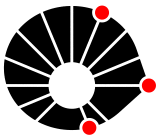
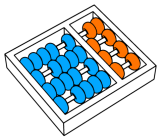




Análise de dados de geolocalização do veículo circular interno da Unicamp

L. A. Melo *L. F. Gonzalez* *J. F. Borin*

Relatório Técnico - IC-PFG-19-23
Projeto Final de Graduação
2019 - Julho

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

Análise de dados de geolocalização do veículo circular interno da Unicamp

Leonardo Alves de Melo* Luis Fernando Gonzalez † Juliana Freitag Borin ‡

Resumo

Este trabalho destinou-se a propor e analisar algoritmos que inferem em tempo real trajetos do veículo circular interno da Unicamp de modo a minimizar a possibilidade de falha humana ao se indicar manualmente qual itinerário está sendo realizado pelo motorista. O melhor algoritmo investigado alcançou 96% de acerto entre os dados de caminhos completos e com ele foi possível propor uma melhoria para o sistema.

1 Introdução

Uma das aplicações esperadas para a Internet das Coisas é a gestão e manutenção inteligente de linhas de ônibus públicos. Neste contexto, uma análise detalhada da geolocalização de veículos que realizam um determinado trajeto permite melhorarias na experiência dos seus usuários. Tomando como exemplo a pesquisa de Gong et. al. [1], este tipo de análise permitiu a predição do tempo de chegada de ônibus em cada ponto utilizando como base dados do histórico das leituras de dispositivos GPS presentes em cada veículo. Os pesquisadores desenvolveram um modelo híbrido de predição o qual foi testado em Shenyang, China, e obteve resultados melhores do que a tabela predefinida de horários.

Outra aplicação desenvolvida por Chen et. al. [2] utilizou os dados de localização de táxis para identificar padrões de mobilidade e propor novas rotas de ônibus noturnos. O projeto consistiu em agrupar as principais áreas em que os táxis costumam pegar e deixar os clientes, criando assim candidatos para serem possíveis pontos para as novas rotas de ônibus, de modo que os pontos iniciais correspondem onde os taxistas mais pegam os clientes, e os finais onde mais os deixam. Uma vez tendo os pontos, foi aplicado um algoritmo para gerar grafos que correspondem aos melhores percursos a serem feitos por esses ônibus, dos quais os piores candidatos a pontos eram retirados com bases em restrições predefinidas. Foi possível demonstrar que as rotas propostas se saíram melhor do que as recentemente criadas em Hangzhou, China.

No intuito de melhorar a previsão de tráfego e tempo de viagem, a pesquisa de Simmons et. al. [3] conseguiu prever com 98% de certeza o trajeto que um determinado motorista irá realizar tomando como base suas viagens passadas. Foi utilizado um algoritmo de Modelo

*Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP.
†Konker Labs
‡Instituto de Computação, Universidade Estadual de Campinas, 13081-970 Campinas, SP.

Escondido de Markov para estruturar as rotas de chegadas dos motoristas usando dados de GPS dos carros. Com o modelo foi possível fazer essa predição em tempo real do destino dos motoristas e otimizar o planejamento das viagens, sugerindo caminhos que evitem muito tráfego.

Neste projeto foi tratado o sistema que gerencia os ônibus circulares internos da Unicamp e como a proposta de algoritmos de análise de suas posições pode melhorar a experiência de seus usuários e a gestão do transporte interno da Universidade, reduzindo o número de falhas humanas, estendendo os trabalhos do *Smart Campus* [4].

2 Justificativa

Para poder comportar todos os mais de 31 mil alunos do campus de Barão Geraldo, a Universidade Estadual de Campinas oferece um serviço de transporte conhecido como "Veículo Circular Interno", o qual é composto por ônibus que fazem diferentes trajetos dentro da universidade. Como forma de melhorar esse serviço, a iniciativa *Smart Campus*, a qual compreende um conjunto de projetos que visam implementar o conceito de Internet das Coisas na Unicamp, propôs colocar sensores GPS em cada veículo, de modo a coletar seus dados de coordenadas geográficas e enviar em tempo real para um servidor, permitindo que os usuários do serviço pudessem visualizar o ônibus em tempo real por meio do aplicativo "Unicamp Serviços".

Atualmente, o motorista do ônibus deve apertar um botão no aparelho que contém o GPS para identificar qual o trajeto que ele realizará. Eventualmente pode ocorrer do motorista apertar o botão errado ou simplesmente esquecer de apertá-lo, gerando uma inconsistência e fazendo com que usuários de uma determinada linha não possam encontrá-la por meio do aplicativo. Uma imagem representativa dessa inconsistência pode ser vista na Figura 1, onde o veículo, representado em amarelo, não se encontra dentro do trajeto esperado, representado em azul, decorrido de uma falha humana. Se fosse possível identificar programaticamente qual a rota que o transporte está realizando com base nas leituras instantâneas de sua posição esse tipo de problema não existiria.

3 Objetivos

Este projeto teve como objetivo melhorar a experiência dos usuários do veículo circular interno da Unicamp, desenvolvendo um algoritmo capaz de prever qual a rota que o ônibus se encaixa com base nas leituras instantâneas de sua posição e com este algoritmo propor modos de se minimizar a falha humana.

4 Desenvolvimento do Trabalho

O desenvolvimento do trabalho consistiu em três etapas: na primeira, as características do sistema de ônibus da Unicamp foram estudadas; na segunda, foi construída uma base de dados com todos os caminhos realizados por todos os circulares em um determinado período

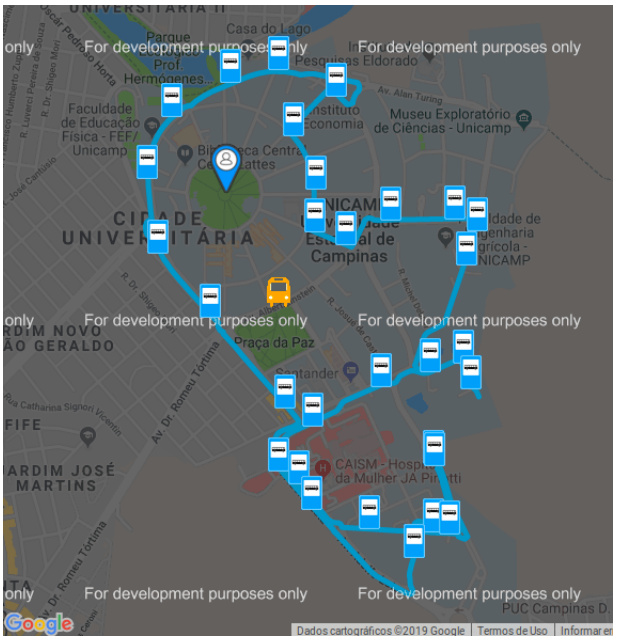


Figura 1: Foto de um circular realizando o trajeto incorreto

de tempo; na terceira, foram investigados diversos algoritmos para tentar inferir a rota com base nas leituras. As três etapas são discutidas com maiores detalhes a seguir.

4.1 Etapa 1: Entendendo o Problema

Existem atualmente 4 linhas que atendem o campus de Barão Geraldo: *Circular 1*, *Circular 2 FEC*, *Circular 2 Museu* e *Circular Noturno*, em que todas começam e terminam no terminal de ônibus da Unicamp, como pode ser visto em suas trajetórias na Figura 2, em que o marcador vermelho representa o terminal de ônibus. Cada veículo possui um dispositivo composto por um módulo GSM GPS, um sensor de temperatura, um microcontrolador e botões de configuração, e foi implantado para o projeto do *Circulino* [5]. Os dispositivos não são dispostos de forma unívoca nos ônibus e nem mesmo nas rotas. Dessa forma, um dispositivo que fez uma rota Circular 1 dentro do Ônibus A, pode, muitas vezes dentro do mesmo dia, fazer a rota Circular 2 no Ônibus A ou mesmo no Ônibus B, ou seja, se um motorista esquecer de trocar a configuração do dispositivo que indica a linha entre um trajeto e outro, a exibição das informações do ônibus para o usuário ficará inconsistente.

O servidor que recebe as informações dos dispositivos dos circulares contém uma base de dados com uma lista de 9.829.780 entidades JSONs, as quais representam cada leitura instantânea de um dispositivo de um circular coletado entre 11 de Junho de 2018 e 18 de Abril de 2019. Essa base de dados foi utilizada neste projeto e um exemplo de uma entidade JSON dela pode ser visto na Figura 3, onde cada campo é explicado abaixo.

- *timestamp*: Data e hora em que a informação do dispositivo chegou no banco de dados.

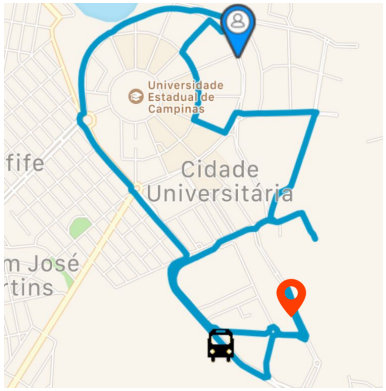
- *payload*
  - *numero_satelites*: Número de satélites vistos pelo módulo GPS no momento.
  - *hora_coletado*: Data e hora de coleta baseado no relógio do dispositivo.
  - *_lat*: Dado de latitude do módulo GPS.
  - *_pdop*: Dado de pdop do módulo GPS.
  - *_elev*: Dado de altitude do dispositivo.
  - *id_linha*: Informação da linha configurada pelo motorista.
  - *velocidade_media_gps*: Dado da velocidade média.
  - *temperatura*: Dado da temperatura do dispositivo.
  - *_lon*: Dado de longitude do módulo GPS.
- *ingestedTimestamp*: Data e hora em que a informação do dispositivo chegou no banco de dados.
- *incoming*
  - *deviceGuid*: Identificador único do dispositivo.
  - *channel*: Canal de envio de dados no servidor.
- *geolocation*
  - *lat*: Dado de latitude do módulo GPS.
  - *lon*: Dado de longitude do módulo GPS.
  - *elev*: Dado de altitude.

4.2 Etapa 2: Construindo a Base de Dados de Caminhos

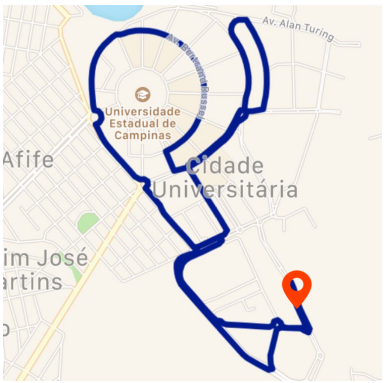
A partir da base de dados estudada na etapa 1, foi realizado um pré-processamento dos dados de modo a agrupar as entidades em caminhos. O conceito de "caminho"para este projeto é definido como sendo o conjunto de leituras enviadas em sequência e ininterruptamente por um determinado dispositivo em que somente a primeira e a última leitura se encontram próximas em relação ao ponto do terminal de ônibus.

O algoritmo de pré-processamento levou em conta que as entidades já se encontravam agrupadas por *deviceGuid* e ordenadas por *hora_coletado* no banco de dados e pode ser visto no Algoritmo 1. Ao final, é esperado que o caminho gerado comece a 100 metros de proximidade do terminal, se afaste dele ao realizar a trajetória e termine a 100 metros de proximidade dele. O número 100 foi escolhido para englobar toda a região da rua em que os ônibus costumam ficar estacionados em fila, de modo que um caminho não comece enquanto o veículo se mantiver nessa rua.

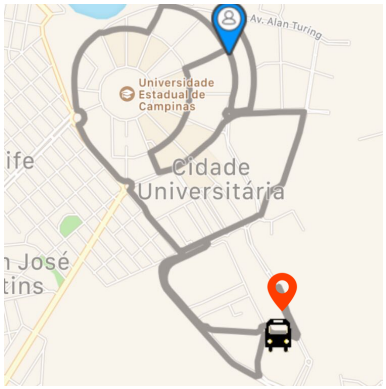
Um exemplo de um caminho gerado por esse pré-processamento pode ser visto na Figura 4, onde é possível ver que os pontos referenciam o trajeto do *Circular 2 Museu*, em que



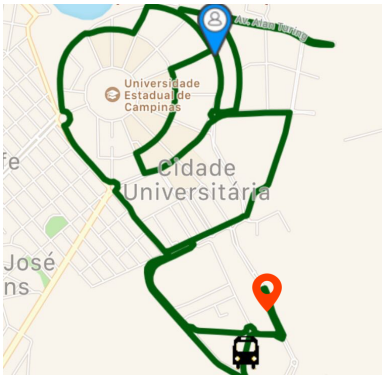
(a) Circular 1



(b) Circular Noturno



(c) Circular 2 FEC



(d) Circular 2 Museu

Figura 2: Trajetórias dos circulares

```json
{
  "timestamp": "2018-06-11T17:32:01.844Z",
  "payload": {
    "numero_satelites": 12,
    "hora_coletado": "2018-06-11 14:31:55",
    "_lat": -22.829637,
    "_pdop": 1.5,
    "_elev": 647.6,
    "id_linha": 2,
    "velocidade_media_gps": 0.02,
    "temperatura": 42.03,
    "_lon": -47.060962
  },
  "ingestedTimestamp": "2018-06-11T17:32:01.844Z",
  "incoming": {
    "deviceGuid": "6ce968a1-a32b-4f8c-bc39-4463f50f4591",
    "channel": "info"
  },
  "geolocation": {
    "lat": -22.829637,
    "lon": -47.060962,
    "elev": 647.6
  }
}
```


Figura 3: Exemplo de entidade JSON presente no Banco de Dados

Algorithm 1 Pré-processamento de dados para geração de caminhos

```
function CriaCaminhos(baseDeDados[])
    caminhos ← [ ]
    N ← length(baseDeDados)
    estado ← iniciando
    for i ← 1 to N do
        leitura ← baseDeDados[i]
        if estado == iniciando then
            novoCaminho ← [ ]
            if distanciaEntre(leitura, terminalDeOnibus) > 100 then
                estado ← percorrendo
                novoCaminho.append(leitura)
        else if estado == percorrendo then
            novoCaminho.append(leitura)
            if distanciaEntre(leitura, terminalDeOnibus) < 100 then
                estado ← iniciando
                caminhos.append(novoCaminho)
    return caminhos
```

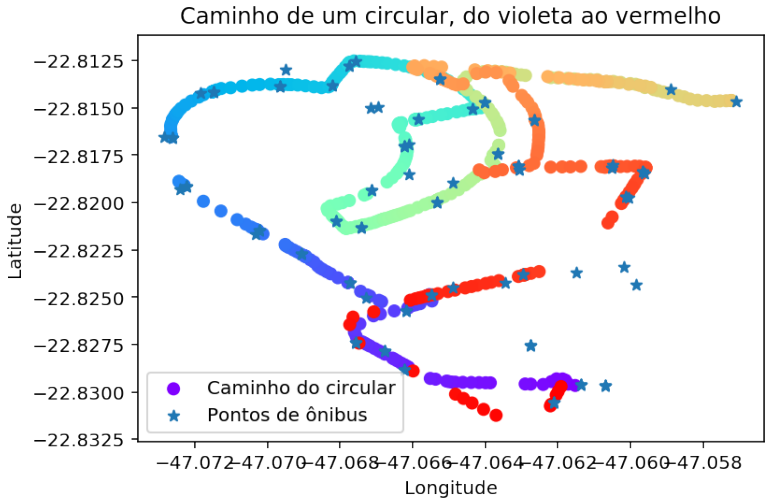


Figura 4: Exemplo de caminho do circular. O caminho foi reconstruído a partir dos dados enviados pelo ônibus. A localização dos pontos de ônibus foi obtida junto com o setor de transporte da Universidade

o ônibus inicia sua trajetória no ponto mais violeta e termina no ponto mais avermelhado passando por todo espectro de cores. Uma vez tendo todos os caminhos bem definidos, foi possível obter um arquivo *CSV* com os campos de latitude, longitude, id da linha, data e hora de coleta e índice do caminho, em que linhas que tenham o mesmo índice do caminho pertencem ao mesmo caminho, e o id da linha representa qual informação da rota foi configurada pelo motorista.

4.3 Etapa 3: Algoritmos para análise dos dados de geolocalização

Foram propostos e testados ao total quatro algoritmos, em que um é programado explicitamente e três são de aprendizado de máquina. Todos podem ser vistos a seguir.

4.3.1 Algoritmo de Georreferência

O primeiro algoritmo programado explicitamente foi chamado de *Algoritmo de Georreferência* e avalia cada uma das coordenadas de um dado caminho com pontos específicos definidos em que os trajetos dos circulares passam. Os pontos podem ser vistos na Figura 5, em que o ponto em vermelho corresponde ao ponto em que apenas o *Circular 2 Museu* passa e foi denominado *pontoCircularMuseu*, o ponto em azul corresponde ao ponto em que o *Circular 2 Museu*, o *Circular 2 FEC* e o *Circular 1* passam e foi chamado de *pontoCircularFec*, e o ponto verde corresponde ao ponto em que apenas o *Circular 1* passa e foi denominado *pontoCircular1*, como pode ser deduzido ao comparar com a Figura 2.

Uma vez tendo os pontos definidos, foi possível propor um algoritmo que pode ser visto no Algoritmo 2, em que os valores usados para as distâncias foram obtidos com base em uma estimativa da proximidade que um circular deveria passar por um ponto para ser

Figura 5: Pontos utilizados no *Algoritmo de Georreferência*

considerado de uma determinada linha. Desse modo é possível por eliminação determinar qual é o trajeto que o ônibus está realizando.

Algorithm 2 Georreferência

```
1: function TestaAlgoritmoGeorreferência(Caminho[])
2:     if menorDistanciaEntre(caminho, pontoCircularMuseu) < 300 then
3:         return circularMuseu
4:     if menorDistanciaEntre(caminho, pontoCircular1) < 250 then
5:         return circular1
6:     if menorDistanciaEntre(caminho, pontoCircularFec) < 250 then
7:         return circularFec
       return circularNoturno
```

4.3.2 Algoritmos de Aprendizado de Máquina

Foram investigados os algoritmos de *Naïve Bayes* [6], Rede Neural [7] e *Random Forest* [8]. O primeiro é baseado no *Teorema de Bayes* e analisa cada *feature* independentemente, o segundo é composto por nós interconectados que operam funções complexas entre as *features* e o terceiro tenta encontrar a melhor combinação de condicionais entre as *features* que leve ao resultado ótimo. Levando em conta que todos esses algoritmos exigem uma entrada fixa, ou seja, a quantidade de *features* de entrada deve ser igual para todos os caminhos testados,

o que não é verdade porque cada caminho tem um número diferente de leituras, foi aplicado dois novos pré-processamentos dos dados para normalizar isso. O primeiro é chamado de subamostragem e o método utilizado consiste em transformar todos os valores dentro de uma janela de tempo em um único valor, correspondendo a média destes. A subamostragem foi feita para transformar a taxa de aproximadamente 0,5Hz de aquisição de dados em valores espaçados de 10 segundos, contendo o valor médio de latitude, longitude, e hora. No caso, foi escolhida uma janela de dez segundos e um passo também de dez segundos porque esse é o tempo estimado de uma parada do ônibus no ponto. Tomando como base que um trajeto do circular demora entre 20 a 30 minutos para ser concluído, então após aplicar o primeiro pré-processamento espera-se obter caminhos de quantidade de leituras entre 120 e 180, a partir disso o segundo pré-processamento foi proposto: a discretização do comprimento total do percurso em 60 partes de iguais comprimentos e obtendo dessa forma o valor médio das variáveis estudadas para cada uma dessas regiões. O número 60 foi escolhido por ser metade do mínimo esperado de quantidade de leituras em um caminho, de modo que em um caminho normal haverá entre 2 e 3 leituras agrupadas por cada grupo aglutinado, o que não representa grande perda de informação e minimiza o uso de dados e tempo de processamento.

4.3.3 Treinos e Testes dos Algoritmos

Levando em conta que um algoritmo buscado precisa ter capacidade de inferência da rota do ônibus em tempo real durante o trajeto e não apenas no término, foi necessário aumentar a base de dados de caminhos de modo a simular subcaminhos, que são definidos como sendo o conjunto das i primeiras leituras de um caminho. Assim sendo, dado um caminho C completo de n leituras, criou-se um novo subcaminho apenas com a primeira leitura de C, outro apenas com a primeira e segunda leituras de C, e assim sucessivamente até obter n-1 novos subcaminhos, o que simularia C sendo percorrido. Todos os caminhos e subcaminhos foram então separados em 80% para treino e 20% para teste, em que para os algoritmos de *Random Forest* e Rede Neural dividiu-se a base de treino em 80% para treino de parâmetros e 20% para validação cruzada, de modo a encontrar os melhores parâmetros maximizando a porcentagem de acerto. Uma vez tendo os melhores parâmetros, os resultados foram testados na base de dados de teste e comparados com seus id linha correspondentes.

5 Resultados

Um gráfico comparativo com os resultados de cada algoritmo pode ser visto na Figura 6. Percebe-se que o algoritmo de *Random Forest* foi o que obteve o melhor resultado, alcançando uma média de 96% de acerto para os caminhos completos, e ficando acima de todos os outros em todo o alcance do gráfico. É esperado que a *Random Forest* se saísse melhor porque este algoritmo trabalha encontrando a melhor combinação possível de condicionais relacionando as *features* que otimizem a porcentagem de acerto, ou seja, o *Algoritmo de Georreferência* é um subconjunto dessa combinação de condicionais, o que satisfaz ele estar inferior ao primeiro. A Rede Neural apresentou um resultado fraco porque provavelmente não conseguiu convergir para a quantidade de neurônios testada em cada camada,

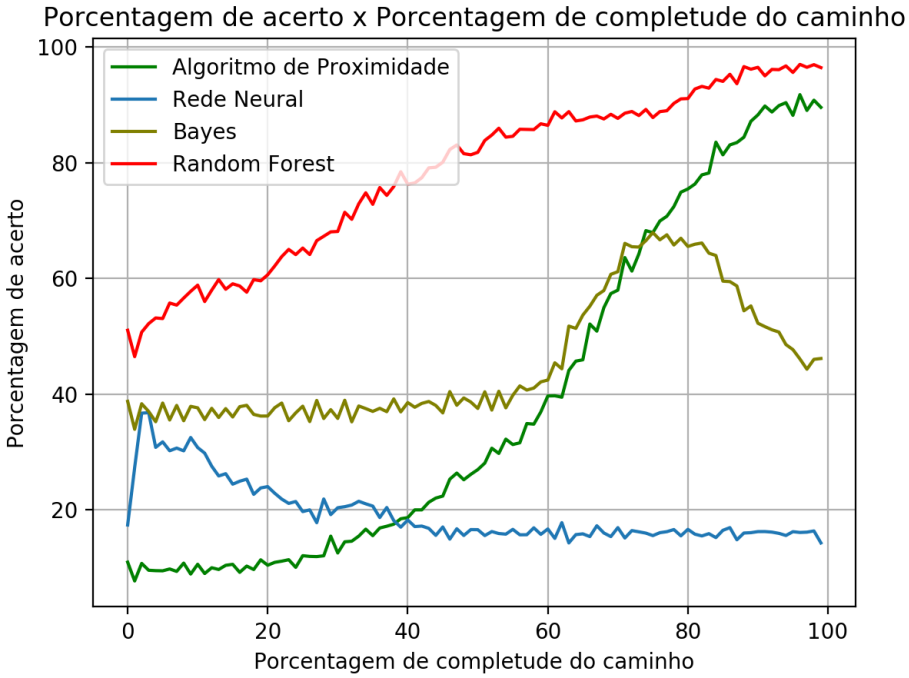


Figura 6: Gráfico comparativo com os algoritmos trabalhados em porcentagem de completude

necessitando de mais testes com mais variadas quantidades de camadas e neurônios. Era esperado que o algoritmo de *Naïve Bayes* melhorasse seu acerto com uma maior porcentagem de completude do caminho, mas esse algoritmo apresentou uma queda brusca ao final. As previsões equivocadas de cada algoritmo podem ser explicadas pela imprecisão dos valores do GPS, que podem variar por metros, e pela presença de caminhos com configurações erradas no banco de dados, os quais atrapalham o treino e a checagem dos resultados.

Como a porcentagem de acerto para os 25% iniciais do melhor algoritmo é abaixo de 70%, ainda não é possível propor substituir completamente a configuração manual do trajeto do circular, porém já é possível propor um tipo de sistema que sugere ao motorista rever o ajuste que ele havia colocado caso o algoritmo tenha um grau de certeza maior que um certo limiar de que a configuração possa estar equivocada, podendo assim reduzir o número de falhas humanas.

6 Conclusões

Neste projeto foi possível investigar as características dos dados de geolocalização dos veículos dos circulares internos da Unicamp, bem como propor um algoritmo que alerte ao motorista em tempo real que provavelmente o percurso realizado difere do trajeto configurado no dispositivo do ônibus, podendo assim reduzir o número de falhas humanas.

É esperado que como continuação deste projeto novos algoritmos sejam testados e que a porcentagem de acerto possa ser tal que o ajuste do dispositivo seja automática.

7 Agradecimentos

Agradecemos o suporte da equipe do projeto *Circulino* [5], que integra o projeto *Smart Campus Unicamp* [4].

Referências

[1] J. Gong, M. Liu and S. Zhang, "Hybrid dynamic prediction model of bus arrival time based on weighted of historical and real-time GPS data" *2013 25th Chinese Control and Decision Conference (CCDC)*, Guiyang, 2013, pp. 972-976.

[2] C. Chen, D. Zhang, N. Li and Z. Zhou, "B-Planner: Planning Bidirectional Night Bus Routes Using Large-Scale Taxi GPS Traces", in *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 4, pp. 1451-1465, Aug. 2014.

[3] R. Simmons, B. Browning, Yilu Zhang and V. Sadekar, "Learning to Predict Driver Route and Destination Intent", *2006 IEEE Intelligent Transportation Systems Conference*, Toronto, Ont., 2006, pp. 127-132.

[4] SMART Campus Unicamp - Internet Das Coisas. **Smart Campus**, 2018. Disponível em: <http://smartcampus.prefeitura.unicamp.br>. Acesso em: 30, Abril de 2019.

[5] Barbosa R.A., Sousa R.P., Oliveira F.A., Oliveira H.C., Luz P.D.G., Manera L.T. (2019) Circulino: An IoT Solution Applied in the University Transport Service. In: Iano Y., Arthur R., Saotome O., Vieira Estrela V., Loschi H. (eds) Proceedings of the 4th Brazilian Technology Symposium (BTSym'18). BTSym 2018. Smart Innovation, Systems and Technologies, vol 140. Springer, Cham

[6] Naive Bayes Classifier. **Naive Bayes Classifier**, 2019. Disponível em: <https://www.ic.unicamp.br/rocha/teaching/2011s2/mc906/aulas/naive-bayes-classifier.pdf>. Acesso em: 09, Julho de 2019.

[7] An Introduction to Neural Networks. **An Introduction to Neural Networks**, 2019. Disponível em: <https://www.inf.ed.ac.uk/teaching/courses/nlu/assets/reading/Gurney_et_al.pdf

[8] Random Forests. **Random Forests**, 2019. Disponível em: <https://www.stat.berkeley.edu/breiman/randomforest2001.pdf>. Acesso em: 09, Julho de 2019.