grupos de carona.

REFERÊNCIAS

ACEA, E. A. M. A. *Size Distribution of Vehicle Fleet*. 2017. Https://www.acea.be/statistics/tag/category/size-distribution-of-vehicle-fleet. Acessado em: 1 de junho de 2019. Citado na página 14.

AMARAL, F. *Introdução à ciência de dados: mineração de dados e big data*. [S.l.]: Alta Books Editora, 2016. Citado na página 21.

AMIGOEXPRESS. *AmigoExpress, o que é isso?* 2019. Https://www.amigoexpress.com/faq/p. Acessado em: 29 de agosto de 2019. Citado 2 vezes nas páginas 16 e 30.

ANKERST, M. et al. Optics: ordering points to identify the clustering structure. *ACM Sigmod record*, ACM New York, NY, USA, v. 28, n. 2, p. 49–60, 1999. Citado na página 26.

ANTONESCU, B.; MOAYYED, M. T.; BASAGNI, S. Clustering algorithms and validation indices for a wide mmwave spectrum. *Information*, Multidisciplinary Digital Publishing Institute, v. 10, n. 9, p. 287, 2019. Citado na página 27.

ARORA, P.; VARSHNEY, S. et al. Analysis of k-means and k-medoids algorithm for big data. *Procedia Computer Science*, Elsevier, v. 78, p. 507–512, 2016. Citado na página 23.

BARCELOS, L. R.; SILVA, N. R. da. *MOBILIDADE URBANA NO BRASIL: UM DIREITO SOCIAL. URBAN MOBILITY IN BRAZIL: A SOCIAL RIGHT.* 2018. Http://periodicos.pucminas.br/index.php/virtuajus/article/view/19051/19051-68208-1. Citado na página 18.

BLABLACAR. *Como Funciona?* 2019. Https://blog.blablacar.com.br/about-us. Acessado em: 29 de agosto de 2019. Citado 3 vezes nas páginas 16, 29 e 30.

CAMARGO, M.; SANTOS, R. Estudo comparativo de algoritmos de agrupamento para a definição de zonas de manejo. In: *Os Anais da X Escola Regional de Informática de Mato Grosso*. Porto Alegre, RS, Brasil: SBC, 2019. p. 55–60. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/eri-mt/article/view/8594>. Citado na página 27.

CAMPELLO, R. J.; MOULAVI, D.; SANDER, J. Density-based clustering based on hierarchical density estimates. In: SPRINGER. *Pacific-Asia conference on knowledge discovery and data mining*. [S.l.], 2013. p. 160–172. Citado na página 26.

CARPOOL, W. 2019. Https://www.waze.com/pt-BR/carpool. Acessado em: 29 de agosto de 2019. Citado 2 vezes nas páginas 16 e 31.

CARVALHO, C. H. R. de. *EMISSÕES RELATIVAS DE POLUENTES DO TRANSPORTE MOTORIZADO DE PASSAGEIROS NOS GRANDES CENTROS URBANOS BRASILEIROS*. 2011. Http://www.ipea.gov.br/portal/images/stories/PDFs/TDs/td$_1606.pdf. Acessado em: 23 de junho de 2019. Citado na página 15.

COMMISSION, E. *Urban mobility*. 2019. Https://ec.europa.eu/transport/themes/urban/urban$_{m}obility_{e}n.Acessadoem : 23dejunhode2019.Citadonapgina16$.

CORDEIRO, A. d. A. et al. Carona solidária: uma opção de mobilidade no município de florianópolis-sc. Florianópolis, SC, 2017. Citado na página 40.

CRUZ, M. O.; MACEDO, H.; GUIMARAES, A. Grouping similar trajectories for carpooling purposes. In: IEEE. *2015 Brazilian Conference on Intelligent Systems (BRACIS)*. [S.l.], 2015. p. 234–239. Citado 2 vezes nas páginas 38 e 39.

CRUZ, M. O. da. On the similarity of users in carpooling recommendation computational systems. 2016. Citado 4 vezes nas páginas 18, 19, 29 e 31.

DAVIES, D. L.; BOULDIN, D. W. A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, n. 2, p. 224–227, 1979. Citado 2 vezes nas páginas 27 e 28.

DÜNTGEN, C.; BEHR, T.; GÜTING, R. H. Berlinmod: a benchmark for moving object databases. Springer, 2008. Citado na página 35.

EKMAN, F. et al. Working day movement model. 2008. Citado na página 29.

FERREIRA, R. M. et al. Pré-processamento de dados de trajetórias para mineração de dados e análise de similaridade. Florianópolis, SC, 2017. Citado na página 36.

FORBES; MCCARTHY, N. *The World's Worst Cities For Traffic Congestion [Infographic]*. 2019. Https://www.forbes.com/sites/niallmccarthy/2019/06/05/the-worlds-worst-cities-for-traffic-congestion-infographic/2ac04ea012bc. Acessado em: 1 de junho de 2019. Citado na página 15.

FURUHATA, M. et al. Ridesharing: The state-of-the-art and future directions. 2013. Citado na página 18.

G1; RAMALHO, G. *Brasil perde R$ 267 bilhões por ano com congestionamentos*. 2018. Https://g1.globo.com/globonews/noticia/2018/08/07/brasil-perde-r-267-bi-por-ano-com-congestionamentos.ghtml. Acessado em: 23 de junho de 2019. Citado na página 15.

GAO, X.; YU, F. Trajectory clustering using a new distance based on minimum convex hull. In: IEEE. *2017 Joint 17th World Congress of International Fuzzy Systems Association and 9th International Conference on Soft Computing and Intelligent Systems (IFSA-SCIS)*. [S.l.], 2017. p. 1–6. Citado na página 24.

GHOSEIRI, K.; HAGHAN, A.; HAMEDI, M. *Real-Time Rideshare Matching Problem*. 2011. Http://www.mautc.psu.edu/docs/umd-2009-05.pdf. Citado na página 19.

GOLDSCHMIDT, R. R. *Uma Introdução à Inteligência Computacional: fundamentos, ferramentas e aplicações*. [S.l.: s.n.], 2010. Citado na página 20.

GOMES, J. C.; PIMENTA, R. M.; SCHNEIDER, M. Mineração de dados na pesquisa em ciência da informação: Desafios e oportunidades. In: *ENANCIB 2019*. [S.l.: s.n.], 2019. Citado na página 21.

HAENLEIN, M.; KAPLAN, A. A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. 2019. Acessado em 20 de janeiro de 2020. Citado na página 19.

HOSEA, S. P.; HARIKRISHNAN, V.; RAJKUMAR, K. Artificial intelligence. 2011. Citado 2 vezes nas páginas 19 e 20.

HOU, J.; GAO, H.; LI, X. Dsets-dbscan: a parameter-free clustering algorithm. *IEEE Transactions on Image Processing*, IEEE, v. 25, n. 7, p. 3182–3193, 2016. Citado na página 24.

KRONBAUER, A. M.; FONTOURA, L. M.; WINCK, A. T. Um estudo sobre processos para avaliação de algoritmos de agrupamento de dados. *Revista ComInG-Communications and Innovations Gazette*, v. 1, n. 1, p. 34–45, 2016. Citado na página 26.

LADEIRA, L. Z. et al. Serviço de sugestão de rotas seguras para veículos. In: SBC. *Anais do XXXVII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. [S.l.], 2019. p. 608–621. Citado na página 24.

LI, Q. et al. Mining user similarity based on location history. 2008. Citado na página 31.

LU, H. et al. Brain intelligence: Go beyond artificial intelligence. 2017. Acessado em 20 de janeiro de 2020. Citado na página 19.

MACEDO, C. M. d. *Aplicação de algoritmos de agrupamento para descoberta de padrões de defeito em software JavaScript*. Tese (Doutorado) — Universidade de São Paulo, 2018. Citado na página 25.

MACEDO, E. E. F. D.; MACEDO, M. E. E. F. D. Uma abordagem sobre o nível de aceitação da carona solidária como medida sustentável na cidade de joão pessoa. Universidade Federal da Paraíba, 2019. Citado na página 40.

MARATHE, M. et al. *A Survey of clustering algorithms for similarity search*. 2017. Http://acadpubl.eu/jsi/2017-114-7-ICPCIT-2017/articles/12/36.pdf. Citado na página 22.

MCINNES, L.; HEALY, J.; ASTELS, S. *How HDBSCAN Works*. 2016. Disponível em: <https://hdbscan.readthedocs.io/en/latest/how_hdbscan_works.html>. Citado na página 27.

MONTEIRO, C.; SILVA, F. da; MURTA, C. Pré-processamento e análise de dados de táxis. In: *Anais do XLIV Seminário Integrado de Software e Hardware*. Porto Alegre, RS, Brasil: SBC, 2017. ISSN 2595-6205. Disponível em: <https://sol.sbc.org.br/index-.php/semish/article/view/3368>. Citado na página 36.

NING, S.; YAN, M. Discussion on research and development of artificial intelligence. 2019. Citado na página 19.

NORONHA, C. A. F. de; FERREIRA, K. R.; QUEIROZ, G. R. de. Web service para geocodificação de endereços em bancos de dados espaço-temporais. 2017. Citado na página 34.

OLIVEIRA, S. F. C. de; SILVA, C. A. U. da; SILVA, M. Y. de O. O uso do instagram e do twitter para identificação e mapeamento de usos do solo em áreas urbanas: o caso de fortaleza-ce/the use of instagram and twitter for identification and mapping of soil uses in urban areas: the case of fortaleza-ce. *Brazilian Journal of Development*, v. 5, n. 9, p. 16128–16149, 2019. Citado na página 24.

OPAS/OMS, O. M. d. S. *Folha informativa - Acidentes de trânsito*. 2019. Https://www.paho.org/bra/index.php?option=com_contentview=articleid=5147:acidentes-de-transito-folha-informativaItemid=779. Acessado em: 1 de junho de 2019. Citado na página 15.

PANDEY, G.; MCBRIDE, J. R.; EUSTICE, R. M. Ford campus vision and lidar data set. *The International Journal of Robotics Research*, SAGE Publications Sage UK: London, England, v. 30, n. 13, p. 1543–1552, 2011. Citado na página 36.

RASCHKA, S.; MIRJALILI, V. *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*. [S.l.]: Packt Publishing Ltd, 2019. Citado 4 vezes nas páginas 20, 21, 22 e 24.

RESENDE, P. R. d.; CUZZUOL, V. M. A. Rideuff: desenvolvimento de aplicativo de carona solidária na universidade federal fluminense. Universidade Federal Fluminense. Citado na página 40.

RODRÍGUEZ, J. et al. Cluster validation using an ensemble of supervised classifiers. *Knowledge-Based Systems*, Elsevier, v. 145, p. 134–144, 2018. Citado na página 26.

SANTOS, J. A. d. *Algoritmos rápidos para estimativas de densidade hierárquicas e suas aplicações em mineração de dados*. Tese (Doutorado) — Universidade de São Paulo, 2018. Citado na página 25.

SANTOS, M. D. S. d. *Déficit da mobilidade urbana: lacunas do planejamento a nível nacional*. 2019. Http://repositorio.enap.gov.br/handle/1/3992. Citado na página 18.

SAXENA, A. et al. A review of clustering techniques and developments. *Neurocomputing*, Elsevier, v. 267, p. 664–681, 2017. Citado 2 vezes nas páginas 22 e 23.

SEIDEL, E. J. et al. Comparação entre o método ward e o método k-médias no agrupamento de produtores de leite. *Ciência e Natura*, v. 30, n. 1, p. 07–15, 2008. Citado na página 24.

SILVA, A. C. S. Análise de uso de banco de dados por sistemas de automação industrial. 2018. Citado na página 36.

SIMÕES, J. P. F. *Análise de dados e Machine Learning na mobilidade urbana*. Tese (Doutorado), 2019. Citado 2 vezes nas páginas 26 e 40.

SINDIPEçAS. *Relatório da Frota Circulante*. 2019. Https://www.sindipecas.org.br/area-atuacao/?co=sa=frota-circulante. Acessado em: 14 de dezembro de 2019. Citado na página 14.

SOUSA, M. C. D. Observatório da mobilidade: Plano de ação integrado para cidades seguras, inclusivas e democráticas. 2018. Citado 2 vezes nas páginas 14 e 16.

SOUZA, K. R. d. *Estimativas de emissões de gases poluentes por veículos automotores rodoviários nos municípios paulistas e sua relação com a saúde*. 2017. Http://www.teses.usp.br/teses/disponiveis/11/11132/tde-15032018-101122/pt-br.php. Citado na página 15.

STANFORD. *K Means*. 2013. Disponível em: <https://stanford.edu/\~cpiech/cs221-/handouts/kmeans.html>. Citado na página 23.

STATISTA. *Number of cars sold worldwide from 1990 to 2019 (in million units)*. 2019. Https://www.statista.com/statistics/200002/international-car-sales-since-1990/. Acessado em: 1 de junho de 2019. Citado na página 14.

STATISTA. *Number of motor vehicles registered in the United States from 1990 to 2017*. 2019. Https://www.statista.com/statistics/183505/number-of-vehicles-in-the-united-states-since-1990/. Acessado em: 1 de junho de 2019. Citado na página 14.

UBER. *Descubra o que é e como usar o Uber Juntos (antigo Uber Pool)*. 2018. Https://www.uber.com/pt-BR/blog/o-que-uber-pool/. Acessado em: 29 de agosto de 2019. Citado 2 vezes nas páginas 16 e 29.

WANG, W.-T. et al. Adaptive density-based spatial clustering of applications with noise (dbscan) according to data. In: IEEE. *2015 International Conference on Machine Learning and Cybernetics (ICMLC)*. [S.l.], 2015. v. 1, p. 445–451. Citado na página 24.

WUNDER. 2019. Https://www.wundermobility.com/carpool. Acessado em: 29 de agosto de 2019. Citado na página 32.

ZUMPY. 2019. Http://zumpy.com.br. Acessado em: 29 de agosto de 2019. Citado na página 32.