

Relatório final - Congestionamento veicular

**Diego Luiz da Silva, Ph.D.
Raphael Yokoingawa de Camargo, PhD.**

SUMÁRIO

Introdução	3
Objetivos do estudo e método de análise	4
O caso da cidade de São Paulo - Brasil	5
Waze API	8
Comparação CET e Waze	11
Protótipo de plataforma de dados	16
Instalação	20
Mapa da arquitetura de dados	20
Configurando de uma nova cidade no painel	22
Conclusão	26
Referências	27
Apêndice A	28

Introdução

A população urbana aumentou exponencialmente - de 751 milhões em 1950 para 4,2 bilhões em 2018 - e continuará essa tendência (ONU, 2018). O processo de urbanização gera impactos no modo como os indivíduos se relacionam com a cidade, sobretudo em como gerenciar de maneira apropriada a escalabilidade na provisão de serviços e circulação de bens para que o desenvolvimento seja sustentável.

O efeito deste aumento implica prestar atenção a muitos aspectos de sustentabilidade urbana, no entanto, enfatizamos o aspecto da mobilidade e a gestão de tráfego.

Um dos fenômenos presentes na maioria das cidades e megacidades de médio e grande porte em todo o mundo é o congestionamento veicular. Na União Européia estima-se que o custo do tempo perdido no trânsito foi equivalente a 1,4% do PIB da região (European Commission, 2020), enquanto que os Estados Unidos apontam perdas na ordem de 0,7% do PIB nacional (Cebr, 2014). Apesar de poucos estudos de congestionamento urbano na América Latina (AL) avaliando os impactos e os custos sociais, índices como o elaborado pela INRIX¹, mostrou que em 2019, das dez cidades mais congestionadas no mundo, três são latinoamericanas. O ranking do indicador de tráfego sulamericano elaborado pela empresa Tom Tom², elencou quatro cidades latinoamericanas como as mais congestionadas em 2019, Fig. 1.



a) Lima, Peru



b) Quito, Equador

Fig. 1 - Cidades como Lima, Quito, Bogotá, Recife, Cidade do México figuram como uma das mais congestionadas no mundo de acordo com rankings internacionais INRIX e Tom Tom.

Os determinantes de congestionamento nas cidades da AL podem ser de diversas naturezas, sendo no nível macro principalmente afetada pelo aumento da taxa de urbanização, a infraestrutura viária que favorece o transporte individual, a baixa acessibilidade e

¹ <https://inrix.com/scorecard/>

² https://www.tomtom.com/en_gb/traffic-index/ranking/

interoperabilidade do transporte público urbano, além dos desafios em financiar o seu desenvolvimento, (BID, 2021).

O fenômeno de congestionamento veicular pode ser recorrente ou não recorrente, sendo o rendimento medido principalmente pela observação da velocidade da via, fluxo, densidade, comprimento e duração da fila (BID, 2021). Geralmente, os resultados de monitoramento são gerados por inspeção visual de Circuito Fechado de TV (CFTV) ou em pontos de observação estratégicos. Com o advento de novas tecnologias digitais de tráfego urbano, as cidades podem contar com um abundante volume de dados em tempo *quasi-real*. Este atual cenário traz novas perspectivas de gestão e principalmente na tomada de decisão, seja no nível operacional, planejamento ou de políticas públicas.

Neste estudo desenvolvemos um protótipo de plataforma de dados provenientes da *Application Programming Interface* (API) do Waze para as cidades: São Paulo, Montevidéu, Quito, Xalapa e para o distrito de Miraflores em Lima. Os dados são provenientes de duas fontes principais: dados armazenados e pré-processados na nuvem da Amazon Web Services (AWS) e diretamente da API do Waze. O primeiro fornece dados para visualização da série histórica e o último em tempo *quasi-real*. Nós utilizamos como referência o método utilizado pela Companhia de Engenharia de Tráfego de São Paulo (CET) para validação do método proposto. O método utilizado pela CET foi escolhido pela disponibilidade de dados da série histórica do índice de congestionamento de São Paulo. Além disso, o método é invariante ao tipo de cidade, ou seja, a sua aplicabilidade pode ser facilmente estendida e desta forma possibilita a validação dos resultados e ajustes do modelo para as demais cidades.

Objetivos do estudo e método de análise

O objetivo geral da atividade é validar se os dados gerados pelo aplicativo de navegação colaborativa Waze são capazes de descrever o congestionamento para as cidades de São Paulo, Montevidéu, Quito, Xalapa e para o distrito de Miraflores em Lima.

Aplicamos métodos de análise exploratória de dados e observamos a correlação entre a curva real de tráfego e a curva obtida pelos dados do aplicativo. Para a curva real utilizamos os dados de observação utilizados pela Companhia de Engenharia de Tráfego de São Paulo. A escolha é devido a maturidade do modelo de gestão de tráfego em São Paulo, a validação com analistas experientes que acompanham o indicador de lentidão, a disponibilidade de dados históricos para comparação, e a familiaridade com os dados do aplicativo Waze já em uso na Companhia em outros serviços de monitoramento. O método da CET é invariante quanto ao tipo de cidade, podendo ser aplicado ou utilizado como validação nas demais cidades de nosso estudo.

No caso dos dados serem minimamente satisfatórios em termos de qualidade e significativa representação da condição de tráfego, a etapa seguinte é o desenvolvimento de um protótipo para gestão de tráfego das cidades. O protótipo possui as seguintes premissas:

- Usabilidade e foco na experiência do usuário;
- Código aberto;
- Interface que proporcione o diagnóstico do tráfego utilizando os dados históricos e tempo real;
- Portátil;
- Extensível;
- Adaptável;
- Simples execução e manutenção;
- Escalabilidade da monitoração;
- Baixo custo.

O protótipo não representa um produto final, pois dependerá da infra-estrutura que cada cidade disponibiliza em ambiente de produção e adaptações serão necessárias. Detalhamos as principais adaptações nos capítulos seguintes.

O caso da cidade de São Paulo - Brasil

A medição do congestionamento no trânsito da cidade de São Paulo teve início em meados de 1980, sendo um importante indicador de desempenho, que registra nos dias úteis, no horário entre 07 e 20 horas, sua extensão nas vias monitoradas pela Companhia de Engenharia de Tráfego de São Paulo (CET-SP). Este indicador é amplamente divulgado por órgãos da imprensa e consolidou-se como uma referência dos padrões da qualidade do trânsito na cidade, sendo utilizado também para a elaboração de diversos estudos técnicos e estatísticos.



c) Agente de trânsito monitorando a via



d) Centro de Monitoramento

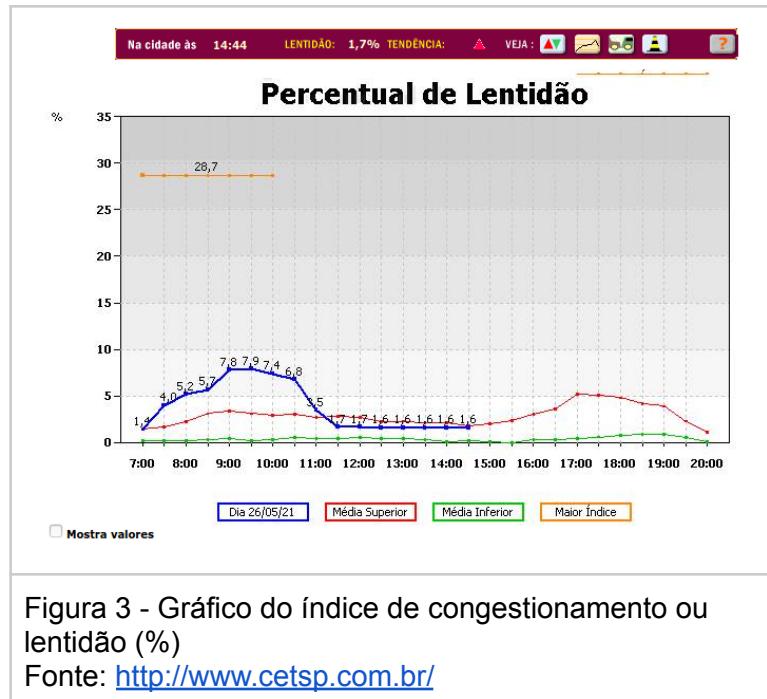
Fig. 2 - Inspeção do trânsito na cidade de São Paulo

Inicialmente as informações eram coletadas por inspeção visual de funcionários alocados no topo de altos edifícios estrategicamente localizados na cidade, também conhecido por Postos Avançados de Campo (PAC), Fig. 2. Atualmente, a coleta é realizada por meio de sistema de câmeras de Circuito Fechado de Televisão (CFTV) e as imagens são transmitidas para as Centrais de Monitoramento, onde são realizadas análises e o envio de orientações aos agentes operacionais.

Até meados de 2007 o índice de congestionamento do trânsito era representado em quilômetros de vias monitoradas, posteriormente a CET passou a apresentar o indicador no formato percentual, representado pela relação entre a extensão em quilômetros de vias congestionadas e o total de quilômetros monitorados que atualmente é de 868 km.

$$\text{Indice de congestionamento} = \frac{\text{Vias congestionadas}}{\text{Vias monitoradas}} \times 100 \quad \text{Eq. (1)}$$

De acordo com a CET-SP³, o sistema viário monitorado é contabilizado considerando a dupla mão de circulação, nas vias de sentido único ou pistas segregadas com canteiro no mesmo sentido de circulação, por exemplo, pista local/central/expressa das Marginais, contabiliza-se a extensão total de cada pista.



³ <http://www.cetsp.com.br/transito-agora/mapa-de-fluidez.aspx>

O indicador calculado a cada 30 minutos é comparado com a série histórica do mesmo dia nos 12 meses anteriores, Fig. 3. No cálculo estão descartados os valores dos meses de janeiro, fevereiro, julho e dezembro, feriados e emendas de feriados e valores cuja diferença em relação ao valor médio ultrapassa 1,5 desvio padrão. Os limites inferior e superior correspondem a um desvio padrão das amostras selecionadas e representa o intervalo de valores do indicador considerados aceitáveis, quando não há ocorrência de grande impacto no trânsito.

A CET-SP também disponibiliza as informações de trânsito em sua *homepage* no formato de mapas, tabelas e gráficos, Fig. 4. A informação de lentidão do tráfego é detalhada por região da cidade—Norte, Sul, Leste, Oeste e Centro— além de informações de incidentes nas vias.



Além disso, os dados de congestionamento fornecem a tendência do comportamento do tráfego comparando com o registro imediatamente anterior. O resultado da diferença entre os pares de dados consecutivos fornece a classificação de tendência alta, estável e baixa.

Waze API

O Waze é uma empresa de tecnologia que fornece o serviço de navegação colaborativa. Além de expor informações em tempo real das condições de trânsito nas cidades, também fornece informações de status dos trechos de via com a classificação de congestionamento, incidentes, acidentes graves e sugestões de rotas, Fig. 5.



Figura 5 - Waze Map e indicação das vias congestionadas, bloqueios de vias e alertas.

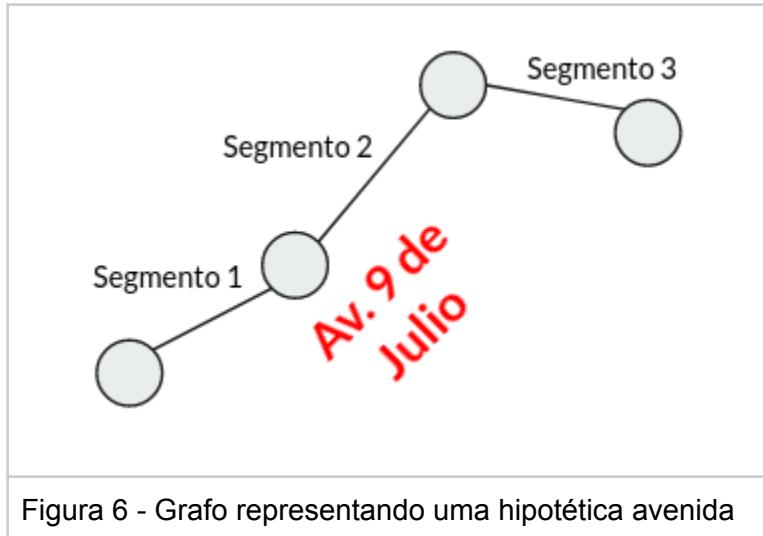
Por ser uma ferramenta colaborativa, onde os usuários compartilham seus dados de geolocalização e notificam os alertas de incidentes, a qualidade dos dados está diretamente relacionada ao volume de usuários cadastrados na aplicação. Em 2019, São Paulo possuía aproximadamente 4,5 milhões de usuários ativos.⁴ A impressionante cifra foi um dos motivos da escolha da cidade na comparação e validação dos dados fornecidos pela API do Waze.

As informações coletadas são adicionadas nas camadas tendo como base o mapa viário da cidade. Cabe ressaltar que o monitoramento do Waze é feito em todas as vias da cidade. O mapa é representado por um objeto matemático chamado de grafo. O grafo é composto de vértices e arestas que se conectam, simulando o viário da cidade. Na Fig. 6, utilizamos uma hipotética avenida para identificar os nós representados pelas circunferências em cinza, e as

4

https://www.correiobrasiliense.com.br/app/noticia/economia/2019/03/18/internas_economia.743566/entre-vista-com-leandro-esposito-gerente-geral-do-waze-no-brasil.shtml

arestas, designando os segmentos. Os segmentos de uma mesma via podem possuir nomenclaturas diferentes, por exemplo, a Avenida Simón Bolívar, em Quito, possui os seguintes segmentos: Av. Simón Bolívar (Sur), Av. Simón Bolívar (Norte), Paso Deprimido Av. Simón Bolívar. Além disso, as vias podem ter segmentos paralelos, como é o caso das pistas da Marginal Tietê, em São Paulo.



Analizar os segmentos do mapa viário do Waze é fundamental para obter uma tabela de conversão da nomenclatura das vias. Desta forma cada cidade poderá decidir se quer incluir o conjunto dos segmentos ou apenas um trecho específico de via para monitoramento.

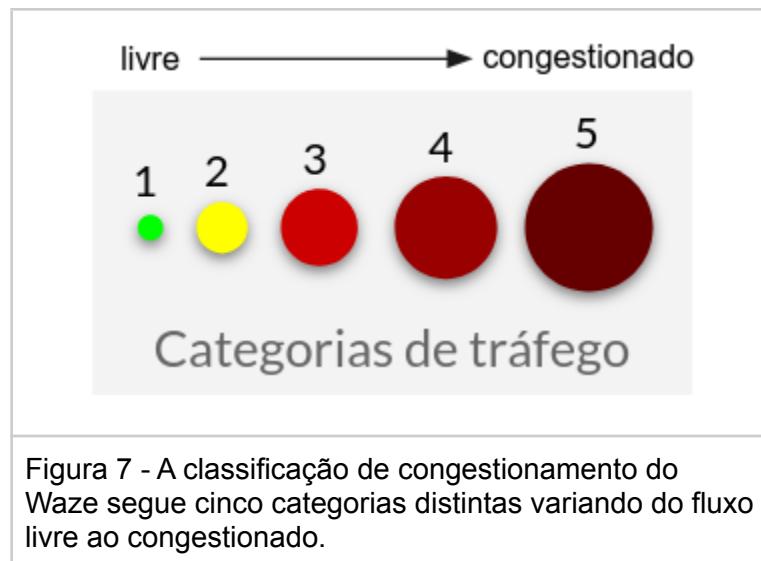
Para análise de congestionamento utilizamos apenas os dados armazenados pela Waze *Traffic jams* que consiste nas informações de desaceleração de tráfego com base na localização e velocidade dos usuários. Os campos do arquivo são populados e expostos pela API através de uma chamada utilizando o protocolo HTTP. Os campos utilizados na elaboração da prova de conceito estão identificados na Tabela 1.

Tabela 1 - Campos utilizados na comparação entre CET e Waze extraídos das informações geradas pela Waze *Traffic jams*

Nome	Tipo	Descrição
pubMillis	Timestamp	Unix time
line	Latitude e Longitude	Linestring
length	Integer	Comprimento em metros
street	string	Nome da via

city	string	Nome da cidade
country	string	Nome do país
level	Integer	Categoria de congestionamento (0= fluxo livre - 5= Bloqueado)

A extração dos dados foi realizada na tabela de dados disponibilizada pela Fundação Getúlio Vargas (FGV) que os armazena em uma nuvem dedicada para o projeto *Big Data* para o Desenvolvimento Urbano Sustentável⁵. Os dados representam uma fotografia do viário minuto a minuto, sendo os dados crus processados antes do armazenamento.



O campo `level` é essencial na comparação com os dados da CET, Fig. 7. O Waze distingue o congestionamento em cinco categorias:

- **Categoria 1:** Fluxo livre, os veículos se locomovem no limite da velocidade da via;
- **Categoria 2:** Tráfego leve, pequenas variações de velocidade, mas sem comprometimento do fluxo;
- **Categoria 3:** Tráfego moderado, iminência de congestionamento;
- **Categoria 4:** Tráfego pesado, redução de velocidade e atrasos significativos;
- **Categoria 5:** Tráfego parado, veículos com significativa redução de velocidade ou completamente parados.

⁵ <https://smartcities-bigdata.fgv.br/o-projeto>

Os campos `length`, `street` e `line` contribuem para detecção de segmentos únicos no grafo e evitar redundância no somatório do congestionamento.

Comparação CET e Waze

Para detectar o efeito do tráfego selecionamos uma sexta-feira, véspera de Carnaval, do ano de 2019 (01/03/2019). O motivo é por ser uma data de grande movimentação nas vias da cidade o que contribui na detecção dos picos de congestionamento e por não ter sido influenciada pela restrição da mobilidade urbana devido a pandemia de COVID-19. Além disso, de acordo com o Instituto Nacional de Meteorologia (INMET) não houve a eventos climáticos extremos, sendo observado apenas uma precipitação máxima de 11mm às 5:00am, temperatura média de 21°C, e umidade relativa média de 85%.

As categorias utilizadas pelo aplicativo Waze para classificação do tráfego foram divididas em dois grupos:

- **Grupo 1:** agrupa as categorias 3, 4 e 5;
- **Grupo 2:** agrupa as categorias 4 e 5 somente.

A categoria 3 na classificação do Waze representa uma zona de transição entre o tráfego livre e o intenso, portanto a inclusão desta categoria é para entender se a inspeção visual realizada pela CET está mais próxima da categoria 3 ou 4 na detecção da iminência de congestionamento.

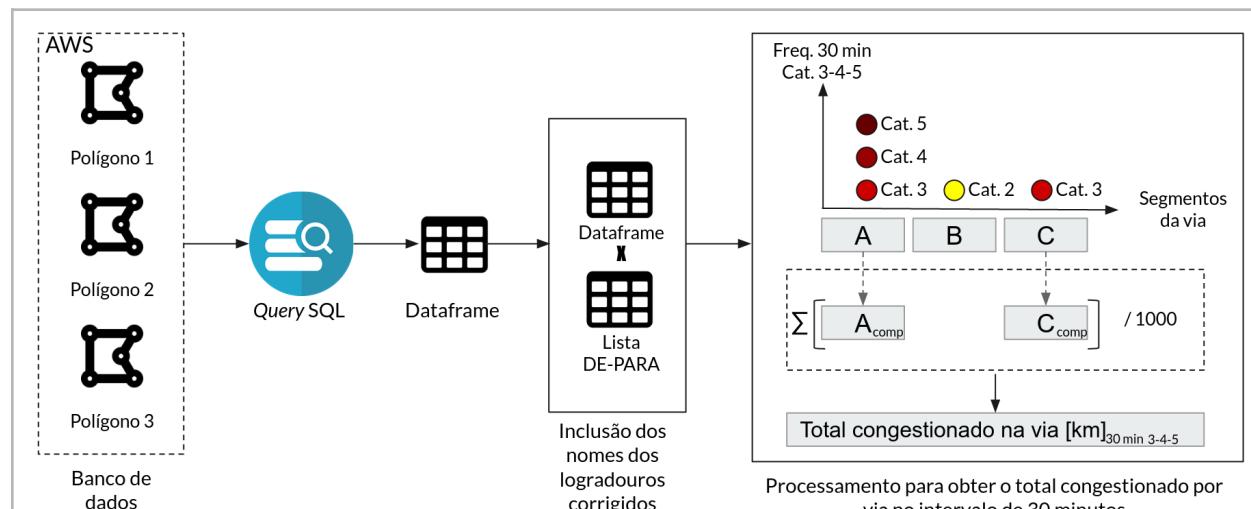
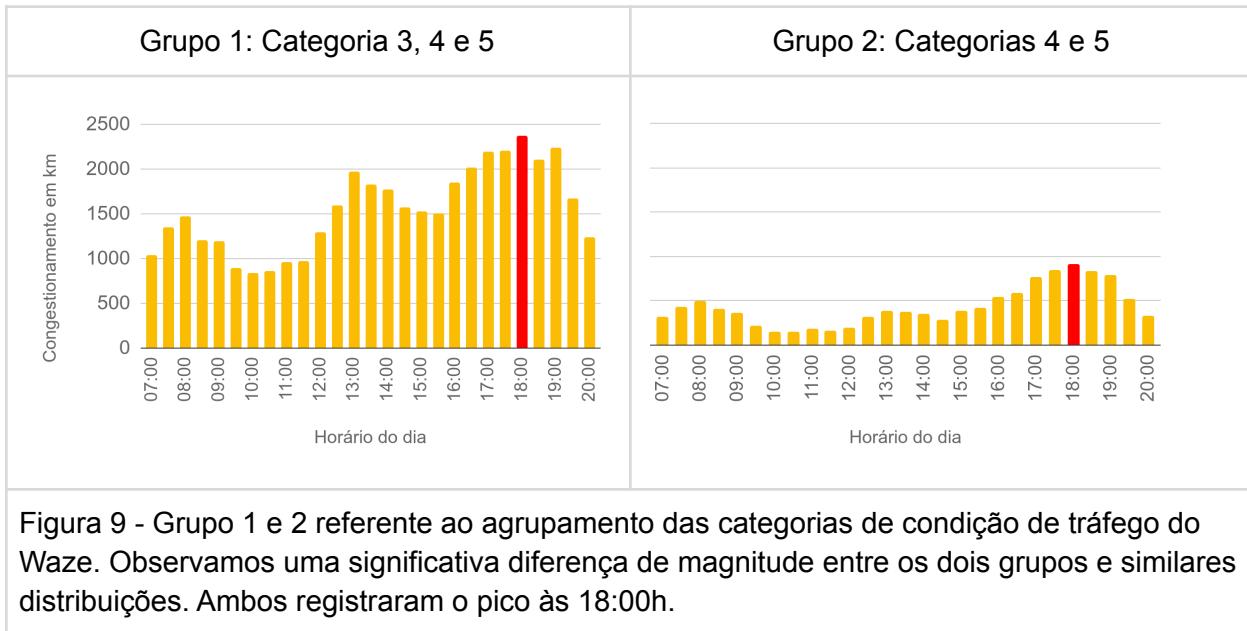


Figura 8 - Fluxo do processamento para obtenção do total congestionado por via na janela de 30 minutos. No exemplo, é mostrado o cálculo para o Grupo 1 (categorias: 3, 4 e 5). O Grupo 2 é obtido bastando excluir os resultados menores que a categoria 4. O total congestionado é convertido em quilômetros que corresponde ao comprimento da fila no intervalo de tempo.

Os dados coletados minuto-a-minuto do Waze foram agrupados no intervalo de 30 minutos seguindo o modelo utilizado pela CET, Fig. 8. As duplicidades de segmentos em cada intervalo foram removidas e obtemos o congestionamento para segmentos únicos para as categorias de interesse (3, 4 e 5). Em seguida, somamos os valores do comprimento de congestionamento e obtemos uma aproximação do valor da fila em metros, posteriormente convertido em quilômetros, para o intervalo de 30 minutos iniciando às 7 h até às 20 h, conforme Figura 9.



Cabe ressaltar que a categoria 3 pode inserir ruídos nos dados de congestionamento por ser uma fase intermediária entre o fluxo livre/leve e pesado/parado. Muitos segmentos são classificados como 3 pelo algoritmo do aplicativo, porém em poucos instantes mudam para categoria 2 ou 4. Este efeito gera a diferença da magnitude do congestionamento entre os grupos e uma limitação na reconstrução do status de tráfego no passado. No entanto, como estamos interessados na forma da curva e sua tendência, e menos nos valores absolutos, não temos um impacto significativo na análise e validação.

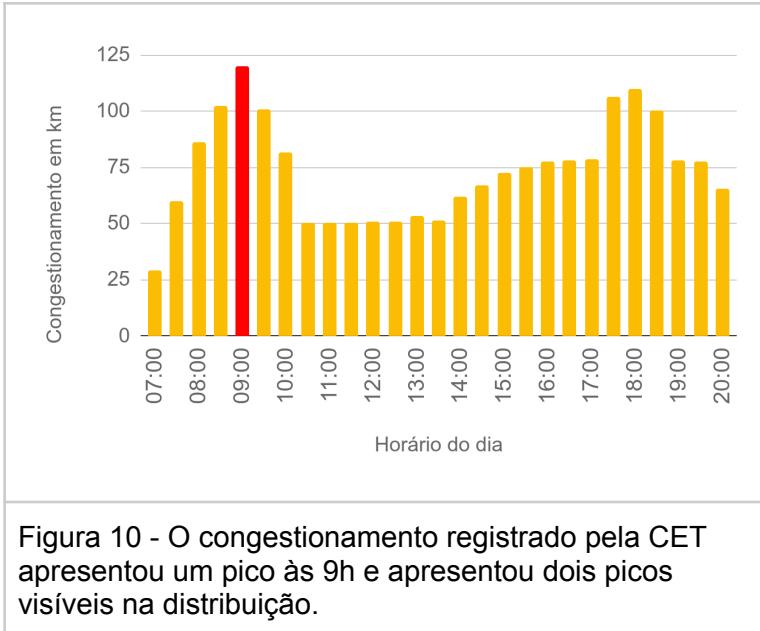


Figura 10 - O congestionamento registrado pela CET apresentou um pico às 9h e apresentou dois picos visíveis na distribuição.

O pico máximo registrado pelos dados coletados pela API do Waze ocorreu às 18h para ambos os grupos, Fig. 9. A CET registrou um pico máximo às 9h no valor de 120 km de fila e outro às 18h de 110 km, Fig. 10. Observamos que os Grupos 1 e 2 também detectaram uma flutuação no entre-picos às 13h que foi levemente detectado pela CET. A sensibilidade da detecção de tráfego se dá pela quantidade de segmentos monitorados. A CET seleciona trechos de via ao passo que nos grupos observamos toda sua extensão. O resultado é observável na suavização da curva dos grupos o qual agrupa a média de um conjunto de observações dos segmentos nos intervalos de 30 minutos, enquanto a CET extrai uma observação por segmento monitorado no mesmo intervalo.

Para comparar os picos, selecionamos as duas vias mais congestionadas registradas às 9h e às 18h. Na Tabela 2, o Grupo 1 capturou as mesmas vias em comparação com a CET às 9h, apenas a ordem está inversa. O Grupo 2 registrou a Marginal Pinheiros e a Av. dos Bandeirantes. A Av. dos Bandeirantes teve 7 km de congestionamento, de acordo com os dados CET. Na Tabela 3, observamos o congestionamento às 18h, e notamos que a correspondência foi exata entre os Grupos 1 e 2 com as vias detectadas pela CET.

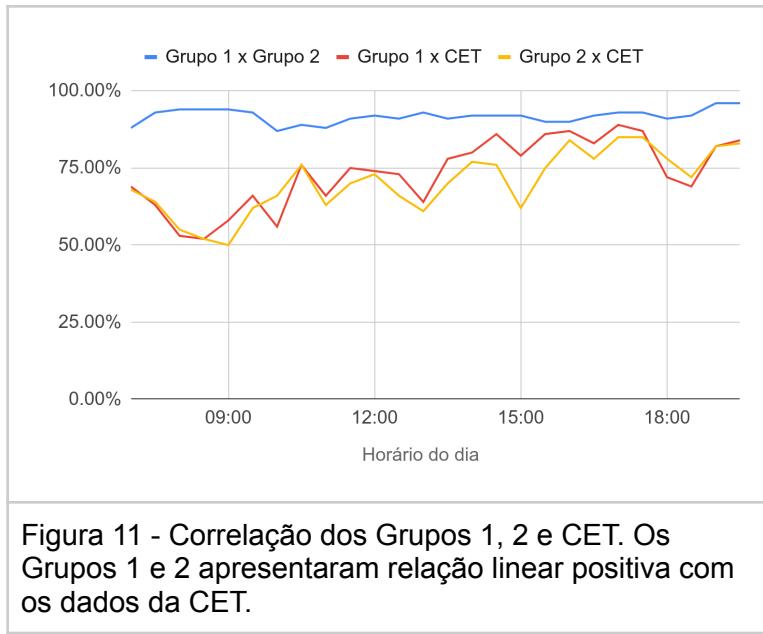
Tabela 2 - Duas primeiras vias mais congestionadas às 9h pelos Grupos 1, Grupo 2 e CET

Grupo 1	Grupo 2	CET
Via	Via	Via
Marg. Pinheiros	Marg. Pinheiros	Marg. Tietê
Marg. Tietê	Av. dos Bandeirantes	Marg. Pinheiros

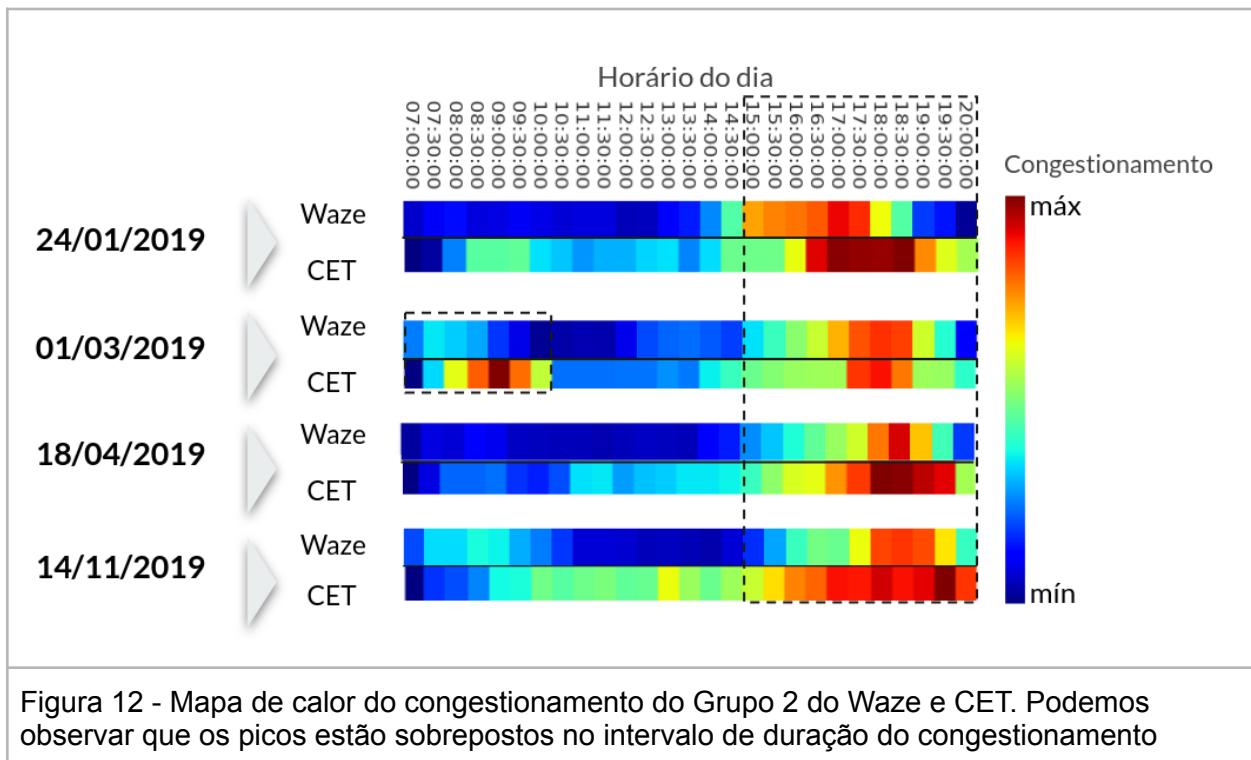
Tabela 3 - Duas primeiras vias mais congestionadas às 18h pelos Grupos 1, Grupo 2 e CET

Grupo 1	Grupo 2	CET
Via	Via	Via
Marg. Tietê	Marg. Tietê	Marg. Tietê
Marg. PInheiros	Marg. Pinheiros	Marg. Pinheiros

Como os valores absolutos de congestionamento não são diretamente comparáveis, calculamos a correlação para obter a relação linear entre as curvas. Selecioneamos 30 dias do ano de 2019 em diferentes meses e obtemos a média da correlação de congestionamento para cada intervalo de 30 minutos do dia das 7h até 20h, Fig. 11. Em seguida realizamos três comparações: Grupo 1 x CET, Grupo x CET, Grupo 1 x Grupo 2. Podemos observar que a relação linear entre cada grupo com a CET é positiva ficando acima de 50%. O Grupo 1, em geral, teve um desempenho levemente melhor quando comparado ao Grupo 2. Um outro ponto importante é que a relação linear apresentou uma tendência crescente em direção ao final da tarde para as curvas do Grupo 1 x CET e Grupo 2 x CET. Isto pode estar relacionado com a quantidade de usuários conectados ao aplicativo ou na diferença de classificação do congestionamento entre o método do Waze e da CET no horário da manhã. Neste caso, para identificar a causa seria recomendável realizar um estudo mais detalhado comparando a classificação registrada no aplicativo e o que é detectado em campo por inspeção visual. No entanto, mesmo com algumas flutuações observamos que é possível utilizar a classificação do Waze como uma aproximação do congestionamento real. A correlação entre os grupos ficou acima de 88% com uma certa estabilidade ao longo do dia, indicando que o Grupo 1 e o Grupo 2 possuem forte relação linear positiva.



Na Figura 12, selecionamos quatro vésperas de feriados em dias úteis somente no Grupo 2 que representa tráfego pesado/parado. Nós observamos que o intervalo de duração do congestionamento nos dias selecionados estão sobrepostos aos dados coletados da CET. Apenas o dia 01/03/2019, conforme mencionado anteriormente, registrou pico máximo no período da manhã. O mapa de calor reforça a utilização dos dados do Waze como uma fonte confiável de detecção de congestionamento da cidade.



indicando que o Waze consegue identificar filas nas vias.

Apesar dos resultados positivos no uso dos dados do Waze observados em São Paulo, reforçamos que a qualidade dos dados depende da base de usuários que o aplicativo possui em cada cidade. Obviamente, uma base menor pode ter uma janela de observação maior para que mais dados sejam incluídos em cada subconjunto de observações dos segmentos. Além disso, pode-se agregar os segmentos e obter uma classificação da via em vez de um trecho.

Um esforço adicional por parte dos gestores de tráfego das cidades é a correta associação dos nomes dos logradouros de cada trecho de via no Waze versus o que está registrado nos mapas da cidade. Em geral, os nomes possuem diferenças e a equivalência é fundamental para uma análise correta do tráfego.

Quanto à escolha do Grupo 1 ou 2, para a cidade de São Paulo ambos tiveram desempenho similar. Em diálogo com analistas da CET, um estudo interno na companhia revelou que o Grupo 2 seria mais indicado para uso em produção, já que o número de falsos positivos era relativamente alto quando incluíam a categoria 3 e cruzarem com os dados do CFTV no mesmo instante.

Protótipo de plataforma de dados

Uma vez validado o potencial de uso dos dados do Waze como gestão de tráfego urbano, o próximo passo foi elaborar uma prova de conceito que servisse de referência inicial para um produto mais robusto no futuro, Fig. 13.

O painel possui quatro divisões:

- **Controle** - filtros para definir o intervalo a ser analisado, a cidade, o grupo 1 (categorias 3-5 do Waze), grupo 2 (categorias 4-5 do Waze), tipo do dia (útil, final de semana, ou ambos), a via monitorada, tipo de dado (histórico ou tempo real).
- **Indicador de congestionamento** - representado pelo valor do congestionamento em quilômetros e um delta referente a diferença de uma linha de base que pode ser uma média ou uma curva histórica versus a curva real.
- **Mapa dos trechos congestionados** - aponta os segmentos congestionados no mapa para os dados históricos e em tempo real.
- **Histograma do volume de congestionamento** - para os dados históricos a agregação temporal é a cada 30 minutos, enquanto o tempo real é a cada 5 minutos. A curva de referência pode ser calculada seguindo um critério de análise da série histórica, mas para ilustração adotamos a média nos dados históricos e a curva de congestionamento da observação de um único dia.

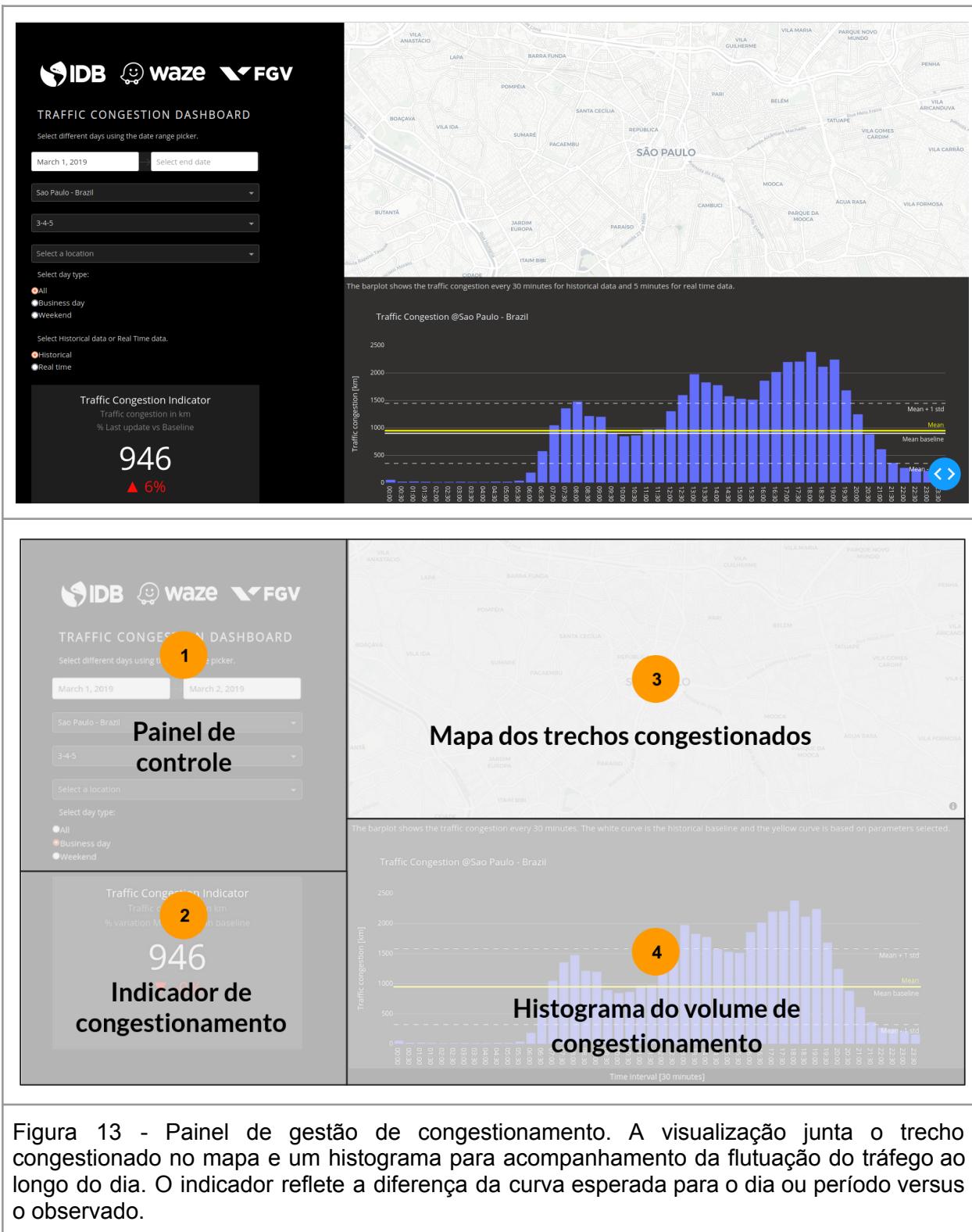


Figura 13 - Painel de gestão de congestionamento. A visualização junta o trecho congestionado no mapa e um histograma para acompanhamento da flutuação do tráfego ao longo do dia. O indicador reflete a diferença da curva esperada para o dia ou período versus o observado.

Os dados seguem dois fluxos distintos: históricos e tempo real. Para os dados históricos a arquitetura inicia na captura dos dados na API do Waze via solicitação HTTP, posteriormente é

processado e inserido no bucket 'aws-athena-query-results-east-2' e base de dados 'cities', ambos na nuvem da FGV configurada na *Amazon Web Service* (AWS). Como utilizamos ferramentas *Open Source* no desenvolvimento do painel utilizamos o Python como linguagem de programação e as ferramentas Dash da Plotly, Pandas, Numpy e Geopandas, conforme ilustrado na Fig. 14.

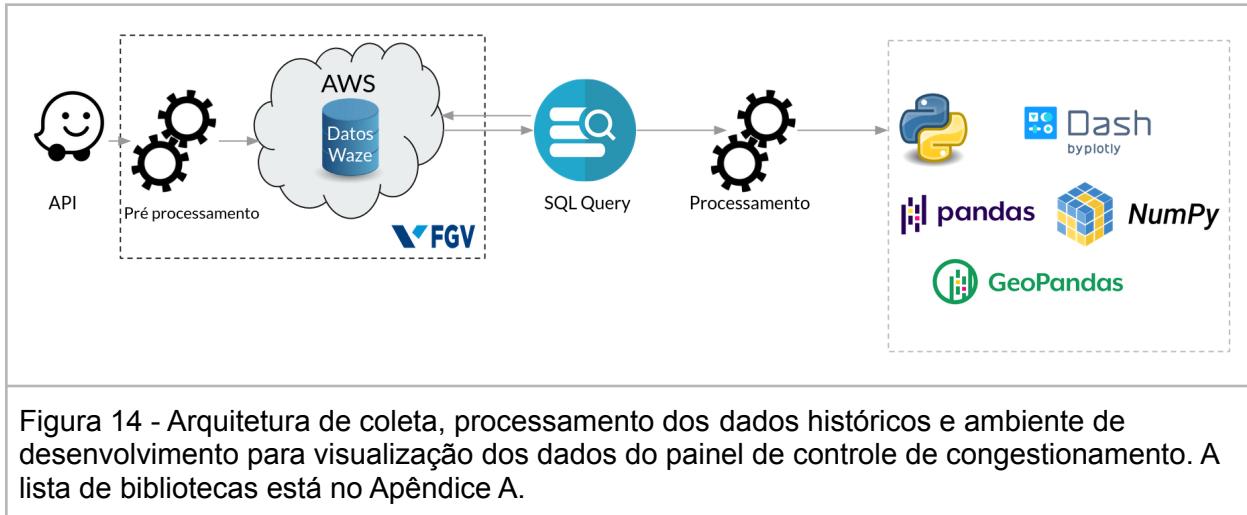


Figura 14 - Arquitetura de coleta, processamento dos dados históricos e ambiente de desenvolvimento para visualização dos dados do painel de controle de congestionamento. A lista de bibliotecas está no Apêndice A.

Para acesso dos serviços da AWS fora do ambiente da nuvem é necessário instalar duas bibliotecas: Boto3⁶ e AWScli. Boto3 acessa os serviços de nuvem da Amazon direto do IDE, bastando primeiramente configurar as credenciais de acesso, para isto usamos o AWScli para configurar os arquivos `~.aws/credentials` e `~.aws/config`. No primeiro arquivo é configurado a chave de usuário e chave de acesso, no último configurações de acesso à nuvem como região, grupo de trabalho, banco de dados. Após estas configurações é possível realizar *queries* dentro dos blocos dos *scripts* sem a necessidade de acessar a suíte da AWS na área logada via *web browser*.

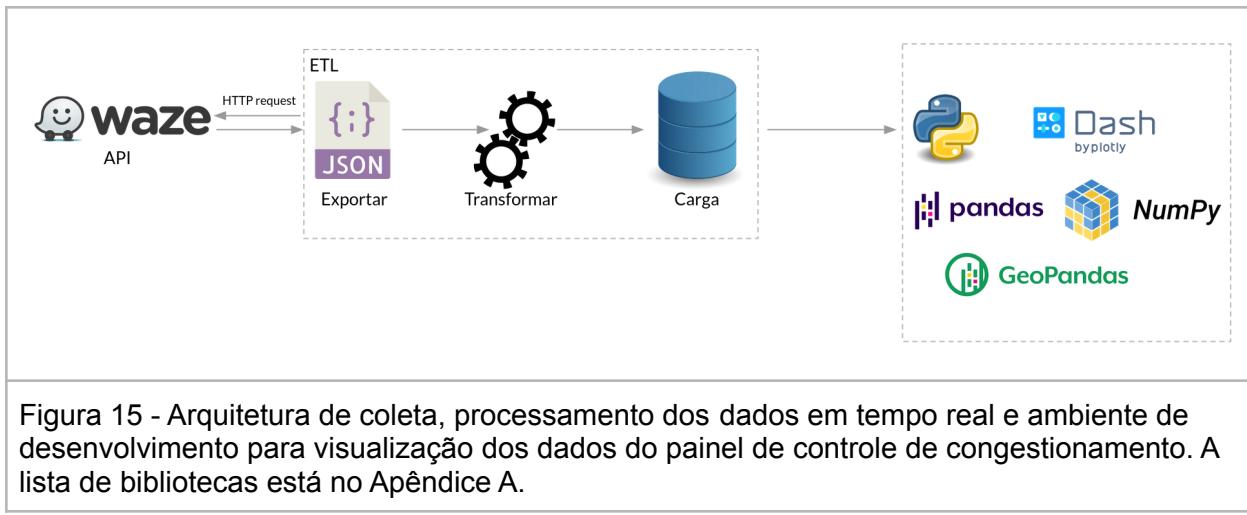


Figura 15 - Arquitetura de coleta, processamento dos dados em tempo real e ambiente de desenvolvimento para visualização dos dados do painel de controle de congestionamento. A lista de bibliotecas está no Apêndice A.

⁶ <https://boto3.amazonaws.com/v1/documentation/api/latest/index.html>

Para os dados em tempo real realizamos chamadas diretas na API do Waze sem passar pela nuvem da FGV, desta forma diminuímos a latência e os dados são processados e visualizados de acordo com a configuração do intervalo de requisições, atualmente no intervalo de 5 minutos. Em seguida transformamos os dados para serem consumidos pelos scripts de visualização. Para realizar chamadas direto na API é necessário ser cadastrado como Partner do Waze e obter a credencial Clientid. Além disso, é necessário configurar o polígono da cidade que no caso de Xalapa, Quito, Montevideo, e o distrito de Miraflores a chamada comporta o retorno referente apenas um polígono, porém para São Paulo é necessário separar em dois ou mais polígonos e posteriormente concatenar os resultados.

O arquivo de DE-PARA das vias monitoradas é essencial no painel de controle, já que converte os trechos com as denominações do Waze para os nomes dos logradouros que os gestores de tráfego das cidades estão mais acostumados, ver Tabela 4. Para exemplificar, a 'Av. Galo Plaza Lasso' pode ser identificada no Waze como: 'Av. Galo Plaza Lasso (Vía Lateral) > (N)', 'Av. Galo Plaza Lasso (Vía Central)', 'Av. Galo Plaza Lasso', 'Salida Hacia Av. Galo Plaza Lasso > (N)', 'Salida Hacia Av. Galo Plaza Lasso (Vía Lateral) > (N)', e 'Salida Av. Galo Plaza Lasso > (S)'.

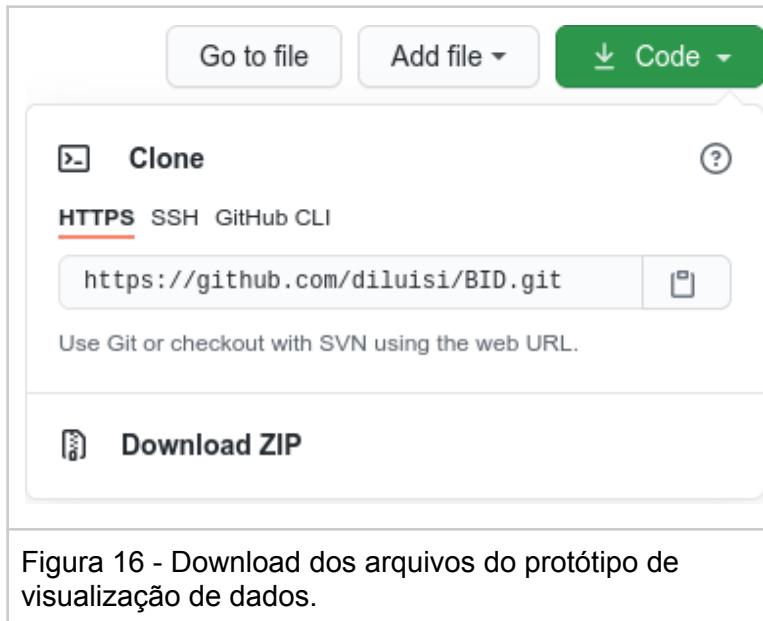
Tabela 4 - Descrição dos campos do arquivo de DE-PARA de vias

Nome	Tipo	Descrição
new_street	string	Nomenclatura utilizada pelas equipes de gestão de tráfego das cidades
street	string	Nomenclatura utilizada pelo Waze

Para a cidade de São Paulo, por exemplo, que já possui os logradouros das vias monitoradas de acordo com o mapa da prefeitura, o arquivo servirá de referência para conversão dos trechos do identificados no Waze para o arquivo original da CET. As demais cidades que não possuem um arquivo de vias monitoradas deverão primeiramente selecioná-las e em seguida identificá-las no mapa do Waze. Vale ressaltar que os trechos com vias nos dois sentidos poderão ter nomenclaturas distintas. A escolha das vias pode obedecer um critério de volume de tráfego, classificação viária, por região de interesse, ou mesmo uma seleção sem um critério pré-determinado.

Instalação

A instalação do protótipo de visualização de dados de congestionamento é realizada após o *download* do projeto no repositório <https://github.com/diluisi/BID>, conforme Fig. 16, e a instalação das bibliotecas python para este projeto, conforme Apêndice A. O arquivo de captura dos dados da API do Waze processa separadamente os dados e salva em arquivo *.csv. Este processo deverá ser substituído por carga no banco de dados na infra-estrutura de tecnologia de cada cidade. Esta etapa não está contemplada em nosso escopo.



Mapa da arquitetura de dados

Na Fig. 16, esquematizamos a visão geral da arquitetura dados atual. Para o protótipo não focamos em otimizações de código ou de arquitetura, portanto cabe ressaltar que o resultado final pode haver algumas restrições de performance. Para a instalação em ambiente de produção será necessário ajustar o modelo para diminuir a latência nas buscas de dados da AWS, que atualmente apresenta o maior gargalo no tempo de execução. Além disso, o pré-processamento dos dados diretamente dos dados da API do Waze reduzem o espaço de armazenamento, bem como facilita na visualização de dados em tempo real.

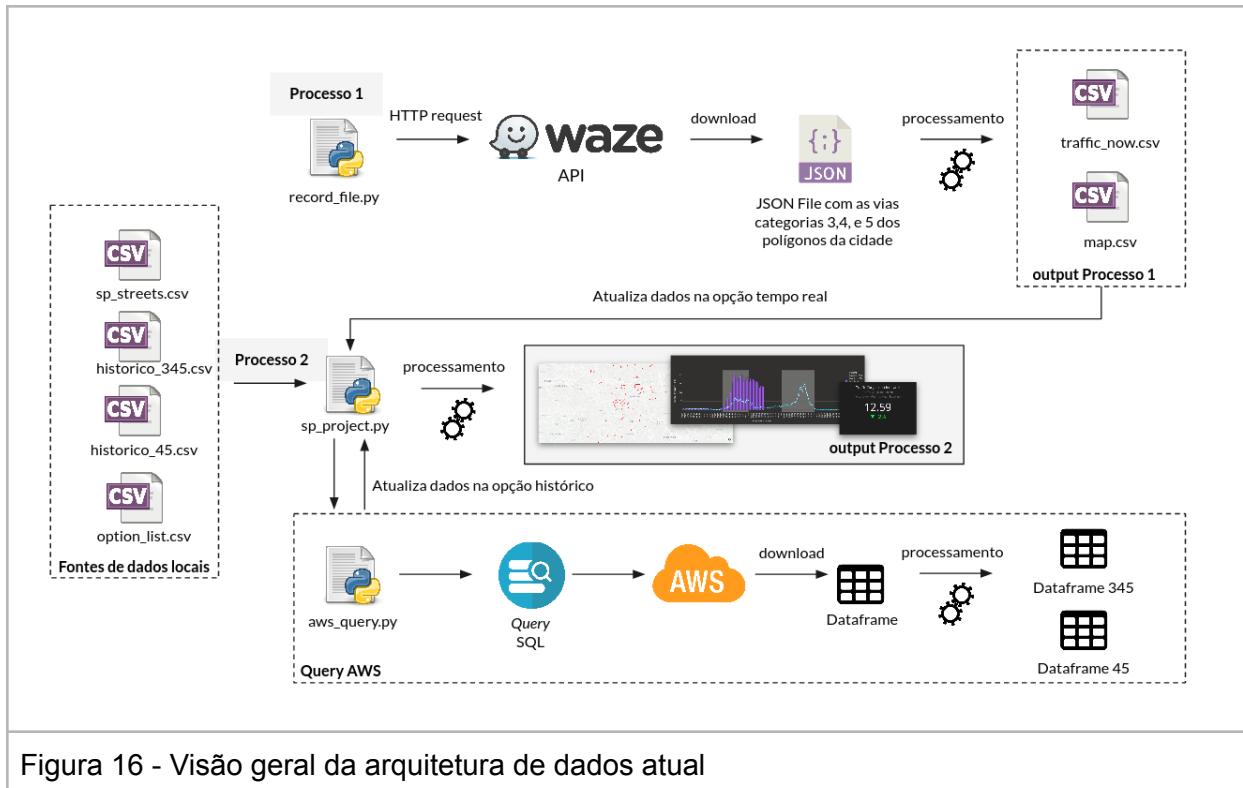


Figura 16 - Visão geral da arquitetura de dados atual

Na Tabela 5, descrevemos os arquivos e processos da arquitetura atual:

Tabela 5 - Descrição da arquitetura dados atual

Nome	Descrição
Processo 1	Processo de gravação dos dados em tempo real da API do Waze
Processo 2	Processo core de geração da visualização de dados
<code>sp_streets.csv</code>	Arquivo de DE-PARA das vias monitoradas
<code>historico_345.csv</code>	Curva de referência para o Grupo 1. Utilizamos de um único dia, mas o ideal é que a curva seja o resultado de um modelo de análise histórica com as particularidades de tráfego de cada cidade.
<code>historico_45.csv</code>	Curva de referência para o Grupo 2. Utilizamos de um único dia, mas o ideal é que a curva seja o resultado de um modelo de análise histórica com as particularidades de tráfego de cada cidade.
<code>option_list</code>	Opções de seleção de vias (útil quando se analisa mais de uma

	cidade)
record_file.py	Script de processamento de dados históricos
Waze API	Interface de equesição HTTP dos dados do Waze
JSON	Formato de saída da requisição dos dados da API do Waze
traffic_now.csv	Armazenamento dos dados extraídos da API. Visualização dos dados históricos agrupados a cada 30 minutos
map.csv	Armazenamento temporário dos dados de latitude e longitude dos trechos congestionados do Grupo 1 e Grupo 2. Este arquivo é sobreescrito a cada atualização.
aws_query.py	Envio de solicitação da query para a AWS de acordo com o intervalo selecionado no painel. Retorna dois <i>dataframes</i> para cada grupo.
sp_project.py	Script de processamento e geração da visualização de dados

Configurando de uma nova cidade no painel

Para cada arquivo destacamos os trechos de códigos ou campos que devem ser modificados para executar o projeto em uma nova cidade.

No arquivo aws_query.py é necessário modificar a *query* executada na AWS, conforme trecho de código abaixo:

aws_query.py
<pre>sql = ('SELECT "pub_utc_date", "polygon_slug", "uuid", "street", "length", ' ' "level", "delay", "type", "line_geojson" ' ' FROM "cities"."br_saopaulo_waze_jams" ' ' WHERE year=2019 AND month = '+str(1)+' AND day='+str(1)+ ' AND city=\São Paulo\\'')</pre>

O arquivo sp_project.py contém o código de visualização e neste *script* é necessário modificar o *drop-down* com as opções de cidade.

sp_project.py

```
dcc.Dropdown(  
    id="region-dropdown",  
    options=[  
        {"label": "Sao Paulo -  
Brazil", "value": "Sao Paulo - Brazil"},  
    ],  
    placeholder="Select a category",  
    value='Sao Paulo - Brazil',  
    style={"width": "45rem"},  
    clearable=False,  
)
```

O mapa possui coordenadas para centralizá-lo, a alteração seguinte é somente das coordenadas da cidade que permite visualizar os seus limites geográficos. O valor da chave mpbx_center no dicionário coords é passado para a função de atualização do mapa px.scatter_mapbox, a alteração é na posição da cidade na lista da chave region do dicionário coords. No exemplo abaixo cada cidade está em uma posição, São Paulo é “0”, Quito “1”, Miraflores “2”, e Montevideo “3”.

sp_project.py

```
coords = pd.DataFrame({"region": ["Sao Paulo - Brazil", "Quito - Ecuador",  
"Miraflores - Peru", "Montevideo - Uruguay"],  
  
"city_la": [-23.506226, -0.111780, -12.029391, -34.782417],  
  
"city_lo": [-46.6336332, -78.450622, -77.028059, -56.222555],  
  
"mpbx_center": [-23.550164466, -0.202431, -12.124605, -34.905283]})  
. . .  
  
figb = px.scatter_mapbox(lat=[coords[coords['region']==region].city_la.iloc[0]],  
lon=[coords[coords['region']==region].city_lo.iloc[0]], zoom=3, height=500)  
figb.update_layout(mapbox_style="carto-positron",  
mapbox_zoom=12, mapbox_center_lat = coords[coords['region']==region].mpbx_center.iloc[0],  
margin={"r":0,"t":0,"l":0,"b":0})
```

No arquivo `record_file.py` o `script` é executado continuamente para coleta dos dados a cada cinco minutos da API do Waze. Neste caso é preciso modificar a lista do id dos polígonos, o arquivo de DE-PARA e os pontos de latitude e longitude da poligonal.

record_file.py

```
cities = ['Sao Paulo 1', 'Sao Paulo 2', 'Sao Paulo 3']

de_para_sp = pd.read_csv('/sp_streets.txt')

WAZE_POLYGONS = {
    'Sao Paulo 1': {
        'city': 'São Paulo',
        'points':
        '-46.89388,-23.65469;-46.8972,-23.79037;-46.32591,-23.791;-46.32865,-23.6443
1;-46.89388,-23.65469',
    },
    'Sao Paulo 2': {
        'city': 'São Paulo',
        'points':
        '-46.9344,-23.4539;-46.91986,-23.55706;-46.27304,-23.52685;-46.30119,-23.359
91;-46.9344,-23.4539',
    },
    'Sao Paulo 3': {
        'city': 'São Paulo',
        'points':
        '-46.2656,-23.52821;-46.92789,-23.5558;-46.90063,-23.65647;-46.31814,-23.643
91;-46.2656,-23.52821',
    }
}
```

Abaixo estão os polígonos que estão definidos na base de dados AWS.

```
# polígonos de cada cidade
WAZE_POLYGONS = {
    'Xalapa': {
        'city': 'Xalapa',
        'points':
        '-97.16627,19.797331;-96.57464,19.794427;-96.57660,19.272967;-97.1750
0,19.276073;-97.16627,19.797331',
    },
    'Quito': {
        'city': 'Quito',
        'points':
        '-78.486,0.133;-78.595,-0.176;-78.648,-0.36;-78.445,-0.404;-78.25,0.1
14;-78.486,0.133',
    },
}
```

```

'Montevideo': {
    'city': 'Montevideo',
    'points':
    '-56.324,-34.601;-56.45,-34.881;-55.934,-35.003;-55.39,-34.854;-56.32
4,-34.601',
    },
    'Sao Paulo 1': {
        'city': 'São Paulo',
        'points':
        '-46.89388,-23.65469;-46.8972,-23.79037;-46.32591,-23.791;-46.32865,-
23.64431;-46.89388,-23.65469',
    },
    'Sao Paulo 2': {
        'city': 'São Paulo',
        'points':
        '-46.9344,-23.4539;-46.91986,-23.55706;-46.27304,-23.52685;-46.30119,
-23.35991;-46.9344,-23.4539',
    },
    'Sao Paulo 3': {
        'city': 'São Paulo',
        'points':
        '-46.2656,-23.52821;-46.92789,-23.5558;-46.90063,-23.65647;-46.31814,
-23.64391;-46.2656,-23.52821',
    },
    'Lima':{
        'city':'Miraflores',
        'points':'-77.086,-11.894;-77.187,-12.048;-77.053,-12.211;-76.94,-12.
258;-76.824,-12.197;-76.846,-11.925;-77.086,-11.894'
    }
}

```

Polígonos de cada cidade

O arquivo `record_file.py` foi desenvolvido para atender a cidade de São Paulo que possui três poligonais que precisam ser concatenadas. Para cidades que possuem apenas uma poligonal o código pode ser simplificado.

Além das modificações mencionadas anteriormente, ainda é necessário ajustar os arquivos de dados locais para que correspondam aos da cidade a ser incluída.

Conclusão

Neste estudo avaliamos o uso dos dados extraídos do Waze como referência na gestão de congestionamento. Avaliamos e comparamos os dados na cidade de São Paulo, por já possuir um método de inspeção do trânsito e uma base de dados histórica.

O resultado da comparação dos dados do Waze com os da CET foi satisfatório e mostrou que a API conseguiu identificar as flutuações de congestionamento. Não identificamos uma diferença significativa entre os Grupos 1 e 2, ou seja, ambos tiveram correlações similares quando comparados com os dados da CET.

O passo decorrente da avaliação foi a criação de um protótipo para gestão de tráfego utilizando as premissas:

- Usabilidade e foco na experiência do usuário;
- Código aberto;
- Interface que proporcione o diagnóstico do tráfego utilizando os dados históricos e tempo real;
- Portável;
- Extensível;
- Adaptável;
- Simples execução e manutenção;
- Escalabilidade da monitoração;
- Baixo custo.

Procuramos simplificar o processo de gestão a partir da extração direta da API do Waze e a visualização dos trechos monitorados no mapa e no histograma. Além disso, computamos um indicador de congestionamento que é o somatório dos trechos congestionados a cada intervalo e o delta que indica a variação em relação a uma curva de referência.

O painel por ser desenvolvido em código aberto poderá ser incrementado, conforme necessidade de cada cidade, podendo agregar modelos mais robustos e gráficos adicionais.

Este projeto foi desenvolvido em colaboração com a Fundação Getúlio Vargas, Banco Interamericano de Desenvolvimento e a companhia Waze.

Referências

BID. (2021a)

<https://publications.iadb.org/publications/spanish/document/Congestion-urbana-en-America-Latina-y-el-Caribe-Caracter%C3%ADsticas-costos-mitigacion.pdf> - Acessado em Maio/21

BID. (2021b)

<https://blogs.iadb.org/transporte/es/big-data-un-aliado-de-las-ciudades-en-la-lucha-contra-la-congestion/>

Cebr.(2014)

[https://www.ibtta.org/sites/default/files/documents/MAF/Costs-of-Congestion-INRIX-Cebr-Report%20\(3\).pdf](https://www.ibtta.org/sites/default/files/documents/MAF/Costs-of-Congestion-INRIX-Cebr-Report%20(3).pdf) - Acessado em Maio/21

European Comission. (2020)

<https://op.europa.eu/en/publication-detail/-/publication/9781f65f-8448-11ea-bf12-01aa75ed71a1> - Acessado em Maio/21

ONU. (2018)

<https://www.un.org/development/desa/es/news/population/2018-world-urbanization-prospects.html> - Acessado em Maio/2021

Waze Traffic-data Specification Document version 2.8

Apêndice A

Lista das ferramentas utilizadas para o desenvolvimento do painel de congestionamento:

Nome	Versão	Descrição
Python	3.7.4	Linguagem de Programação
Pandas	0.25.1	Biblioteca Python
Numpy	1.17.2	Biblioteca Python
json	2.0.9	Biblioteca Python
requests	2.22.0	Biblioteca Python
Dash	1.19.0	Biblioteca Python
Plotly	4.14.3	Biblioteca Python
GeoPandas	0.6.2	Biblioteca Python
shapely	1.6.4.post2	Biblioteca Python
retry	retry-0.9.2	Biblioteca Python

As bibliotecas instalam automaticamente as dependências, por isso não listamos as dependências de cada biblioteca.

Para instalação de bibliotecas Python recomendamos os instaladores pip⁷ ou conda⁸.

⁷ <https://pypi.org/>

⁸ <https://www.anaconda.com/products/individual>