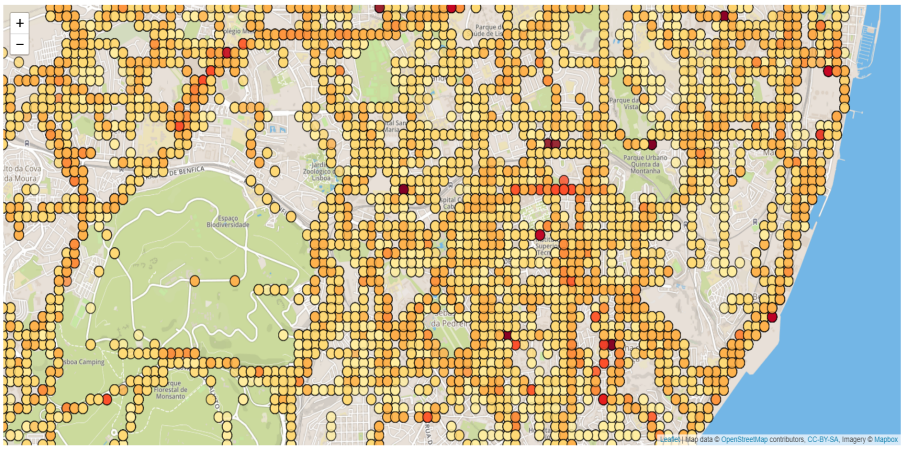
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores



Sistema de visualização analítica de PM2.5: caso de Lisboa

Rúben Miguel Gomes Taborda

(Licenciado)

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Doutor Nuno Datia
Doutora Matilde Pato

Júri:

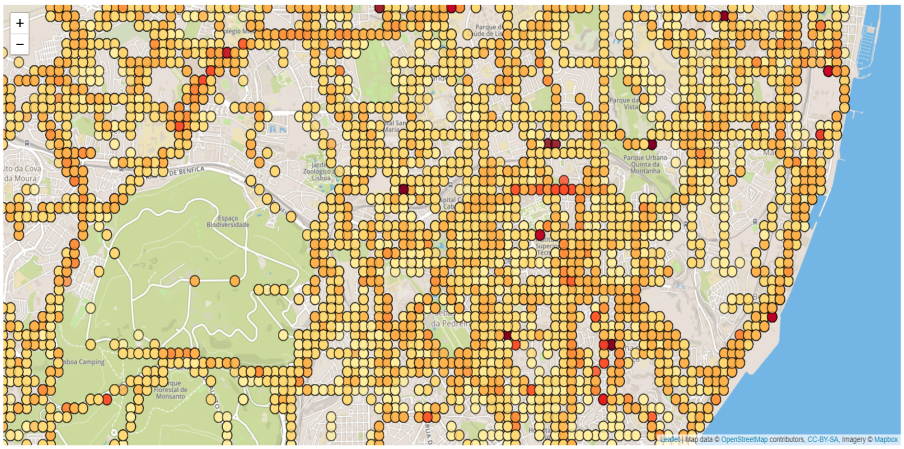
Presidente: Doutor José Manuel De Campos Lages Garcia Simão

Vogais: Doutor Ricardo Filipe Da Cruz Dos Santos de Almeida e Silva
Doutora Matilde Pós-De-Mina-Pato

Setembro, 2020

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Área Departamental de Engenharia de Electrónica e Telecomunicações e de Computadores



Sistema de visualização analítica de PM2.5: caso de Lisboa

Rúben Miguel Gomes Taborda

(Licenciado)

Dissertação para obtenção do Grau de Mestre
em Engenharia Informática e de Computadores

Orientadores : Doutor Nuno Datia
Doutora Matilde Pato

Júri:

Presidente: Doutor José Manuel De Campos Lages Garcia Simão

Vogais: Doutor Ricardo Filipe Da Cruz Dos Santos de Almeida e Silva
Doutora Matilde Pós-De-Mina-Pato

Setembro, 2020

Aos meus pais.

Agradecimentos

Aos meus orientadores, por todo o apoio e disponibilidade que me deram ao longo da realização da dissertação. A todos os meus amigos, que me ajudaram a manter me são e animado durante esta etapa. Aos meus familiares, especialmente aos meus pais, pois sem eles nada disto seria possível.

Resumo

A qualidade do ar é monitorizada através de dados colectados em estações fixas seleccionadas em uma região, geralmente a cidade. Tal abordagem não permite uma compreensão apurada sobre a qualidade do ar, nomeadamente, em áreas distantes das estações que colectam estes dados, especialmente em áreas urbanas residenciais. Neste relatório, descrevemos uma plataforma onde os responsáveis de decisão do conselho municipal podem visualizar dados da poluição do ar, usando uma visualização através de um *dashboard* interactivo baseado num mapa com resolução espacial múltipla. Os dados de qualidade do ar são recolhidos, detectando as condições ambientais da cidade de Lisboa. Os dados de poluição do ar são então integrados a outros dados ambientais e exibidos no *dashboard*. Esses dados incluem, entre outros, dados de mobilidade espaço-temporal, fornecendo informações contextuais sobre a poluição do ar. A solução é feita sob medida para os responsáveis de decisão do conselho municipal, permitindo um melhor entendimento das questões de qualidade do ar e actuando como uma ferramenta de apoio para diferentes comunidades, explorando sinergias para promover a sustentabilidade da cidade.

O primeiro passo para a concretização da aplicação foi o desenvolvimento das componentes e serviços que permitem obter, processar e guardar os dados relativos ao poluente, seguidamente a realização de um *proxy*, controlador que permite-se fazer a gestão dos dados a enviar para a aplicação.

Por fim seguiu-se o processamento de mais fontes de dados, neste caso de trânsito, que nos permitisse juntar aos demais, criando assim um modelo capaz de prever o valor do poluente para um certo dia.

Palavras-chave: Poluição do ar, visualização interactiva, cidades inteligentes, informações contextuais.

Abstract

Air quality is monitored using data recollected using fixed selected stations in a region, generally the city. Such approach does not support a fine-grained comprehension about the air quality, namely, in areas distant from the collector' stations, specially in residential urban sites. In this report, we describe a platform where city council decision-makers can visualize air pollution data, using an interactive map-based dashboard visualization with multiple spatial resolution. The air quality data is collected, detecting environmental conditions in the Lisbon's city. Air pollution data is then integrated with other environmental data and displayed into the dashboard. Such data includes, among other, spatio-temporal mobility data, providing contextual information about air pollution. The solution is tailored to city council decision-makers, enabling a better understanding of air quality issues, and acting as a support tool for different communities, exploiting synergies to promote the sustainability of the city.

The first step in implementing the application was the development of components and services that allow obtaining, processing and saving data related to the pollutant, followed by a *proxy*, a controller that allows data management to be carried out send to the application.

Finally, more data sources were processed, in this case of transit, which would allow us to join the others, thus creating a model capable of predicting the pollutant's value for a certain day.

Keywords: Air pollution, Interactive visualization, Smart cities, Contextual information

Índice

Lista de Figuras

Lista de Tabelas

Lista de Listagens

Lista de Abreviaturas e Siglas

μ**g/m**^{3}	Micrograma por metro cúbico. 6, 7, 8, 9
AEI	Indicador de exposição média de concentrações para PM2.5 (do inglês *Average Exposure Indicator for PM2.5 concentrations*). 5, 6, 8, 9
AUC	*Area Under the ROC Curve*. 40
AUCPR	*Area Under the Precision-Recall Curve*. 40
AutoML	*Automatic Machine Learning*. 40, 43, 44, 47
CML	Câmara Municipal de Lisboa. 1, 3
EAQI	*EU Air Quality Index*. 57
EEA	Agência Europeia do Ambiente (do inglês *European Environment Agency*). 5
EEE	Espaço Económico Europeu. xv, 6, 7, 8
EMEL	Empresa Municipal de Mobilidade e Estacionamento de Lisboa. 3
GBDT	*Gradient Boosting Decision Tree*. 40
GBM	*Gradient Boost Machine*. 40, 43
GLM	*Generalized Linear Model*. 45
IDW	Inverso da potência das distâncias (do inglês *Inverse Distance Weighting*). 12
IPMA	Instituto Português do Mar e da Atmosfera. 3, 64

MAE *Mean Absolute Error*. 40, 44
MCC *Matthews Correlation Coefficient*. 40
MOJO *Model ObJect, Optimized*. 47
MSE *Mean Squared Error*. 40, 44

OcK *Ordinary coKriging*. 12
OK *Ordinary Kriging method*, conhecido por "Processo Gaussiano de Regressão". 12
OMS Organização Mundial de Saúde. 2, 5, 7, 64

PM partículas finas (do inglês *Particulate Matter*). xiii, xiv, xv, xvi, xix, xxi, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 17, 19, 21, 22, 23, 24, 25, 27, 28, 29, 30, 31, 32, 33, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 53, 54, 55, 56, 59, 61, 62, 63, 65
POJO *Plain Old Java Object*. 47

RF *Random Forest*. 40, 43
RLM Regressão Linear Múltipla. 12
RMSE *Root Mean Squared Error*. 40, 44
RMSLE *Root Mean Squared Logarithmic Error*. 40, 44

UE União Europeia. xv, 5, 6, 8, 9, 10, 11

YLL *Years of Life Lost*. xv, 8, 11

1

Introdução

O novo relatório oficial do Painel Intergovernamental de Mudanças Climáticas define para o mundo uma meta clara: "devemos reduzir as emissões de gases de efeito estufa para a rede zero até meados deste século". Acelerar a transição para um crescimento limpo e sustentável aumentando os esforços no combate à pobreza, elevar os padrões de vida e melhorar a prosperidade é uma escolha nossa. A 28 de junho de 2016, a Câmara Municipal de Lisboa (CML) aderiu ao Pacto de Autarcas para o Clima e Energia (Covenant of Mayors for Climate and Energy) e na página da Assembleia Municipal lemos "Lisboa é a primeira capital europeia a aderir ao Pacto de Autarcas para o Clima e Energia"[1]. A capital portuguesa passou a ter a responsabilidade em reduzir as suas emissões de dióxido de carbono (CO_2) em pelo menos 40% até 2030.

Quando falamos de poluição atmosférica, pensamos normalmente em níveis de CO_2, o que é redutor. O CO_2 é apenas um dos componentes que está associado à poluição atmosférica. A designação mais correta para os poluentes na atmosfera das grandes cidades industrializadas e em vias de desenvolvimento designa-se por "*photochemical smog*" (*Smog*). *Smog* é um tipo de poluição atmosférica produzido quando a luz ultravioleta do sol reage com óxidos de azoto (NOx) na atmosfera. Causado principalmente pelos veículos motorizados, os óxidos de azoto são introduzidos na atmosfera que combinados com a água (H_2O) formam ácido nítrico (HNO_3), ou com a luz solar que juntamente com o oxigénio molecular produz ozono (O_3). Quando exposto à radiação ultravioleta o dióxido de azoto (NO_2) passa por uma série complexa de reacções

[1]Publicado na página da Assembleia Municipal de Lisboa a 15/07/2016, https://www.am-lisboa.pt/101000/1/005455,072016/index.htm

com hidrocarbonetos para produzir os componentes da poluição fotoquímica, uma mistura de ozono, ácido nítrico, aldeídos, peroxiacetilnitrato (PANs) e outros poluentes secundários [11]. Os aviões, barcos, a actividade industrial e a geração de energia também são uma fonte de poluição considerável [17].

A poluição atmosférica tem um impacto enorme na saúde da população [19]. Dados da Organização Mundial de Saúde (OMS) afirmam que a poluição do ar é um fator de risco crítico para doenças não transmissíveis, causando cerca de 24% de mortes por doenças cardiovasculares, 25% por acidente vascular cerebral, 43% por doença pulmonar obstrutiva crónica (DPOC) e 29% associadas ao cancro do pulmão.

A exposição a partículas finas (do inglês *Particulate Matter*) (PM) é particularmente perigosa para a saúde. Estas partículas, de diâmetro inferior a 2.5 μm (PM2.5) e inferior a 10 μm (PM10), são consideradas a sexta maior causa de morte prematura no Sudeste Asiático [5]. Em particular, as PM2.5 penetram facilmente nos pulmões, irritando e corroendo os alvéolos, provocando dificuldades respiratórias e várias doenças pulmonares [32].

"*Uma ferramenta de visualização simples de dados de PM2.5 pode ajudar os responsáveis técnicos e políticos da Câmara Municipal de Lisboa a compreender os eventos da poluição em algumas áreas da cidade? "*. As ferramentas de visualização têm componentes fundamentais — as representações visuais — que são construídas com base na combinação de diferentes codificações visuais como comprimento, posição, tamanho, saturação de cor e outros. Um componente central das representações visuais são os mapeamentos de valores de dados para representações gráficas. Os dados e a codificação visual são projectados para desenvolver ferramentas de visualização interactivas [1, 31]. Essas ferramentas de visualização permitem que os usuários especifiquem mapeamentos directos entre os seus dados e a representação visual, sem exigir que os utilizadores tenham habilidades de programação.

Até agora, o conselho de Lisboa não tem dados detalhados e refinados sobre a poluição do ar. Os valores oficialmente reportados provêm da Agência Ambiental Portuguesa (APA), que recolhe dados em cinco estações fixas [2]. Uma vez que Lisboa tem um aeroporto dentro dos limites da cidade, é visitada por vários cruzeiros, e um largo número de autoestradas que cruzam a cidade, é difícil obter uma imagem real da qualidade do ar de Lisboa utilizando os dados oficiais. Sabe-se que essas infraestruturas são importantes na contribuição da poluição atmosférica da cidade [20, 22, 23].

A Câmara Municipal de Lisboa está a utilizar uma plataforma de dados urbanos (PGIL) para gerir a cidade, com muitos dados em tempo real integrados, mas também com

[2]Mais detalhes em https://airindex.eea.europa.eu/

painéis específicos para apoiar a tomada de decisão [24]. Os responsáveis da cidade têm a responsabilidade de propor alternativas sustentáveis de maneira precisa e realista aos seus cidadãos. A capacidade de seleccionar e apresentar informações para apoiar a tomada de decisão é parte integrante de tal processo. As tecnologias de informação permitirão que os responsáveis apoiem melhor a colecta e a análise de dados para projectar, simular e apresentar cenários futuros de maneira eficaz.

Neste trabalho vamos concentrar-nos nas partículas PM2.5 e relacionar o valor da sua concentração tendo em conta o tráfego automóvel e as condições climatéricas. Iremos para isso recorrer à informação de tráfego disponibilizados pela Empresa Municipal de Mobilidade e Estacionamento de Lisboa (EMEL), condições atmosféricas disponibilizados pelo Instituto Português do Mar e da Atmosfera (IPMA) [13], e concentrações de PM2.5 disponibilizados por sensores de baixo custo (baseado em tecnologia laser, como o Grove HM-3301 laser dust sensor[3]).

Assim, pretende-se responder à questão: "Existe uma relação causa efeito entre tráfego e condições atmosféricas e o valor de concentração de PM2.5 em determinadas zonas geográficas da cidade de Lisboa?". Para responder a esse questão serão produzidos os seguintes resultados:

1. Um modelo preditivo da evolução das concentrações das partículas PM2.5;
2. Visualização em mapa do modelo preditivo e de informação de contexto.

Com estes entregáveis será possível ter uma ferramenta para apoio à decisão dos responsáveis técnicos e políticos da CML. Onde estes sobre uma linha temporal vão conseguir perceber sobre as regiões de Lisboa o impacto que os factores já mencionados afectam a evolução das concentrações das partículas PM2.5 e que implicações estas têm no dia-a-dia da vida dos seus cidadãos. Também com este facto podem-se reunir para discutir medidas a aplicar de maneira a reduzir estes níveis, e por conseguinte ver se estas medidas tiveram efeitos pois o objectivo proposto irá permitir verificar os dados estatísticos do modelo e um mapa interactivo sobre a cidade.

1.1 Estrutura da dissertação

A estrutura da dissertação está pensada para dar primeiro a conhecer o trabalho relacionado, no capítulo 2, permitindo desenvolver uma visão mais geral relativamente às aplicações e estudos disponíveis semelhantes ao problema em mãos.

[3] https://www.seeedstudio.com/Grove-Laser-PM2-5-Sensor-HM3301.html/

O capítulo 3 tem como objectivo introduzir a arquitectura da solução, começando por descrever uma parte desta referente ao modelo de dados implementado.

As fontes de dados e respectivos serviços relativos à poluição e trânsito é explicado em detalhe no capítulo 4, onde são descritos os dados e eventuais pré-processamento destes assim como a periodicidade de obtenção dos mesmos.

O modelo preditivo de PM2.5 é tratado no capítulo 5, onde se dá a conhecer como este foi construído, os algoritmos utilizados e a decisão da escolha do modelo consoante as métricas obtidas para cada um deles, assim como a necessidade de ter um serviço de previsão do valor de PM2.5.

O *dashboard* interactivo é descrito no capítulo 6, indicando o processo de construção, o proxy com as opções disponibilizadas para o utilizador interagir com este, ou seja, detalha a interface gráfica.

Por fim, o capítulo 7 apresenta um resumo do trabalho desenvolvido, dos objectivos cumpridos e o que pode ser feito num trabalho futuro.

2

Trabalho Relacionado

Para cumprir o objectivo deste trabalho e torná-lo inovador, é necessário realizar um levantamento de estudos realizados no âmbito do tema do projecto. Também, quais os dados modelados bem como que tipo de algoritmos foram usados sobre os mesmos. É importante, realizar uma pesquisa de aplicações e ferramentas usadas em trabalhos relacionados. Desta forma, podemos contribuir na melhoria destas temáticas de forma construtiva e evolutiva.

2.1 Estudos ambientais

A Agência Europeia do Ambiente (do inglês *European Environment Agency*) (EEA) reportou em 2019 um relatório com uma visão geral e uma análise actualizada sobre a qualidade do ar na Europa de 2000 a 2017 [7]. Neste trabalho, podemos observar os progressos realizados no sentido de cumprir os padrões de qualidade do ar estabelecidos nas duas directivas da União Europeia (UE) e as directrizes da OMS. As últimas descobertas e estimativas de exposição da população e do ecossistema aos poluentes do ar com os maiores impactos, também estão presentes neste documento. Estão reportados duas hipóteses de medida para o índice PM2.5, o Indicador de exposição média de concentrações para PM2.5 (do inglês *Average Exposure Indicator for PM2.5 concentrations*) (AEI), que representa a média dos níveis de concentração num período mínimo de 3 anos. A primeira medida é relativa aos anos de 2008–2010 (AEI 2010), e a segunda em 2009–2010 (AEI 2011). Os dados analisados usaram como referência o AEI 2011,

com excepção da Croácia que se baseou no AEI 2015 (média de 2013-2015). 75% dos dados com cobertura mínima, válidos, foram recebidos de 1.396 estações localizadas em todos os países do Espaço Económico Europeu (EEE)-39, com excepção da Albânia, Grécia, Kosovo, Liechtenstein, Montenegro e Sérvia. No ano de 2017, as concentrações de PM2.5 foram superiores ao valor limite anual (25 $\mu g/m^3$) em 7 estados membros e em 3 outros países, valores apresentados nas Figuras 2.1 e 2.2. Num total de 7% das estações foram registados valores acima do valor limite, com predominância (94% dos casos) em áreas urbanas (83%) e suburbanas (11%).

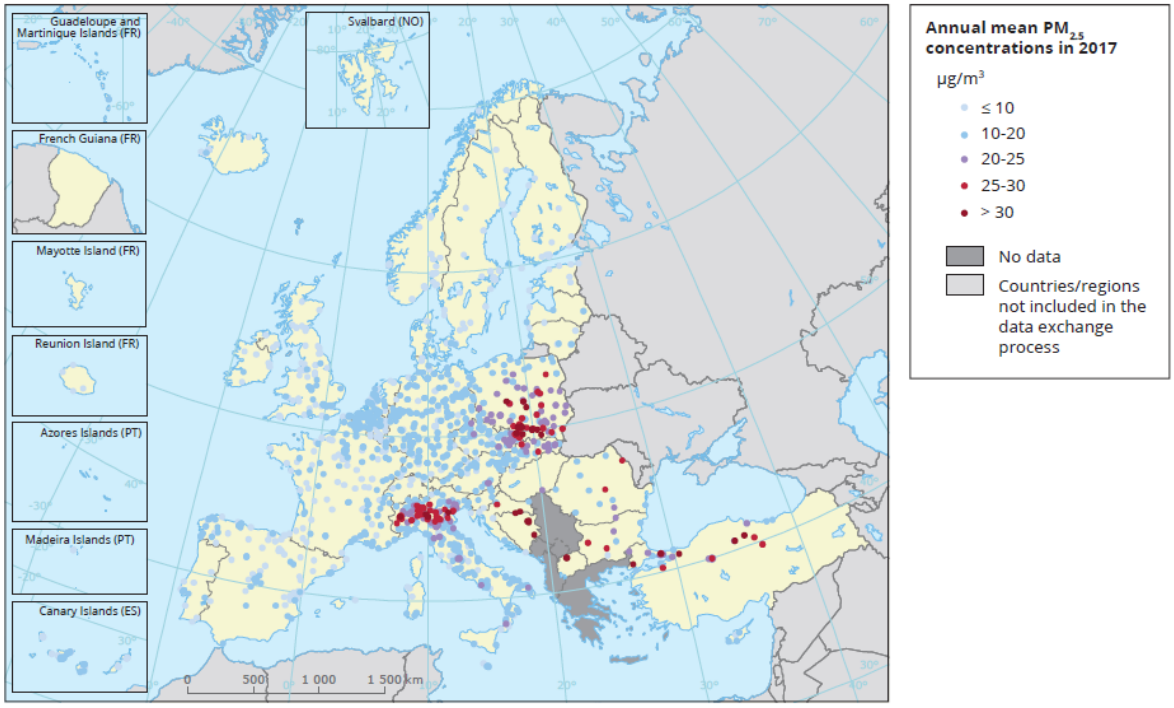


Figura 2.1: Registo anual das concentrações médias de PM2.5 (2017) nos países do EEE39. Os pontos nas duas últimas categorias de cores indicam estações que relatam concentrações acima do valor limite anual da UE. Apenas, foram consideradas estações com mais de 75% dos dados válidos. (Fonte: European Environment Agency [7])

O ano de 2017 foi um dos mais devastadores em termos de incêndios na Europa, com mais de 1,2 milhões de hectares de terra natural queimada em toda a União Europeia (UE) e 127 mortes. O Sistema Europeu de Informação sobre Incêndios Florestais estimou que esses incêndios causavam perdas de cerca de 10 mil milhões de euros. Além do perigo que representam em perdas de vidas e a destruição de recursos naturais e ambientes, o fumo desses incêndios representa um risco substancial para a saúde humana. Oa incêndios florestais emitem grandes quantidades de poluentes, como partículas (PM), óxidos de azoto (NOX), monóxido de carbono (CO), compostos orgânicos

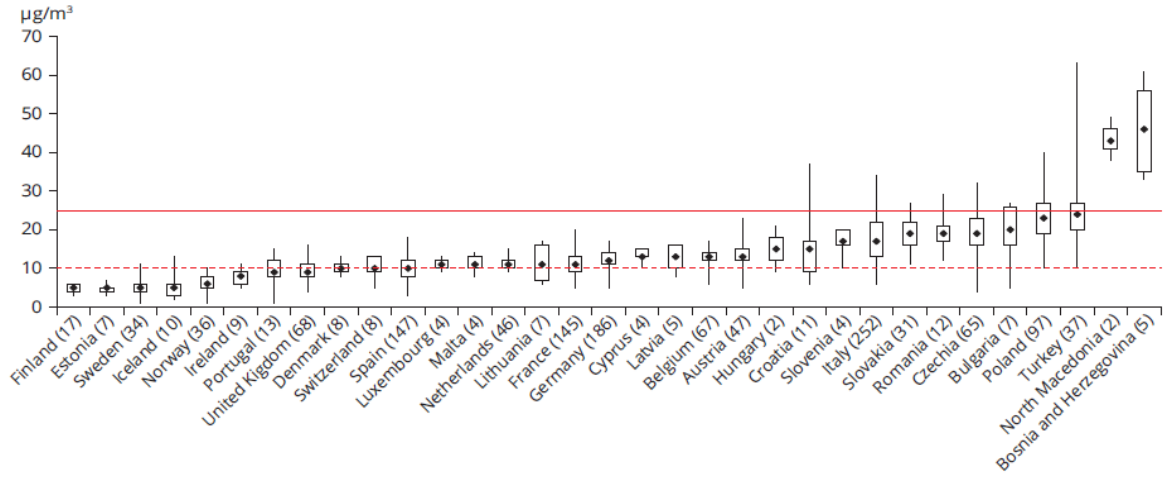


Figura 2.2: Distribuição de valores de concentração média (traço contínuo) e limite (traço ponteado) de PM2.5, em $\mu g/m^3$, nos países do EEE39 (2017). Para cada país, são indicados o número de estações consideradas (entre parêntesis) e os valores mínimo, máximo e médio do percentil 90, 4 registados nas suas estações dos valores de concentração média diária correspondente à 36ª maior média diária. Os rectângulos marcam os percentis 25º e 75º. O gráfico deve ser lido em relação à Figura 2.1, porque há uma dependência do número de estações (Fonte:European Environment Agency [7]).

voláteis compostos (COV) e hidrocarbonetos aromáticos poli-cíclicos (HAP).

Após uma intensa onda de calor em Portugal e condições de extrema secas, verificaram-se incêndios nas áreas montanhosas situadas para o norte/nordeste de Lisboa. O pior incidente ocorreu de 17 a 22 de Junho de 2017 no centro de Portugal, concretamente em Pedrogão Grande. Este incêndio florestal provocou concentrações elevadas de PM2.5 em Portugal ao longo das directrizes diárias da qualidade do ar da OMS em 18 de Junho de 2017. A nuvem de fumo também levou a 20–22 de Junho a níveis elevados de PM2.5 na Galiza e nas Astúrias, no norte da Espanha. Durante o episódio de incêndio, quando centenas de incêndios menores também ocorreram em Portugal e na Espanha, o PM2.5 as concentrações médias diárias, em média na região, aumentaram de 8 para 14 $\mu g/m^3$. A pluma de fogo contribuiu para altas concentrações de PM2.5 observadas em locais franceses e britânicos de 20 a 21 de Junho (quando a PM2.5 aumentou, em média, de 5 para 15 $\mu g/m^3$) e às altas concentrações de PM2.5 observadas na Europa Central e nos locais de monitorização dos Balcãs em 21 e 22 de Junho (quando o PM2.5 aumentou, em média, de 5 para 20 $\mu g/m^3$).

O outro grande incidente ocorreu em Portugal e no noroeste da Espanha, de 13 a 17 de Outubro, quando grandes focos de incêndios florestais, agravados por ventos fortes que foram intensificados pelo furacão Ophelia e pela tempestade de poeira associada,

levaram a mais 49 mortes. A área que foi queimada em Portugal em 2017 foi mais de seis vezes a média para 2007–2016. Os efeitos desse evento, que combinaram as cinzas dos incêndios florestais com a poeira do norte da África provocada pelos remanescentes do furacão Ophelia, puderam ser observados no Reino Unido e na Irlanda, assim como no norte da Europa.

Observações em uma região que abrange todos os EEE39 mostram concentrações de PM semelhantes antes e depois dos dois episódios de incêndios florestais em Junho e Outubro de 2017. No entanto, durante o episódio de incêndios florestais e poeira em Outubro, os níveis médios diários, em média, sobre o EEE39 foram superiores aos do episódio de Junho, sendo 17 e 34 $\mu g/m^3$ para PM2.5. Os níveis médios diários começaram a aumentar em 13 de Outubro e voltaram aos níveis usuais em 20 de Outubro. A pluma do fogo e poeira do norte da África afectou mais locais em toda a Europa. Foi observado um aumento nos níveis de concentrações médias diárias na Europa Central e Oriental a partir de 14 de Outubro e de 16 de Outubro de 2017 para locais na Noruega e na Suécia.

A Figura 2.3 ilustra o indicador de exposição média (AEI) calculado em 2017 (médias 2015–2017) usando as estações designadas para esse fim pelos estados membros (excepto Grécia, Hungria e Noruega). Os pontos mostram as concentrações de PM2.5 nas áreas urbano e suburbano (para estações com pelo menos 75% de cobertura de dados). A linha vertical representa o valor limite de concentração de exposição, para a UE28, fixada em 20 $\mu g/m^3$, a ser atingida até 2015.

Ainda neste relatório, se analisou o impacto na saúde dos vários tipos de poluentes incluindo o PM2.5. Este, é considerado o poluente com maior impacto em termos de mortes prematuras. Os valores foram estimados em países com maior densidade populacional, nomeadamente Alemanha, Itália, Polónia, França e Reino Unido. Contudo, em termos relativos, ao considerar o *Years of Life Lost* (YLL) por 100.000 habitantes, é nos países da Europa Central e Oriental onde se verifica maior influência, são observadas as maiores concentrações, nomeadamente Kosovo, Sérvia, Bulgária, Albânia e Macedónia do Norte. Os menores impactos relativos são encontrados em países situados no norte e noroeste da Europa, como Islândia, Noruega, Suécia, Irlanda e Finlândia. As Figuras 2.4 e 2.5 mostram estes factos.

Mesquita [16], na sua dissertação de doutoramento, fez uma modelação da distribuição espacial da qualidade do ar na cidade de Lisboa usando sistemas de informação geográfica. Os dados analisados foram o dióxido de azoto (NO_2) da rede fixa de estações de monitorização de qualidade do ar e, de 2 campanhas com cerca de 100 tubos de difusão (método de amostragem por difusão passiva numa malha sistemática

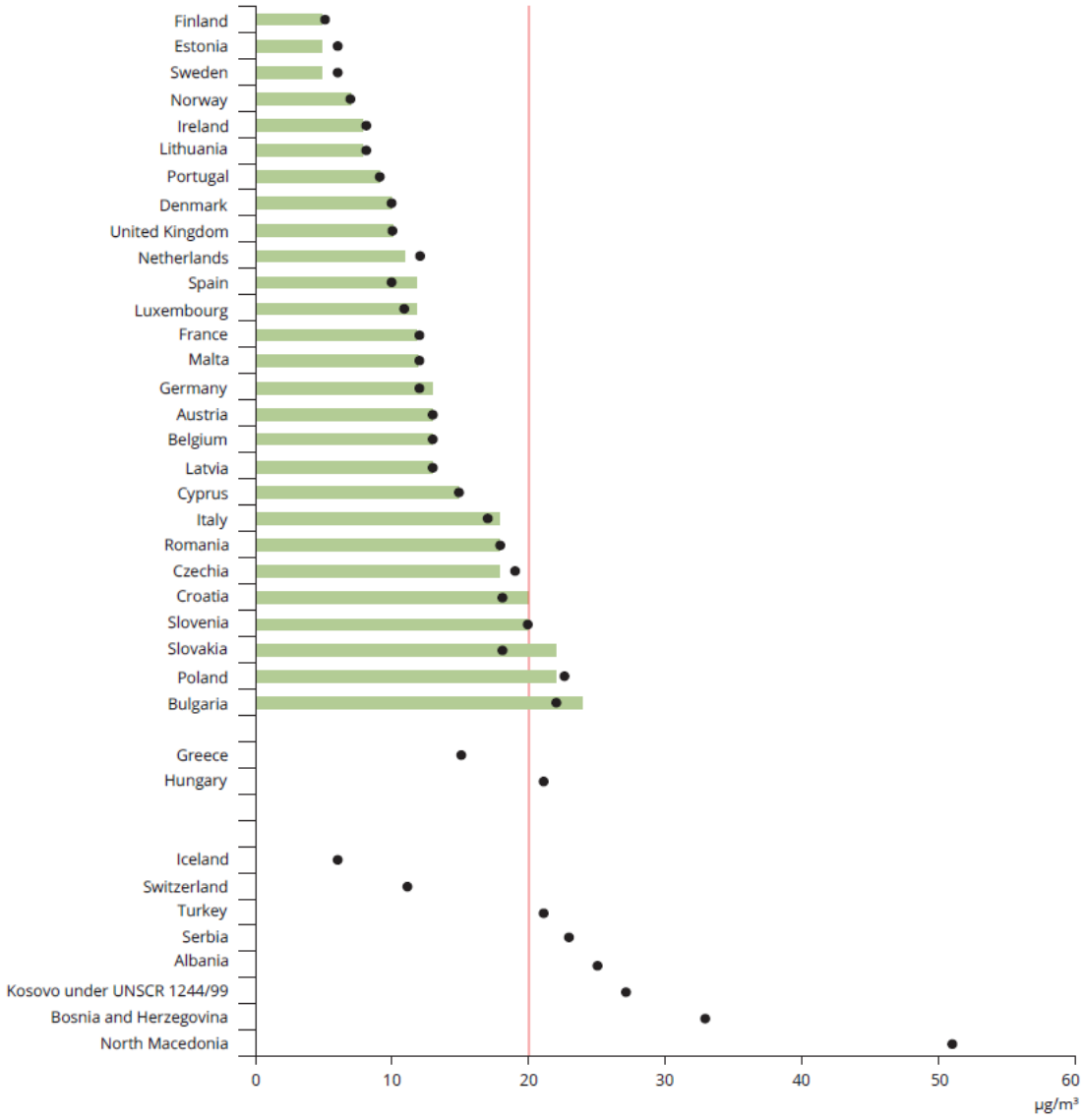


Figura 2.3: Indicador de exposição das concentrações de PM2.5. As barras representam o indicador de exposição média (AEI) calculado em 2017 (média de 2015–2017). Os pontos mostram todas as concentrações médias de PM2.5 em áreas urbanas e suburbanas (média de 2015–2017). A linha vertical representa a concentração limite de exposição a UE28, 20 $\mu g/m^3$, definida até 2015 (Fonte:European Environment Agency [7]).

		$PM_{2.5}$		NO_2		O_3	
Country	**Population (1 000)**	**Annual mean (a)**	**Premature deaths (b)**	**Annual mean (a)**	**Premature deaths (b)**	**SOMO35 (a)**	**Premature deaths (b)**
Austria	8 700	12.0	5 300	18.9	1 000	4 522	270
Belgium	11 311	12.7	7 600	21.7	1 600	2 203	180
Bulgaria	7 154	22.3	13 100	18.8	1 100	3 347	280
Croatia	4 191	19.4	5 300	15.2	260	4 996	190
Cyprus	1 184	13.7	580	24.0	240	5 612	30
Czechia	10 554	16.6	9 600	15.2	240	4 353	350
Denmark	5 707	9.2	2 700	10.4	80	2 293	90
Estonia	1 316	5.9	500	7.8	< 1	1 949	20
Finland	5 487	5.1	1 500	8.0	< 1	1 510	60
France	64 977	10.9	33 200	17.3	7 500	3 420	1 400
Germany	82 176	11.6	59 600	20.2	11 900	3 368	2 400
Greece	10 784	19.6	12 900	19.6	2 900	6 871	640
Hungary	9 830	17.5	12 100	16.6	770	3 952	380
Ireland	4 726	6.8	1 100	11.0	50	1 323	30
Italy	60 666	16.6	58 600	22.1	14 600	6 058	3 000
Latvia	1 969	10.9	1 700	12.0	60	2 773	60
Lithuania	2 889	11.8	2 600	11.7	20	2 456	70
Luxembourg	576	11.4	230	20.7	50	2 211	10
Malta	450	11.1	210	14.9	< 1	5 985	20
Netherlands	16 979	11.3	9 200	20.5	1 500	2 428	270
Poland	37 967	20.6	43 100	15.2	1 500	3 699	1 100
Portugal	9 809	8.3	4 900	15.3	610	4 074	320
Romania	19 761	16.8	23 400	17.6	2 600	2 485	490
Slovakia	5 426	17.6	4 800	13.5	20	4 232	160
Slovenia	2 064	16.0	1 700	15.4	70	5 007	70
Spain	44 145	11.1	24 100	20.0	7 700	5 212	1 500
Sweden	9 851	5.7	2 900	10.7	30	1 819	120
United Kingdom	65 379	9.5	31 800	21.8	11 800	1 161	530
Albania	2 876	22.3	5 100	13.7	70	5 475	180
Andorra	73	12.1	40	18.2	< 1	4 423	< 5
Bosnia and Herzegovina	3 516	28.7	5 400	13.2	20	4 409	120
Iceland	333	4.8	60	10.1	< 1	499	< 5
Kosovo	1 772	27.1	3 800	14.4	20	4 769	100
Liechtenstein	38	10.3	20	17.8	< 1	4 945	< 5
Monaco	38	14.3	30	26.8	10	7 186	< 5
Montenegro	622	20.3	630	11.9	< 1	5 269	20
North Macedonia	2 071	34.6	3 400	17.4	110	4 434	70
Norway	5 211	5.9	1 300	12.4	130	1 502	50
San Marino	33	14.3	30	16.3	< 1	5 667	< 5
Serbia	7 076	24.6	13 700	19.4	1 500	3 508	280
Switzerland	8 327	10.1	3 700	19.7	620	4 842	240
EU-28	**506 028**	**12.9**	**374 000**	**16.3**	**68 000**	**3 547**	**14 000**
Total	**538 014**	**14.4**	**412 000**	**16.3**	**71 000**	**3 811**	**15 100**

Notes: (a) The annual mean (in µg/m³) and the SOMO35 (in µg/m³-days), expressed as population-weighted concentration, is obtained according to the methodology described by ETC/ACM (2019) and references therein and not only from monitoring stations.

(b) Total and EU-28 premature deaths are rounded to the nearest thousand (except for O_3, nearest hundred). The national totals are rounded to the nearest hundred or ten.

Figura 2.4: Quadro de mortes prematuras atribuídos à exposição de PM2.5, NO_2 e O_3, nos 41 países europeus e na UE28 no ano de 2016

	$PM_{2.5}$		NO_2		O_3	
Country	**YLL**	**YLL/10^5 inhabitants**	**YLL**	**YLL/10^5 inhabitants**	**YLL**	**YLL/10^5 inhabitants**
Austria	52 000	598	10 400	120	2 800	32
Belgium	75 800	670	16 400	145	1 900	17
Bulgaria	32 900	1 858	10 800	151	3 000	42
Croatia	51 100	1 219	2 500	60	1 900	45
Cyprus	5 600	473	2 300	194	340	29
Czechia	101 000	957	2 500	24	3 800	36
Denmark	27 800	487	870	15	990	17
Estonia	5 400	410	< 5	< 1	250	19
Finland	15 500	282	< 5	< 1	630	11
France	353 000	543	79 500	122	16 100	25
Germany	591 400	720	118 100	144	24 400	30
Greece	126 100	1 169	27 900	259	6 500	60
Hungary	130 000	1 322	8 300	84	4 200	43
Ireland	12 000	254	560	12	350	7
Italy	550 600	908	137 500	227	29 100	48
Latvia	17 300	879	660	34	630	32
Lithuania	26 400	914	180	6	790	27
Luxembourg	2 500	434	490	85	70	12
Malta	2 400	533	< 5	< 1	190	42
Netherlands	92 500	545	14 700	87	2 900	17
Poland	517 700	1 364	18 500	49	13 800	36
Portugal	46 000	469	5 700	58	3 200	33
Romania	252 400	1 277	27 800	141	5 600	28
Slovakia	55 200	1 017	270	5	2 000	37
Slovenia	18 900	916	810	39	840	41
Spain	244 000	553	77 800	176	16 300	37
Sweden	25 000	254	240	2	1 100	11
United Kingdom	317 600	486	117 500	180	5 600	9
Albania	50 400	1 753	730	25	1 700	59
Andorra	440	602	< 5	< 1	20	27
Bosnia and Herzegovina	58 100	1 652	250	7	1 300	37
Iceland	560	168	< 5	< 1	10	3
Kosovo	37 200	2 100	240	14	910	51
Liechtenstein	200	532	5	< 1	10	27
Monaco	270	707	120	314	20	52
Montenegro	7 400	1 189	< 5	< 1	300	48
North Macedonia	35 200	1 699	1 100	53	760	37
Norway	12 100	232	1 200	23	440	8
San Marino	260	788	< 5	< 1	10	30
Serbia	135 800	1 919	14 800	209	2 900	41
Switzerland	36 500	438	6 100	73	2 500	30
EU-28	**3 848 000**	**800**	**682 000**	**100**	**149 000**	**30**
Total	**4 223 000**	**900**	**707 000**	**100**	**160 000**	**30**

Note: Total and EU-28 YLL figures are rounded to the nearest thousand or hundred. National data are rounded to the nearest hundred or ten.

Figura 2.5: Quadro de *Years of Life Lost* (YLL) atribuídos à exposição de PM2.5, NO_2 e O_3 em 41 países europeus e UE

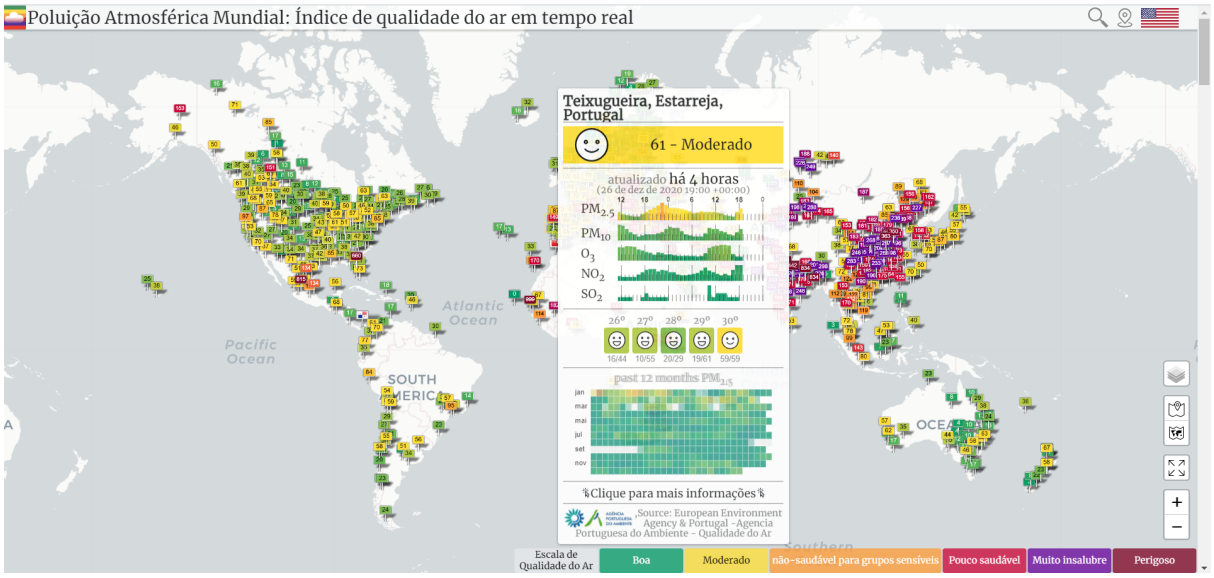


Figura 2.6: Aplicação poluição atmosférica mundial: índice de qualidade do ar em tempo real

de 20 × 20km). As técnicas de interpolação aplicadas foram Inverso da potência das distâncias (do inglês *Inverse Distance Weighting*) (IDW), *Ordinary Kriging method*, conhecido por "Processo Gaussiano de Regressão" (OK), *Ordinary coKriging* (OcK) e Regressão Linear Múltipla (RLM). Os resultados aqui obtidos permitiram concluir que a nível local a RLM é o método de interpolação mais adequado para obter mapas de alta resolução relativos a poluentes relacionados com o tráfego rodoviário. Dos métodos de interpolação testados foi o único que permitiu identificar a população e áreas expostas a níveis de poluição superiores ao permitido legalmente. As limitações encontradas neste estudo , e por consequente trabalho futuro, consiste em aplicar a metodologia a outros poluentes como o PM10 e O_3, a outras áreas urbanas da Região de Lisboa e Vale do Tejo à escala nacional para uma grelha de $1 \times 1km^2$.

Actualmente, encontramos algumas plataformas em que é possível mapear o nível de gases e partículas. Comecemos pela "Qualité de l'air à Paris" [21]. Nesta plataforma é possível observar sobre um mapa o nível de alguns gases (NO_2 e O_3) e também PM (2.5 e 10), num período temporal. Os dados são retirados da agência Airparif. Esta plataforma tem 3 mapas, o primeiro é actualizado a cada hora e permite ver, quase em tempo real, o estado da poluição do ar na capital com um código de cores que vai do verde turquesa ao vermelho escuro dependendo da gravidade. O segundo mapa é uma ferramenta de previsão: são exibidas as estimativas do Airparif. O terceiro, numa versão beta apresenta uma grelha mais precisa da cidade. Baseado num sistema

diferente designado por "Polutrack". Conta com 400 carros eléctricos[1], que percorrem a cidade com sensores a laser capazes de captar PM2.5. Estes carros, transmitem cerca de 2 milhões de leituras diárias, sendo a apresentação no mapa a média dos valores acumulados dos últimos 7 dias permitindo localizar as zonas mais poluídas a médio prazo.

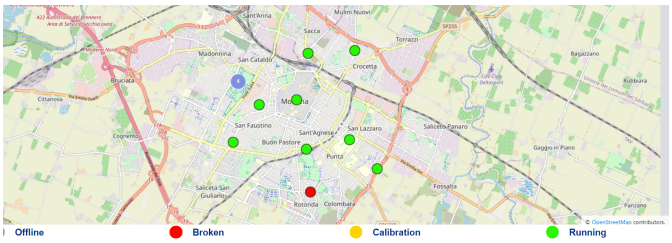
Outra aplicação analisada foi a "Poluição Atmosférica Mundial: Índice de qualidade do ar em tempo real" [28] (Figura 2.6) em que é apresentado sobre um mapa, o valor dos alguns gases e partículas para diversas localizações do mundo. Neste caso também são mostrados valores meteorológicos (e.g. vento e humidade). Esta começa por apresentar um mapa mundo cheio de pontos marcados, estes com um rectângulo que contem o valor numérico do respectivo poluente em análise. Na parte inferior da aplicação contém uma escala de cores a variar desde boa representado por verde, a perigoso representado a vermelho escuro estas ainda permite ter uma breve descrição no sentido de visar o risco para a saúde de cada um destes níveis. No canto inferior direito esta apresenta um conjunto de interacções como aumentar ou diminuir o *zoom* no mapa, mostrar a estação de recolha dos dados mais próxima da nossa localização actual, colocar o mapa na representação inicial e escolher o poluente que quer analisar como CO, SO_2, NO_2, O_3 e PM (2.5 e 10). Ao escolhermos a estação pretendida iremos ser redireccionados para uma nova página da aplicação. Nesta iremos ver mas detalhadamente os valores deste poluentes com uma janela deslizante sobre as ultimas 48 horas, incluindo como informação adicional os dados meteorológicos ai registados. No final desta página é fornecido uma previsão sobre os próximos 4 dias mas neste caso referente apenas ao PM2.5 e aos dados meteorológicos.

Uma outra aplicação, também esta com um objectivo semelhante, de nome "Air Quality Dashboard"[29] desenvolvida no âmbito de um projecto europeu, por um consórcio de 4 universidades em Itália e Espanha e 4 organizações publicas. O projecto disponibiliza uma aplicação web que, após o *login* do utilizador, uma página inicial (Figura 2.7(a)) onde é possível escolher as 3 cidades disponíveis e que tipo de funcionalidade se pretende analisar para cada uma destas.

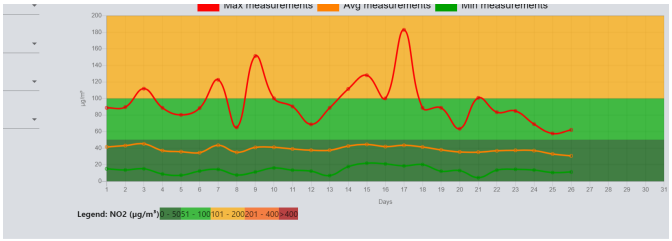
As funcionalidades em questão são o mapa de sensores (Figura 2.7(b)) onde se pode verificar os vários estados destes e a informação recolhida pelos mesmos, os dados estatísticos (Figura 2.7(c)) sobre os poluentes recolhidos segundo uma distribuição temporal, um mapa interactivo (Figura 2.7(d)) onde se pode ver segundo uma escala e o poluente pretendido a variação do mesmo dentro da cidade, e por fim um mapa (Figura 2.7(e)) que pretende dar ao utilizador a possibilidade de seleccionar uma hora do

[1] Master Géographies Numériques Hélène CRETOIS "La carte interactive en temps réel d'Airparif, la géomatique au service de nos poumons"

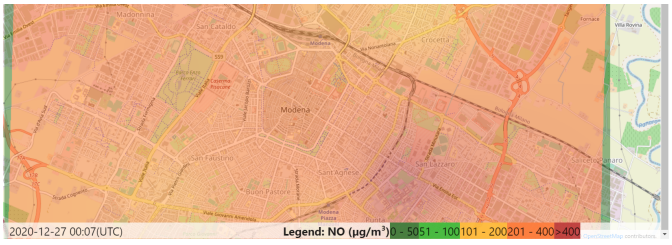
(a) Página inicial da aplicação trafair



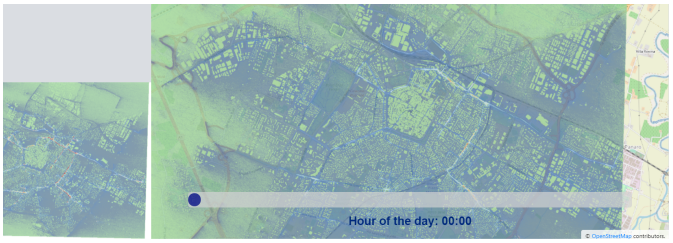
(b) Mapa de sensores na aplicação trafair



(c) Estatísticas sobre os poluentes na aplicação trafair



(d) Mapa interactivo sobre os poluentes na aplicação trafair



(e) Previsão da qualidade do ar de concentração de NOx para o dia seguinte, aplicação trafair

Figura 2.7: Aplicação trafair

dia e observar o valor previsto para o poluente ao longo da cidade.

3

Arquitectura da Solução

Neste capítulo descreve-se a arquitectura de solução ilustrada na Figura 3.1 e o modelo de dados georreferenciados desta. Na arquitectura realçam-se os principais componentes, nomeadamente:

1. Uma aplicação cliente, em ambiente Web (Figura 3.1-Ⓗ), descrito no capítulo 6;
2. Um serviço para obter de uma fonte externa os dados de poluição (PM2.5) recolhidos diariamente (Figura 3.1-Ⓑ);
3. Um serviço para obter dados com informação dos congestionamentos de uma fonte externa (Figura 3.1-Ⓓ);
4. Fonte de dados externa para obtenção de PM2.5 (Figura 3.1-Ⓐ);
5. Fonte de dados de trânsito (Figura 3.1-Ⓒ);
6. Modelo de dados georreferenciados (Figura 3.1-Ⓔ);
7. Um modelo preditivo do valor de poluição para cada área de interesse (Figura 3.1-Ⓕ), descrito no capítulo 5;
8. Um serviço para a preparação e transposição dos dados para uma representação em mapa (Figura 3.1-Ⓖ).

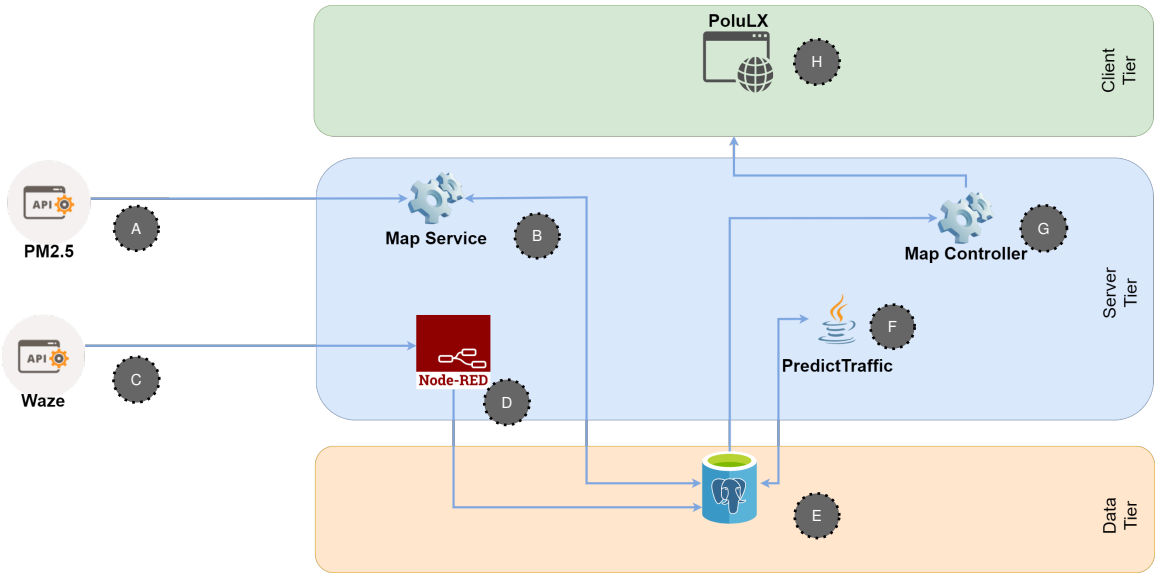


Figura 3.1: Arquitectura da solução

3.1 Modelo de dados georreferenciados

Os dados recolhidos das fontes externas foram armazenados sem se efectuarem transformações drásticas à sua representação original. No entanto, estando a trabalhar com dados georreferenciados, é desejável usar um repositório que tenha implementados tipos geográficos e algoritmos para a sua manipulação. Como iremos descrever nesta secção, em diversos pontos no sistema desenvolvido foi necessário trabalhar os dados para os adequar ao objectivo pretendido, por exemplo, para a interface com o utilizador. Nesse sentido, foi usado o sistema de gestão de base de dados PostgreSQL com extensão PostGIS, para o suporte de dados georreferenciados.

Como ponto de partida à criação do modelo foi necessário criar a base de dados `mapbd` que contém tudo o que o projecto necessita desde tabelas a funções. Para dar suporte à criação de tabelas e funções existem *scripts* com as suas definições de maneira a que a base de dados possa estar actualizada e conter tudo o queseja necessário, estes estão separados consoante o seu propósito. Por exemplo, o *script* `01_script_CRIAR_MODELO` para criar o modelo ou seja as tabelas e suas restrições, o *script* com funções para lidar com a grelha e o mapa, `02_script_CRIAR_FUNCOES_GRELHA_MAPA`, e funções para lidar com os dados do trânsito, `03_script_WAZE` .

Os dados relativos à grelha foram criados e guardados uma única vez, são dados estáticos utilizados como referencia para outros, neste caso para os dados da recolha

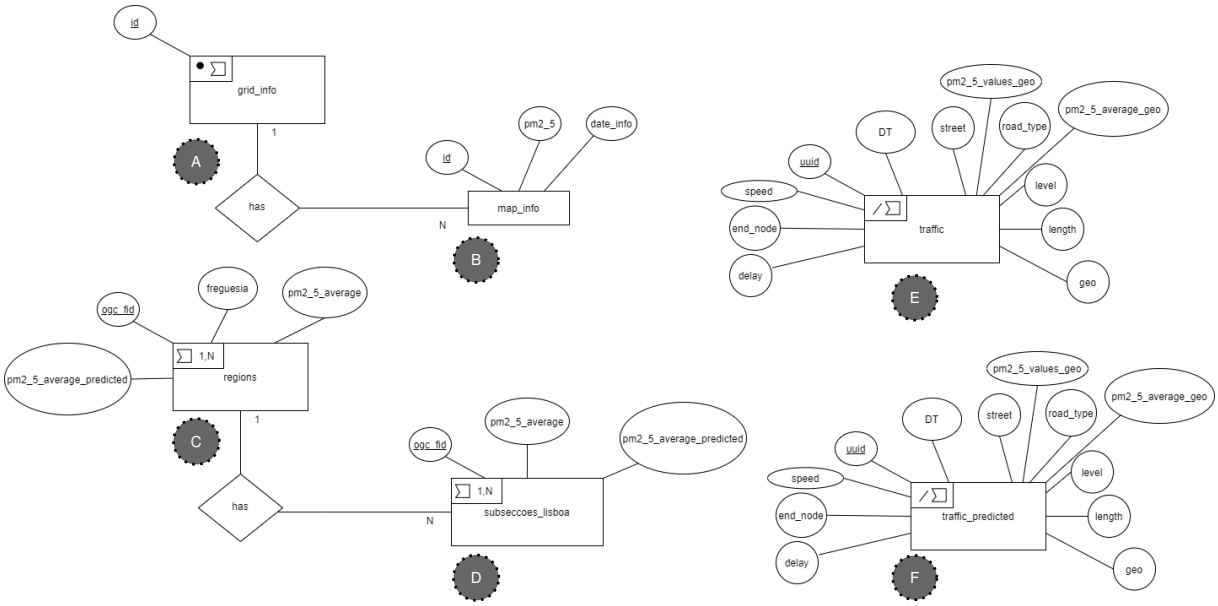


Figura 3.2: Modelo entidade-associação relativo aos dados da aplicação usando pictogramas [2], uma representação estendida para dados geográficos

diária do poluente PM2.5. Para se construírem estes foi necessário criar uma função utilitária denominada `makegrid_2d`, presente na Listagem 3.1, que, como o seu nome indica, é responsável por criar a grelha sobre a área da cidade de Lisboa. Esta função tira partido das funções do PostGIS como `ST_XMin`, `ST_XMax`, `ST_YMin`,`ST_YMax`, `ST_MakePoint`, `ST_Project`, `ST_MakeEnvelope` e `ST_Collect`. Esta recebe a *Bounding Box* da cidade de Lisboa e o comprimento e largura de cada grelha, neste caso 100×100 m^2. A lógica associada a esta função começa por extrair da *Bounding Box* as coordenadas X mínima e máxima através das funções `ST_XMin` e `ST_XMax` respectivamente, fazendo o mesmo para as coordernadas Y mínima e máxima, usando as funções `ST_YMin` e `ST_YMax`, respectivamente. De seguida mantém um ciclo enquanto a coordenada Ymin <= Ymax o mesmo para o Xmin <= Xmax, com isto cria um ponto `ST_MakePoint` com Xmin e Y min e a partir deste projecta outros dois `ST_Project` segundo a distância definida de 100 m^2 e cria um polígono `ST_MakeEnvelope` que corresponde à grelha. Cada um destes polígonos serão armazenados numa lista para que no final, com o auxilio da função `ST_Collect`, seja criado uma colecção de polígonos. Essa é o retorno da função.

Com esta função já se tinha forma de obter os polígonos que formam a grelha e seus pontos centrais, portanto procedeu-se à criação da tabela como consta na Figura 3.2-Ⓐ. Esta contem um identificador gerado automaticamente, um `f_id`, dado criado à posterior para facilitar a referência para a tabela do registo do poluente, `g_grid` o

polígono que representa a grelha e `g_grid_point` o ponto central de cada grelha. Para facilitar a inserção, foi implementada a função `insert_grid_info`, presente na Listagem 3.2, que insere na tabela `grid_info` os dados para esta, tirando partindo da função `makegrid_2d` para obter os polígonos sobre a grelha necessários.

```
1  CREATE OR REPLACE FUNCTION public.makegrid_2d (
2    bound_polygon public.geometry,
3    width_step integer,
4    height_step integer
5  )
6  RETURNS public.geometry AS
7  $body$
8  DECLARE -- Declare variables
9  BEGIN -- inicialize variables with ST_XMin, ST_XMax, ST_YMax, ST_SRID
10   Y := ST_YMin(bound_polygon); --current sector's corner coordinate
11   i := -1;
12   <<yloop>>
13   LOOP
14     IF (Y > Ymax) THEN
15         EXIT;
16     END IF;
17
18     X := Xmin;
19     <<xloop>>
20     LOOP
21       IF (X > Xmax) THEN
22           EXIT;
23       END IF;
24
25       CPoint := ST_SetSRID(ST_MakePoint(X, Y), SRID);
26
27       NextX := ST_X(ST_Project(CPoint, $2, radians(90))::geometry);
28       NextY := ST_Y(ST_Project(CPoint, $3, radians(0))::geometry);
29
30       i := i + 1;
31       sectors[i] := ST_MakeEnvelope(X, Y, NextX, NextY, SRID);
32
33       X := NextX;
34     END LOOP xloop;
35     CPoint := ST_SetSRID(ST_MakePoint(X, Y), SRID);
36     NextY := ST_Y(ST_Project(CPoint, $3, radians(0))::geometry);
37     Y := NextY;
38   END LOOP yloop;
39
40   RETURN ST_Collect(sectors);
41 END;
42 $body$
43 LANGUAGE 'plpgsql';
```

Listagem 3.1: Função `makegrid_2d` para criação dos poligonos da grelha sobre o mapa

```
1  CREATE OR REPLACE FUNCTION insert_grid_info()
2    RETURNS void as
3    $BODY$
4    DECLARE --declare variables
5    BEGIN
6      IF NOT EXISTS (SELECT id FROM public."grid_info") THEN
7        INSERT INTO public."grid_info" (g_grid, g_grid_point)  SELECT ST_SetSRID(q.cell, 4326)
             AS pol, ST_Centroid(ST_SetSRID(q.cell, 4326)) AS g_point
8                    FROM (
9                    SELECT (
10                     ST_Dump(
11                     makegrid_2d(
12                     ST_AsText(ST_GeomFromGeoJSON('{
13                       "type": "Polygon",
14                       "coordinates":
15                                                [[[-9.22983565,38.69139935],
16                                                [-9.08633286,38.69139935],
17                                                [-9.08633286,38.79675837],
18                                                [-9.22983565,38.79675837],
19                                                [-9.22983565,38.69139935]]]
20                     }')),
21                     100, -- width step in meters
22                     100  -- height step in meters
23                   )
24                   )
25                   ) .geom AS cell
26                   )q;
27     END IF;
28   END;
29   $BODY$
30 LANGUAGE 'plpgsql';
```

Listagem 3.2: Descrição da função `insert_grid_info`

De seguida, após se ter a informação relativa à grelha, criou-se uma tabela `map_info` (Figura 3.2-Ⓑ) com o propósito de guardar a informação relativa ao poluente PM2.5, cuja obtenção está descrita na Secção 4.2. Esta tabela começa por ser constituída, por um identificador automático, o valor do PM2.5 obtido, a data em que este foi registado, e uma referencia para a grelha e seu ponto central correspondente ao registo.

Na visualização dos dados, é necessário apresentar os valores registados do PM2.5 em diferentes níveis de granularidade, nomeadamente para as freguesias e secção/subsecção estatísticas do INE. No caso das freguesias, os dados com os limites geográficos foram obtidos da plataforma Lisboa aberta[3], que disponibiliza um catálogo de dados de livre acesso sobre vários contextos da cidade de Lisboa, desde o ambiente à cultura ou desporto e, entre outros. O conjunto de dados dom os limites geográficos das freguesias é disponibilizado em GeoJSON. Foi feita uma verificação dos dados obtidos na ferramenta de código aberto QGIS[4] que permite visualizar e manipular camadas

[3] http://lisboaaberta.cm-lisboa.pt/index.php/pt/
[4] https://qgis.org/pt_PT/site/

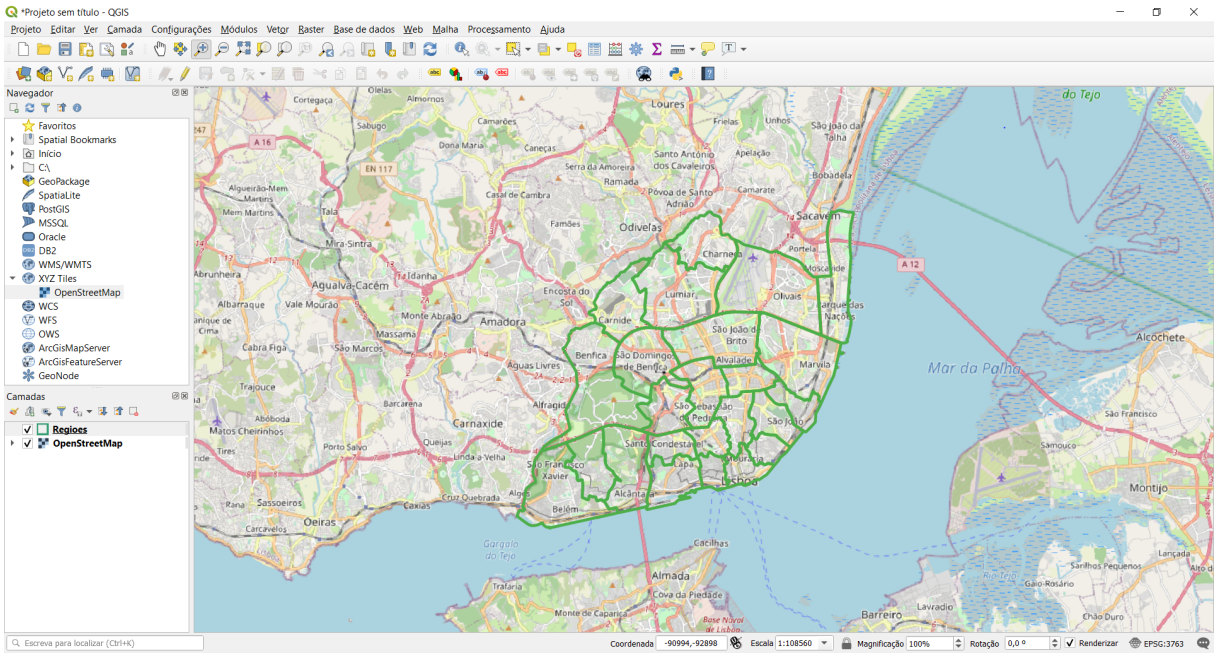


Figura 3.3: Visualização dos dados obtidos relativos às freguesias tirando partido do QGIS. Os limites geográficos estão representados a verde

de informações geográficas. A Figura 3.3 mostra a relação entre os dados obtidos e a sua localização através do QGIS usando um mapa base, e adicionando a este a camada com os dados a confirmar. Para poder representar os níveis de PM2.5 por cada freguesia, é necessário ter esta informação disponível na base de dados. Foi usada a facilidade de exportação do QGIS, que permite pegar numa camada e guardá-la noutro formato, neste caso um ficheiro do tipo *sql*, para ser importado para a nossa base de dados PostgreSQL. Os dados foram colocados na tabela com o nome `regions`, com um conjunto de atributos, nomeadamente, um identificador único, o dado geográfico que corresponde ao polígono que representa a área da freguesia, e um outro conjunto de informações sobre a freguesia como o nome desta, conselho, distrito, a sua área, entre outros sendo que os mais importantes e usados para o problema foram o identificador, o polígono e o nome da freguesia.

Com esta informação presente na base de dados, já é possível cruzá-la com a grelha e com o histórico de medidas do poluente PM2.5. Teve-se de pensar como se iria chegar ao valor de PM2.5 para uma dada freguesia, sendo que cada mediação consiste numa média deslizante dos últimos 7 dias. Como para uma freguesia poderíamos ter mais do que um registo deste valor, ou seja, vários pontos respectivos aos polígonos da grelha poderiam estar contidos na região geográfica das freguesia. Assim, a solução encontrada foi aplicar uma média destes por cada freguesia. Sendo assim o primeiro

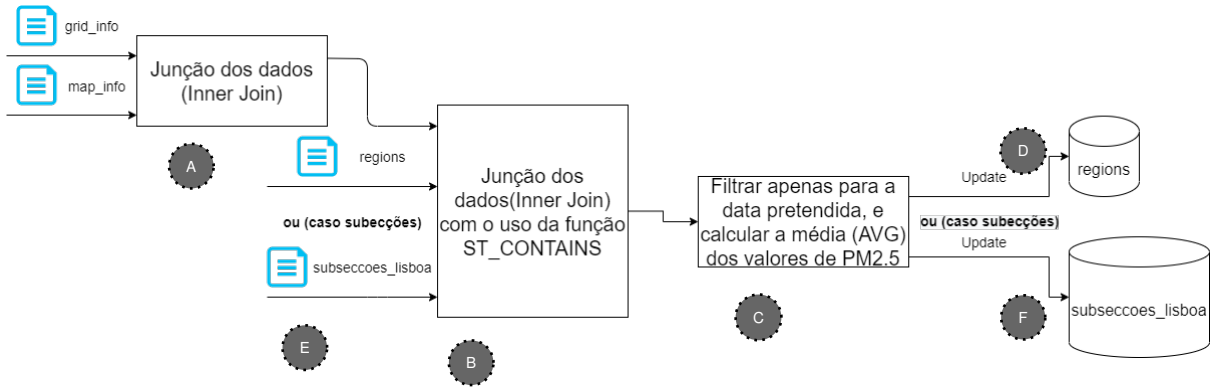


Figura 3.4: Processo para calcular o valor médio de PM2.5 para as freguesias e subsecções-estatísticas

passo foi acrescentar à tabela das `regions` uma coluna para conter este valor, com a vantagem de termos a extensão PostGIS na base de dados onde se encontra esta informação, foi permitido tirar proveito mais uma vez das funções que permite lidar com este tipo de dados. Criou-se a função `calc_avg_pm2_5_regions`, presente na Listagem 3.3, que como o nome sugere calcula o valor médio dos pontos para cada freguesia, esta recebe como parâmetro uma data, data essa que corresponde ao dia pretendido para que seja calculado o valor média para as freguesias.

```
1  CREATE OR REPLACE FUNCTION calc_avg_pm2_5_regions(data_date timestamp)
2    RETURNS void as
3    $BODY$
4    DECLARE
5    BEGIN
6      UPDATE regions
7      SET pm2_5_average = (SELECT AVG(M.pm2_5)
8                FROM map_info AS M INNER JOIN grid_info AS G
9                ON M.f_id = G.f_id INNER JOIN regions AS R
10                 ON (ST_Contains(R.wkb_geometry, G.g_grid_point) AND R.ogc_fid = regions.
                       ogc_fid) WHERE M.date_info = data_date);
11   END;
12   $BODY$
13 LANGUAGE 'plpgsql';
```

Listagem 3.3: Descrição da função `calc_avg_pm2_5_regions`

A função primeiramente começa por cruzar (Figura 3.4-Ⓐ) a informação da tabela `grid_info` esta que contém os polígonos sobre a grelha e respectivos pontos com a tabela `map_info` que guarda os valores de PM2.5 diariamente. Posto isto, pode cruzar-se (Figura 3.4-Ⓑ) a informação dos polígonos relativos à grelha com a da tabela `regions` com a condicionante que era agrupado por freguesia e pontos que tivessem contidos nessa, passo este que é possível com a função `ST_Contains` que tem como definição, dado duas geometrias "A" e "B", a geometria A contém a geometria B se

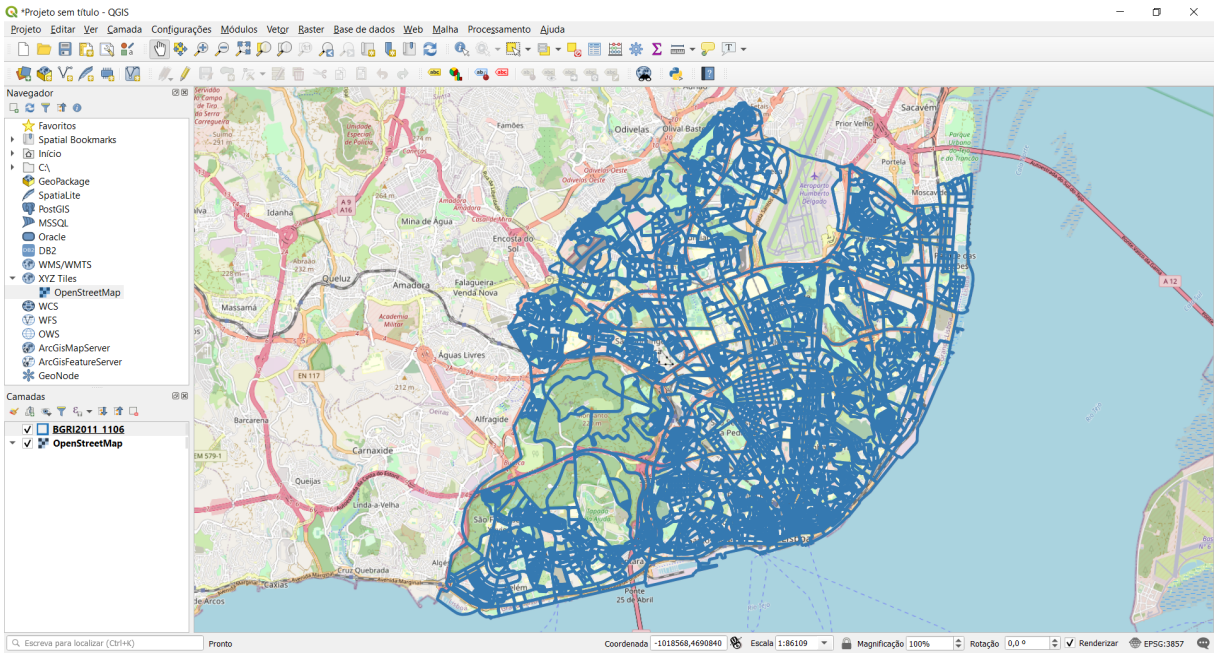


Figura 3.5: Visualização dos dados obtidos relativos à secção/subsecção estatística tirando partido do QGIS

e somente se nenhum ponto de B está no exterior de A, e pelo menos um ponto do interior de B está no interior de A. Neste caso sendo a freguesia representada geograficamente por um polígono e o registo do poluente referente a um ponto geográfico respectivamente isto é válido por cada ponto que estiver contido no polígono. Por fim, é necessário garantir que estas condições são aplicadas apenas para uma data especifica (Figura 3.4-C), para que esta função dê apenas os resultados para um dia e assim aplicar a média do PM2.5 com estes passos todos assegurados, actualizando assim o dado relativo ao poluente na tabela `regions` (Figura 3.4-D).

A informação acerca da secção/subsecção estatística da cidade de Lisboa permite detalhar as análises no interior da área envolvente de cada freguesia. Obteve-se a informação acerca da secção/subsecção estatística dos dados relativos aos censos, fornecidos pelo INE [5]. Estes vêm num formato diferente do das freguesias, em *Shapefile*[6], formato também suportado pela ferramenta QGIS e ilustrado na Figura 3.5. Nesta verifica-se então um maior número de áreas delimitadas, agrupadas pelas freguesias. Para importar os dados para a base de dados, usou-se um processo quase idêntico ao das freguesias, exportando-se a camada de modo a que esta consiga ser importada na base de

[5]http://mapas.ine.pt/download/index2011.phtml
[6]https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf

dados. Criou-se uma tabela com o nome `subseccoes_lisboa`, que contém um identificador único, a informação geográfica que corresponde ao polígono que representa a área da secção/subsecção estatística, e um outro conjunto de informações descritivas que não foram usadas na solução do problema. Sendo que o propósito também é saber o valor registado para o poluente PM2.5 para um certo dia foi igualmente criado um novo atributo à tabela para guardar esse valor. De modo a facilitar a análise e pesquisa dos dados foi criado um atributo para conter o identificador único da freguesia, de modo a que cada secção fique com esse identificador consoante a cada freguesia a que esta pertença. Para tal, foi criado a função `update_id_regions` que cruza os dados das freguesias com as secções, tirando partido mais uma vez da função `ST_Contains`, que verifica se o polígono da secção está contido no polígono da freguesia; caso se confirme este fica com o identificador da respectiva freguesia.

Da mesma maneira que se procedeu para as freguesias, foi criada uma função para guardar o valor médio do PM2.5, `calc_avg_pm2_5_subseccoes`, para cada secção. Por isso a única diferença é que os dados a serem cruzados são os das secções (Figura 3.4-Ⓔ), sendo que os dados geográficos a serem comparados são os pontos dos polígonos da grelha com os polígonos das secções para o dia pretendido, actualizando assim o dado relativo ao poluente na tabela `subseccoes_lisboa` (Figura 3.4-Ⓕ).

A Figura 3.2 representa o modelo entidade-associação relativo as tabelas anteriormente explicadas, apresenta no entanto duas tabelas `traffic` e `traffic_predicted` que não foram explicadas o processo de construção destas, este irá ser explicito na Secção 5.1 visto que o propósito destas estar ligado à criação do modelo para aprendizagem automática de modo a que no fim se consiga tirar previsões com estes dados.

4

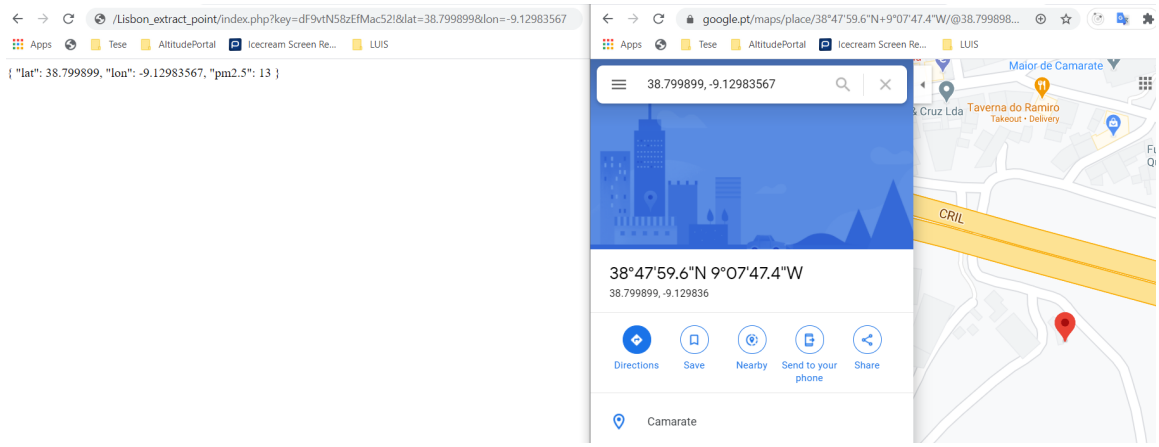
Fontes de dados e Serviços de recolha de dados

Com a solução apresentada, os utilizadores finais podem monitorizar valores do poluente PM2.5, conseguindo identificar os locais na cidade onde os valores estão acima do desejado. Assim, os decisores têm hipótese de tomar medidas no sentido de melhorar a qualidade do ar nesses locais, permitindo atingir o compromisso ambiental assumido pela CML.

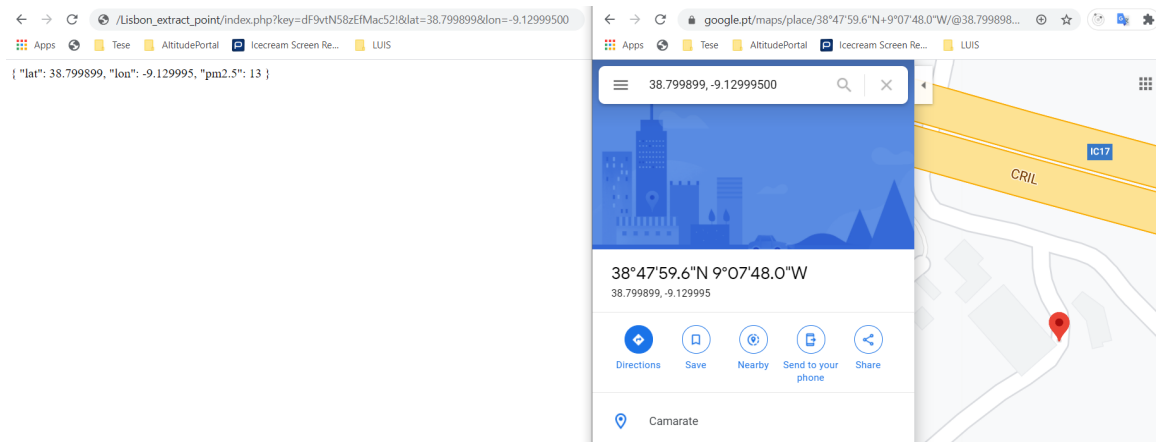
4.1 Fonte de dados externa para obtenção de PM2.5

Para tal é necessário investigar e perceber o comportamento da concentração destas partículas, a diferentes níveis de detalhe e com enfoque nas região administrativas (e.g. freguesias), através de dados que são obtidos por uma fonte externa. Esta fonte recolhe dados sobre a qualidade do ar, disponibilizando vários *endpoints* de acesso, em particular um para obter o PM2.5, esse caracteriza-se por receber a latitude e longitude do ponto geográfico pretendido resultando no fornecimento do valor do PM2.5 para o dia em questão.

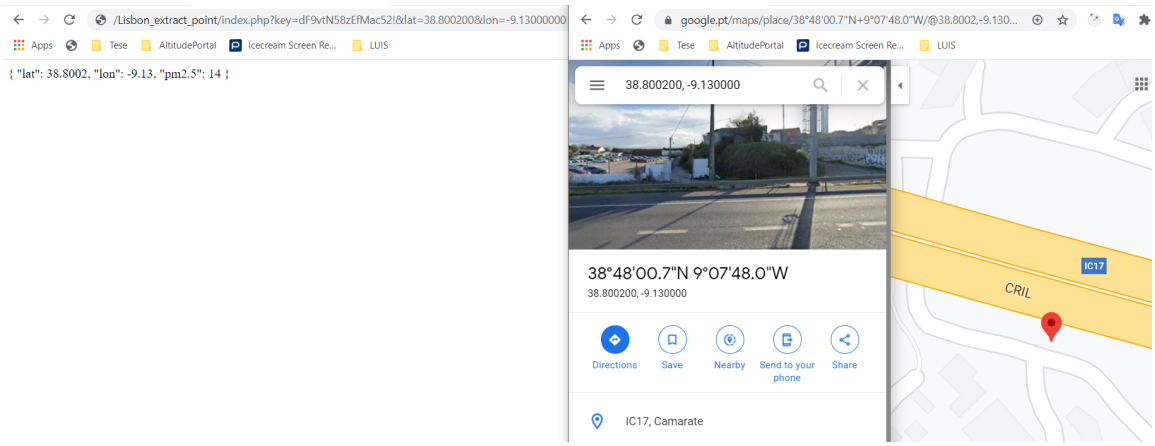
O valor obtido da fonte é agregado usando uma média deslizante a 7 dias, enfatizando a tendência de variação do PM2.5. Assim, a granularidade mínima dos dados disponíveis é o dia, não sendo possível desagregar esses valores, e.g., para cada hora. Cada leitura, antes da agregação, é efectuada numa localização no mapa, com uma variação



(a) 1º Ponto Referência



(b) 2º Ponto deslocado horizontalmente



(c) 3º Ponto deslocado verticalmente

Figura 4.1: Teste à fonte de dados para 3 pontos geográficos distintos.

normal de coordenadas, fruto da flutuação de posição indicada pelo GPS, mas também por recolhas em locais próximos mas distintos (e.g. em diferentes lados de uma rua). Note-se que é necessário fornecer uma localização para obter o valor de poluição. A Figura 4.1(a) ilustra um exemplo de um pedido à fonte, onde se pode observar o resultado obtido por esta do lado direito da figura, e no seu lado esquerdo a localização desse ponto pretendido. A Figura 4.1(b) ilustra o mesmo, mas agora para um ponto aproximadamente 10m afastado do anterior. As Figuras 4.1(a)–4.1(c) apresentam dados recolhidos para o mesmo dia, e como se pode observar pelos dois primeiros apesar de geograficamente estes estarem aproximadamente 10m distantes ambos apresentam o mesmo valor de medição de PM2.5. Já no terceiro caso, igualmente distante destes dois em cerca de 100m, observa-se um valor superior, apesar de este valor ser apenas o aumento em 1 o valor de concentração de PM2.5 dos restantes. Tendo este comportamento ao longo da cidade possa ter justificado a fonte de dados aplicar a granularidade já referida anteriormente para os valores de PM2.5. Nos casos em que para pontos geográficos que para o dia pretendido não foram registadas qualquer medição para o PM2.5 a fonte de dados retorna esta mesma representação apenas com a diferença que para o campo `PM2.5` retorna -1, indicando ausência de valor.

Dada a intrínseca variabilidade associada às coordenados, e para que seja possível manter dados históricos que reportem a uma mesma localização, foi definida uma grelha de 100×100 m^2 sobre a área de interesse (a cidade de Lisboa). A Figura 4.2 ilustra a grelha no mapa, dividindo-o aproximadamente em 14.625 células. Desta forma, para cada medição que é efectuada numa localização que esteja dentro dos limites de uma célula considera-se que reporta a essa célula. Assim, considera-se a coordenada central da célula sempre que se obtém dados da fonte externa. O número de células após a intercepção entre a *Bounding Box* da região de interesse (14.625 células) e os polígonos que definem os limites da cidade de Lisboa é de aproximadamente 8.640.

Os valores recolhidos dentro dos limites da cidade que apresentam um valor de PM2.5 variam consoante o dia, sendo o seu pico máximo, até ao presente, de aproximadamente 3.171 pontos. Dada a situação global da pandemia Covid-19, e principalmente durante o confinamento, observou-se um decréscimo brusco destes pontos na ordem dos 87%.

4.2 Serviço de recolha dos dados da poluição

Os dados disponibilizados pela fonte externa (Figura 3.1-A, e descrito em 4.1) são recolhidos, processados e armazenados. Este processo é feito diariamente para obter

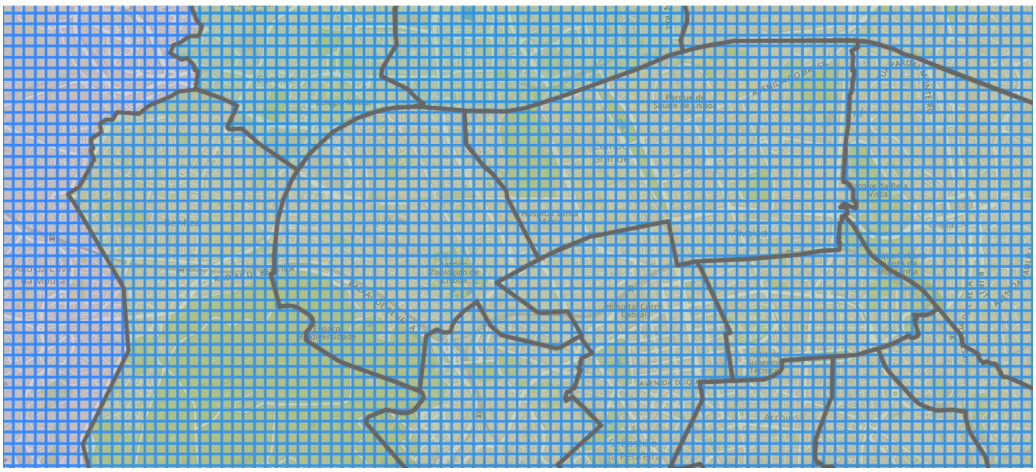


Figura 4.2: Ilustração do reticulado que define as localizações onde os dados de qualidade do ar são reportados, pelo centro de cada célula

o valor PM2.5 para cada ponto da grelha, processá-lo e guardar o resultado de forma persistente numa base de dados, como ilustrado na Figura 3.1-Ⓑ onde aqui se observa o fluxo representado pelas setas azuis na figura. O fluxo Ⓐ representa a obtenção dos pontos diários de PM2.5. O fluxo bidireccional entre Ⓑ e Ⓔ representa a comunicação entre o serviço e a base de dados de forma a saber os pontos a recolher, tal como depois de o valor do poluente diário destes ser recolhido e processado ser guardado na base de dados. Esta que irá conter todo este histórico até ao momento.

A Tabela 4.1 ilustra uma amostra dos dados na base de dados depois destes serem processados. Para além do identificador (`id`) de cada tuplo, contém o valor do poluente PM2.5 obtido, a data em que este foi obtido e uma referência (`f_id`) para a tabela que contém a grelha e o ponto respectivo desta. A Tabela 4.2 apresenta uma amostra onde estes são referenciados e onde podemos então obter cada grelha e seu respectivo ponto central.

id	**pm2_5**	**date_info**	**f_id**
15210	6	20/11/2019 00:00	2638
15211	14	20/11/2019 00:00	2639
15212	4	20/11/2019 00:00	2663
15213	25	20/11/2019 00:00	2706
15214	25	20/11/2019 00:00	2707

Tabela 4.1: Amostra dos dados da tabela `map_info` na base de dados

Map Service foi o nome escolhido para a componente que implementa esta funcionalidade, pois, do ponto de vista tecnológico, é um serviço implementado em Java

id	f_id	g_grid	g_grid_point
1	1	010300...	0101000020E610000032EAD9B0617522C0509D71888E584340
2	2	010300...	0101000020E6100000C4EC2C09CB7422C0519D71888E584340
3	3	010300...	0101000020E610000057EF7F61347422C0519D71888E584340
4	4	010300...	0101000020E6100000EBF1D2B99D7322C0519D71888E584340
5	5	010300...	0101000020E61000007FF42512077322C0519D71888E584340

Tabela 4.2: Amostra dos dados da tabela `grid_info` na base de dados

usando a *framework* Spring[1], que fornece as ferramentas necessárias para que todos os fluxos descritos possam ser cumpridos da melhor forma. Os Algoritmos 1 e 2 descrevem duas funções necessárias a este serviço, respectivamente `InsertMapInfo` e `GetAllPoints`. A primeira (Algoritmo 1) tem a função de inserir na base de dados o valor do poluente PM2.5 no momento em que a acção é executada. Para isso esta recebe como argumento um objecto de nome `PointInfo`, que contém um ponto geográfico (latitude, longitude) usando a norma WGS84 e um identificador que refere a grelha de onde este foi retirado. Este ponto é passado à fonte externa que, como explicado em 4.1, irá permitir que esta forneça o valor do poluente para as respectivas coordenadas. A função `callInsertIntoMap_Info` recebe o valor de PM2.5 obtido, a data em que este foi obtido, e a referência para a grelha do ponto utilizado. Com isto esta insere estes valores na base de dados resultando num tuplo na tabela, como podemos observar na Tabela 4.1. A segunda (Algoritmo 2) recebe cada um dos pontos da grelha, que são usados como referência aos pontos passados como longitude e latitude à fonte de dados utilizado no Algoritmo 1. A função `callGetGrid_Info` recolhe todos estes pontos no formato idêntico ao da Tabela 4.2 apenas aproveitando os atributos `f_id` e `g_grid_point` criando uma lista do objecto `PointInfo`. O ponto de partida diário para o funcionamento deste serviço é composto pela combinação destes dois algoritmos, em que para cada ponto da grelha é obtido o seu valor diário para o poluente em análise.

Algoritmo 1 Inserir valores de PM2.5 para cada ponto

```
1: procedure INSERTMAPINFO(PointInfo point)
2:     pm2_5  = callGetPointFromExternalSource(point)
3:     if pm2_5  != -1 then
4:         callInsertIntoMap_Info(pm2_5, date_info, f_id)
5:     end if
6: end procedure
```

[1] https://spring.io/

Algoritmo 2 Obter todos os pontos da grelha

```
1: procedure GETALLPOINTS
2:     points = [ ]
3:     pointsFromGrid = callGetGrid_Info()
4:     while each points p in pointsFromGrid do
5:         points.add(p)
6:     end while
7:     Return points
8: end procedure
```

4.3 Fonte de dados de trânsito

Sendo-nos possível monitorizar diariamente o valor do poluente PM2.5, sentiu-se a necessidade de justificar e clarear o sentido dos valores que eram obtidos. Dado que o trânsito na cidade de Lisboa [15] é um dos principais causadores do aumento do nível de poluição do ar, usou-se informação de tráfego. Esta informação foi obtida da *Web API*[2] com vários *endpoints* disponiveis de modo a obter a informação desejada. Esta apresenta dados disponibilizados do WAZE[3], na forma de GeoJSON[4], um formato para codificar uma variedade de estruturas de dados geográficos.

A Listagem 4.1 contém um exemplo de dados devolvidos pela WebAPI. Como se pode verificar, a resposta contém uma colecção de dados geográficos, indicando o tipo de geometria e as coordenadas (latitude, longitude) desta. Cada geometria contém propriedades associadas de modo a dar alguma informação ao contexto inserido. Neste caso, por exemplo, o nome da rua, o nó da mesma a que se referem os dados, e informação numérica relativa ao transito como o comprimento deste a velocidade, o atraso em relação a uma situação de tráfego regular, e o nível do mesmo. Da Listagem 4.1 foram usados atributos para a aplicação cliente e para serem usados na representação do mapa. Estes foram as coordenadas representadas na linha 6, e o nome da rua na linha 19 e o `level` (linha 10). Os dados usados para modelação, para além dos mencionados, foram o `length` (linha 11), `speed` (linha 16), `road_type` (linha 17) e `delay` (linha 18).

[2]https://emel.city-platform.com/opendata/
[3]https://www.waze.com/pt-PT/
[4]https://geojson.org/

```
{ "type": "FeatureCollection",
  "totalFeatures": 141,
  "features": [
    { "type": "Feature",
      "geometry": { "type": "MultiLineString",
                    "coordinates":
                       [[[-9.228765,38.70222],[-9.228814,38.701507]]
                    ] },
      "properties": { "country": "PO",
                      "city": "Alges",
                      "level": 5,
                      "length": 79,
                      "turn_type": "NONE",
                      "type": "NONE",
                      "uuid": "1436924985",
                      "end_node": null,
                      "speed": 0,
                      "road_type": 1,
                      "delay": -1,
                      "street": "R. Luis de Camoes",
                      "pub_millis": 1599793749757,
                      "bbox": [-9.228814,38.701507,-9.228765,38.70222]
                    } } }
```

Listagem 4.1: Exemplo dos dados obtidos da fonte externa em GeoJSON

4.4 Serviço automático de recolha de dados de trânsito

Numa fase inicial do trabalho, foram realizados alguns pedidos de teste, para tentar perceber se existia regularidade na actualização da informação e, em caso afirmativo, qual a frequência. Verificou-se que, em média, de 5 em 5 minutos existiam dados novos. Visto que o trânsito, principalmente na cidade de Lisboa, tem variações próprias dos movimentos pendulares, conhecidas como "horas de ponta", é necessário recolher os dados várias vezes ao dia, mantendo a frequência mínima para mesmo assim estar sensível a alterações no tráfego rodoviário. Essa é a granularidade com que os dados são armazenados. No entanto, para efeitos de análise conjunta entre dados de trânsito e poluição, foi necessário efectuar, após a recolha, uma transformação e armazenamento dos dados, uma agregação à mesma granularidade dos dados do poluente PM2.5, ou seja, realizando-se a média do dia. Este serviço a ser implementado em comparação ao serviço implementado para a recolha diária do poluente, o outro requer mais processamento imediato, visto que este realiza desde a formatação/transformação dos dados, leitura e escrita à base de dados e pedidos à fonte externa para obtenção do poluente, e este apenas acesso à fonte e respectiva transposição dos resultados para uma base de dados, por uma questão de desempenho e capacidade de resposta a pedidos, dado que estes iriam acontecer no mínimo de 5 em 5 minutos ao dia, foi necessário analisar uma

nova via tecnológica que fosse mais ajustada a este requisito.

O Node Red [18] foi a ferramenta escolhida que melhor respondeu para dar suporte à recolha e armazenamento dos dados do trânsito, dada a sua facilidade de implementação e capacidade de resposta às necessidades de recolha calendarizada de dados. O Node Red suporta vários sistemas de base de dados, tendo sido usado o SQLite [25]. Assim separa-se a recolha do processamento de dados, utilizando uma base de dados intermédia. Esta ferramenta permite desenvolver fluxo de processamento de dados *flows* usando o paradigma *Flow-based programming* [14]. A Figura 4.3 ilustra o *flow* criado para a obtenção dos dados pretendidos. Este começa por um bloco de *timestamp* (Figura 3.1-Ⓐ) que permite definir a calendarização de início do *flow*. Foi usado um intervalo de 5 minutos. De seguida usa um bloco de função (Figura 3.1-Ⓑ), que adiciona um *header* com a chave adquirida da WebAPI para se poder fazer o pedido aos dados do Waze. Como resultado desse bloco temos um que define o pedido http (Figura 3.1-Ⓒ) aos dados do tráfico provenientes do Waze. Para se poder guardar esse resultado tem de se criar um bloco (Figura 3.1-Ⓓ) que cria a tabela na base de dados esse bloco com o nome de "Create database table" define uma tabela `Waze` com um identificador, uma data que contém a data corrente no momento do pedido, e a resposta em GeoJSON do pedido. Com este bloco definido já podemos ter outro (Figura 3.1-Ⓔ) que usa o resultado para inserir na base de dados. Posto isto, conseguimos definir um ciclo com um intervalo de 5 minutos, permitindo ter dados diários neste período.

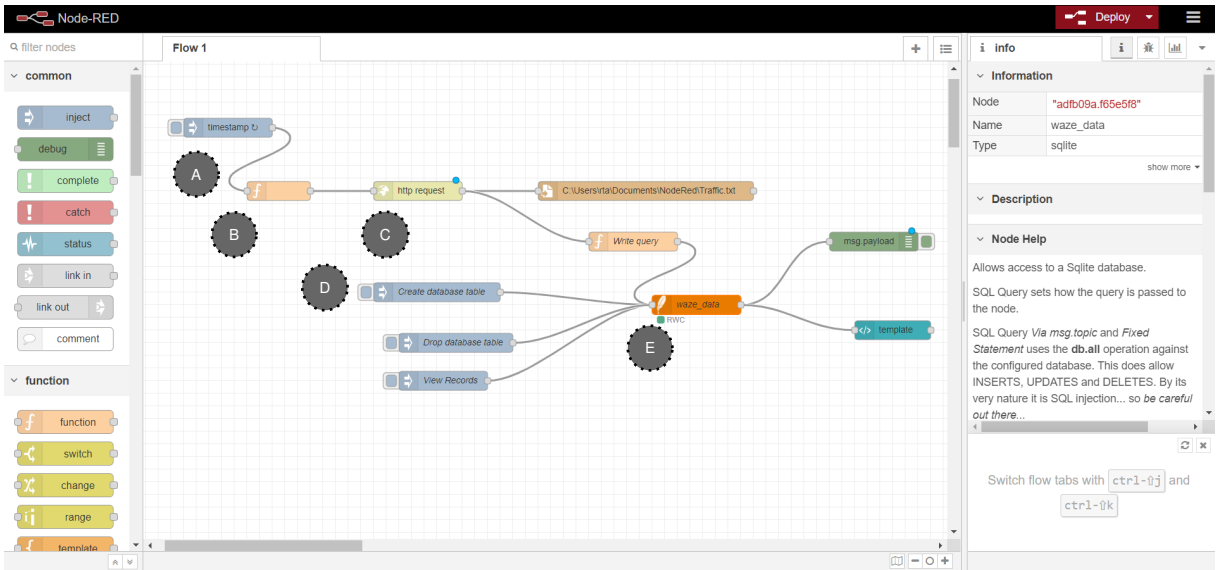


Figura 4.3: Flow usado para o serviço em Node Red

Com esta abordagem, e dado todo o processo construído até ao momento, visto que

os dados recolhidos para o trânsito contêm dados geográficos para além de dados que apenas descrevem o trânsito (e.g nome da rua), estes necessitam de ser transpostos para a base de dados que suporta este tipo de dados através do PostGIS, para tal era necessário fazer o tratamento prévio dos dados.

Os dados resultantes do serviço são exportados para um ficheiro de extensão .db, posteriormente este é convertido para um ficheiro .csv, com o objectivo de melhorar a representação e manipulação dos dados contidos no ficheiro. A última coluna vem com o objecto GeoJSON, como o da Listagem 4.1, por cada linha do ficheiro, que como explicado anteriormente, é necessário decompor as propriedades deste neste caso por coluna dentro deste ficheiro.

Para tal ser solucionado foi realizado um pequeno programa em Python que tem como *input* e *output* respectivamente um ficheiro com os dados e outro que é produzido ficando com os dados formatados. Este é produzindo tirando partido de auxílios de funções que permitam processar cada uma das linhas da coluna `Data`, coluna que contém o objecto GeoJSON numa única coluna, ou seja o conteúdo em formato GeoJSON é decomposto em colunas, deixando de ter a coluna `Data` e passando a ter cada uma das propriedades (Listagem 4.2).

```
1  df = pandas.read_csv('./_2020-03-19/WAZE.csv', converters={'DATA':TrafficParser},header=0)
2  with open('./_2020-03-19/traffic_2020-03-19.csv', mode='wb') as csv_file:
3      # create the csv writer object
4      csvwriter = csv.writer(csv_file, delimiter=',')
5      idxId = 0
6      count = 0
7      listJson = list(df['DATA'])
8      for geojson_data in listJson:
9          if geojson_data['type'] == 'FeatureCollection':
10             parse_feature_collection(df, idxId, geojson_data['features'], csvwriter, count)
11         else:
12             print("Can␣currently␣only␣parse␣FeatureCollections,␣but␣I␣found␣", geojson_data['
                   type'], "␣instead")
13         idxId += 1
14         count += 1
15     csv_file.close()
```

Listagem 4.2: Decomposição das propriedades GeoJSON em colunas csv

Este processo começa por ler um ficheiro proveniente do caminho relativo desta aplicação, a função utilizada para tal pertence à biblioteca pandas[5], esta que fornece um conjunto de funcionalidades relativas a análise e manipulação de dados. A função de leitura contém algumas parametrizações no entanto as utilizadas para o efeito foram o caminho do ficheiro, *converters* um parâmetro que consiste num dicionário de funções para converter valores em certas colunas, neste caso a função `TrafficParser`

[5]https://pandas.pydata.org/

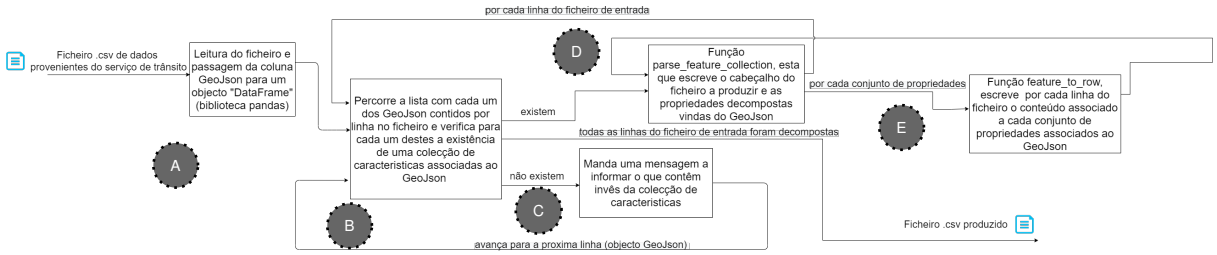


Figura 4.4: Processo de decomposição das propriedades GeoJSON provenientes do serviço de trânsito

pega na coluna "Data" esta que contem os GeoJSON e decompõe-o em objectos python tirando partido da função `json.loads` da biblioteca `json`. Por fim, é usado o parâmetro `headers` este com o valor a 0 que indica que a primeira linha do ficheiro contem os *headers*.

Este processo representado pela Figura 4.4-A obtém da leitura um `DataFrame`, uma estrutura de dados bidimensional com *labels* nos eixos. É através deste que se manipula o conteúdo do ficheiro e se manipulam os dados para atingir o objectivo deste programa.

De seguida através da estrutura de dados obtida acede-se à coluna "Data" transformando o conteúdo desta numa lista, daí resulta o processo representado pela Figura 4.4-B, onde ao iterar sobre a lista verifica-se se cada um dos objectos representativos de GeoJSON na propriedade *type* indica o valor *FeatureCollection*, que representa uma colecção com o conjunto das propriedades formadas por cada um dos objectos GeoJSON. Se estas não indicarem esse valor, é informado que tipo estes de facto representam e o mesmo não é processado não chegando portanto a ser escrito no ficheiro de saída, processo indicado na Figura 4.4-C. No caso em que se verifica é chamada a função `parse_feature_collection` esta que recebe como parâmetros o "df" correspondente ao `DataFrame`, "idxId" que corresponde ao índice corrente do ficheiro, "geojson_data["features"]" corresponde à colecção de propriedades relativas ao objecto iterado, "csvwriter" o ficheiro de *output* onde se vai escrever e por fim "count" que representa a contagem de linhas do novo ficheiro. Este processo que ocorre na função descrita pela Figura 4.4-D começa por escrever os *headers* no ficheiro de saída caso o "count" seja 0, adicionando a este as propriedades contidas no objecto representado pelo GeoJSON. Para além disso esta itera sobre as *features* provenientes do objecto GeoJSON resultando no processo representado pela Figura 4.4-E.

Este processo sendo o último do fluxo deste programa e dado que até ao momento

temos escrito no ficheiro os *headers*, presume-se que este irá escrever o conteúdo associado aos *headers*, ou seja, ao chamar-se a função de seu nome `feature_to_row` esta para cada uma das *features* que estão a ser iteradas pela função previa escreve no ficheiro de saída o conteúdo que é acessível pelo objecto através de `feature['properties']` para o caso das propriedades no caso das coordenadas geográficas desta *feature* acede-se por `feature['geometry']['coordinates']`.

Quando a iteração em Figura 4.4-Ⓑ terminar significa que temos as propriedades dos objectos GeoJSON decompostas no ficheiro de saída, obtendo o resultado pretendido com este programa.

5

Modelo preditivo para PM2.5

Dado a informação recolhida acerca da observação dos valores do poluente de PM2.5 em torno da cidade de Lisboa, e dado o problema a resolver para os utilizadores finais, é considerado como uma mais valia que, seja possível através de factores que possam influenciar a variância dos valores de PM2.5, neste caso os dados do trânsito, que se possa construir modelos capazes de prever estes valores, expandindo a margem temporal dentro da aplicação.

5.1 Geração do modelo preditivo

Para gerar um modelo de previsão da poluição, usando alguma informação de contexto, e.g., dados de trânsito, foi usada a ferramenta *Open Source* H_2O [9]. Esta ferramenta suporta uma interface WEB, H_2O Flow[1], onde se faz o desenvolvimento do modelo, podendo-se usar um leque alargado de tipos de aprendizagem e algoritmos, como por exemplo como *deep learning* e *gradient boosted machines*. Além disso, suporta um *dashboard* que permite analisar os resultados por exemplo em forma de gráficos ou tabelas. Além deste tipo de utilização, existem outras formas de usar esta ferramenta, quer seja através de uma página web com uma interface específica, quer seja por Hadoop ou Spark, ou até mesmo usando Python ou R. Em relação aos algoritmos suportados temos os supervisionados e não supervisionados, e dois que consideram ser misto.

[1] http://docs.h2o.ai/h2o/latest-stable/h2o-docs/flow.html

Os algoritmos supervisionados utilizados foram o Random Forest[2], Gradient Boost Machine[3] e o Deep Learning (Neural Networks)[4]. Como o H_2O fornece a possibilidade de produzir modelos combinando alguns destes algoritmos mencionados, surgiu a utilização de um outro algoritmo supervisionado o *Automatic Machine Learning* (AutoML) [10], onde utilizador introduz as variáveis independentes e dependente, sendo produzido um conjunto de modelos ordenado do melhor para o pior, segundo métricas como: 1) R^2 (R Squared); 2) *Mean Squared Error* (MSE); 3) *Root Mean Squared Error* (RMSE); 4) *Root Mean Squared Logarithmic Error* (RMSLE) e 5) *Mean Absolute Error* (MAE) , no caso do objectivo ser uma regressão, e no caso de ser de um problema classificação, são usadas métricas como: 1) Gini Coefficient; 2) Absolute *Matthews Correlation Coefficient* (MCC); 3) F1; 4) F0.5; 5) F2; 6) Accuracy; 7) Logloss; 8) *Area Under the ROC Curve* (AUC); 9) *Area Under the Precision-Recall Curve* (AUCPR) e 10) Kolmogorov-Smirnov (KS) Metric. Este algoritmo é caracterizado por usar os algoritmos supervisionados ou o mesmo modelo usar vários algoritmos no caso do *Stacked Ensembles*.

O algoritmo *Stacked Ensembles* [30] consiste no uso conjunto de outros algoritmos supervisionados, de maneira a que desempenho preditivo seja melhor que usar apenas um desses algoritmos. Por exemplo, o uso do *Random Forest* (RF) e *Gradient Boost Machine* (GBM) são dois exemplos de algoritmos que podem ser usados neste contexto. *Stacking*, também chamado de *Super Learning* [26] ou *Stacked Regression* [3], é uma classe de algoritmos que envolve o treino de um "meta learner" de segundo nível para encontrar a combinação ideal dos modelos básicos (*weaker learners*) . Ao contrário do *bagging* [2] e do *boosting*, o objectivo do empilhamento é obter grupos fortes e diversos de modelos.

O algoritmo XGBoost [4] pertencente à categoria dos algoritmos supervisionados e implementa um processo chamado *boosting* [8] para produzir modelos precisos. *Boosting* refere-se à técnica de construir modelos sequencialmente, com cada novo modelo tentando corrigir as deficiências do modelo anterior. No *boosting* de árvore, cada novo modelo adicionado ao conjunto é uma árvore de decisão. O XGBoost fornece *boosting* de árvores paralela (também conhecido como *Gradient Boosting Decision Tree* (GBDT) e *Gradient Boost Machine* (GBM)) que resolve muitos problemas de ciência de dados de maneira rápida e precisa. Para muitos problemas, o XGBoost é uma das melhores *frameworks* de *gradient boosting machine* (GBM) dos dias de hoje.

O H_2O foi usado tirando partido dos dados relativos ao trânsito visto que estes já estavam a ser recolhidos e representados no mapa. Posto isto, com os dados que foram

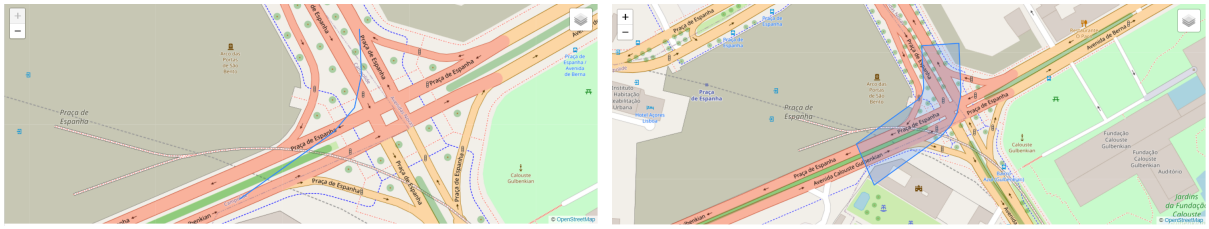
[2]http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/drf.html
[3]http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/gbm.html
[4]http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/deep-learning.html

produzidos pelo serviço descrito na Secção 4.4 neste caso um ficheiro com os dados do trânsito num formato *csv*, era necessário cruzar estes com os registos acerca do valor do poluente. Para isso foi necessário colocar os dados do trânsito na base de dados com suporte geográfico para que seja possível utilizar as funções necessárias para permitir o cruzamento dos dados. Com a função `copy` foi possível passar a informação do ficheiro *csv* formatado para uma tabela de nome `traffic`, que contém os mesmos atributos que o ficheiro. Um destes atributos são as coordenadas geográficas representadas por (latitude, longitude) relativas aos troços com trânsito, mas sendo que é necessário aplicar funções para lidar com tipos geográficos, este atributo necessitava de ser convertido de coordenadas para o tipo geográfico suportado pelo PostGIS (*geometry*). Já a pensar no cruzamento com os dados registados do poluente, visto que a forma de sabermos o valor do poluente para uma data específica, numa linha (e.g rua, cruzamento, autoestrada) de trânsito é dado pela intersecção dessa linha com a grelha do mapa, decidiu-se alargar essa linha para que possa abranger uma área maior desse troço de modo a que este possa interceptar mais pontos com registos do valor do poluente. Sendo assim, para além da conversão das coordenadas para o tipo geográfico suportado, foi ainda acrescentada um outro dado geográfico que representa esse mesmo troço alargado em cerca de 20m. Como se pode verificar na Listagem 5.1, foram adicionadas duas colunas geográficas: uma que resulta da conversão `geom_ml` e outra `geom_m_buffer` que resulta da aplicação da anterior passando pela função `ST_Buffer`, responsável por alargar os troços em 20m.

```
1 SELECT AddGeometryColumn ('public','traffic','geom_ml',0,'MULTILINESTRING',2);
2 SELECT AddGeometryColumn ('public','traffic','geom_ml_buffer',4326,'POLYGON',2);
3 UPDATE public.traffic SET geom_ml = ST_SetSRID(ST_GeomFromText(CONCAT(CONCAT('MULTILINESTRING
      ((',geo), '))')), 4326);
4
5 UPDATE public.traffic SET geom_ml_buffer = ST_Buffer(geom_ml, 0.0002, 'endcap=square join=
      bevel');
```

Listagem 5.1: Tratamento dos dados geográficos do trânsito

A Figura 5.1(a) ilustra um exemplo do troço do trânsito com dimensão predefinida, enquanto que a Figura 5.1(b) representa este após passar por o processo em que é ampliado. Como se pode verificar, ambos representam o mesmo troço sendo que o 2º apresenta uma maior área espacial retirando em certos casos melhor aproveitamento dos polígonos da grelha que registaram o valor do poluente. Se pensarmos, por exemplo, na Avenidade da Liberdade no centro da cidade de Lisboa, esta apresenta uma via central de pelo menos 2 faixas em cada sentido mas paralelamente a ambos os sentidos existem pelo menos mais 1 faixa com alguma margem de distancia pelo meio. Ou seja em termos práticos obtendo uma recta para este troço principal e cruzando apenas com os pontos obtidos da poluição pode ocorrer perda de informação relativo a estes troços



(a) Troço de trânsito (b) Troço de trânsito alongado

Figura 5.1: Comparação de um mesmo troço predefinido e alargado relativos ao trânsito

paralelos dai este aumento de área geográfica vai permitir o cruzamento também com os pontos eventualmente capturados nessa área.

Após a inserção e processamento do dados de trânsito, passou-se então ao cruzamento destes com os dados da poluição, constituindo assim aquele que será o *dataset* a utilizar no H_2O. Primeiramente adicionou-se dois novos atributos `pm2_5_average_geo`, `pm2_5_values_geo`, que representam respectivamente a média dos valores do poluente para cada troço de trânsito e o outro que irá apresentar todos esses valores que foram usados para o cálculo dessa média, não só ajudando a validar esse cálculo como disponibilizar esses valores para, se necessário, serem usados para outro tipo de operações que não a média. Com isto é necessário introduzir os valores correspondentes a estes atributos este processo é feito com o cruzamento de outras tabelas já existentes. Assim, começa-se por cruzar (Figura 5.2-A) os dados de `map_info` com `grid_info` de modo a ter os valores dos poluentes segundo os polígonos da grelha respectivos, estes polígonos que são intersectados (Figura 5.2-B) com os troços de trânsito ampliados

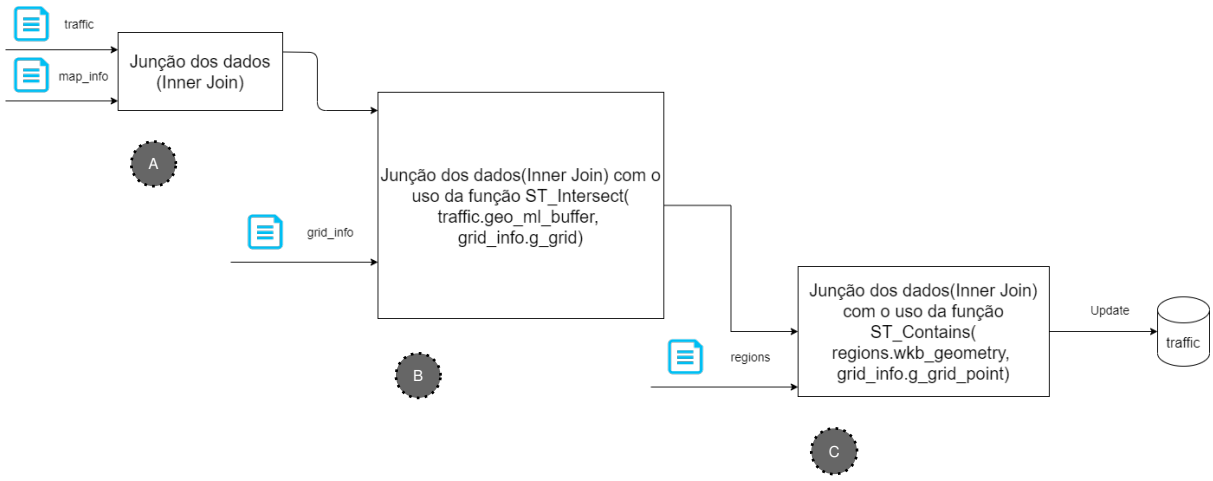


Figura 5.2: Processo de junção dos dados do trânsito com a poluição

`geom_ml_buffer` tirando partido da função `ST_Intersects` que se uma geometria ou geografia compartilha qualquer porção do espaço, então elas cruzam-se, isto para que se contabilize os valores do poluente que se cruzam com o troço do trânsito. Por fim de maneira a ser agrupado por regiões, (Figura 5.2-C) é feito o cruzamento destas com os polígonos da grelha, pela função `ST_Contains`. Com este processo a tabela `traffic` contém a informação necessária para se poder construir o *dataset*, sendo assim esta é copiada para um ficheiro *.csv* que servirá de *input* válido a usar no H_2O. A Listagem 5.2 representa este processo de cruzamento dos dados e extracção destes para um *dataset*.

```
1  ALTER TABLE public.traffic ADD COLUMN pm2_5_average_geo INTEGER;
2  ALTER TABLE public.traffic ADD COLUMN pm2_5_values_geo INTEGER[];
3  UPDATE traffic
4  SET pm2_5_average_geo = sub_q.avg_pm2_5, pm2_5_values_geo = sub_q.values_pm2_5
5  FROM
6      (
7      SELECT AVG(M.pm2_5) as avg_pm2_5, array_agg(M.pm2_5) as values_pm2_5, T."DT", T.uuid
8      FROM traffic AS T
9        INNER JOIN map_info AS M
10       ON (date(T."DT") = date(M.date_info))
11       INNER JOIN grid_info AS G
12       ON (M.f_id = G.f_id AND ST_Intersects(T.geom_ml_buffer, G.g_grid))
13       INNER JOIN regions AS R
14       ON (ST_Contains(R.wkb_geometry, G.g_grid_point))
15     GROUP BY ROLLUP(T."DT", T.uuid)
16     ) AS sub_q
17 WHERE sub_q."DT" = traffic."DT" AND sub_q.uuid = traffic.uuid;
18 \copy (Select * From traffic where pm2_5_average_geo is not null) TO 'C:/Users/rta/DOCUME~1/
       Tese/H20_examples/traffic_2020-03-19WMV.csv' DELIMITER ',' CSV HEADER;
```

Listagem 5.2: Cruzamento dos dados do trânsito com a poluição

Os meios utilizados para o H_2O foram o módulo H_2O em Python e o H_2O Flow, sendo que esta última foi a mais utilizada e que serviu para a construção do modelo final. As primeiras experiências passaram por testar através da utilização em Python o resultado de alguns algoritmos supervisionados sendo eles o RF, GBM e *Deep Learning* (*Neural Networks*), variando os parâmetros de entrada destes e verificando no fim métricas para algoritmos de regressão ou classificação. Os modelos obtidos foram importados para a aplicação web do H_2O de modo a analisar os resultados com amostragem de tabelas e ou gráficos, visto que o H_2O fornece a possibilidade de produzir modelos combinando alguns destes algoritmos mencionados, surgiu a utilização do AutoML. Posto isto o processo decorrido tirando partido da aplicação web do H_2O vai desde a introdução do *dataset* produzido até à análise dos modelos resultantes pela utilização do AutoML e extracção do melhor modelo para se poder usar na predição do classificador do modelo. Iniciando a aplicação web H_2O, de seu nome `H2O FLOW`, isto porque a utilização desta é definir um conjunto de operações definindo um fluxo de processamento (*flow*),

desde a importação do ficheiro, à geração dos modelos, pré e pós processamento e exportação dos modelos para predição.

A primeira operação do *flow* passa por indicar e importar o ficheiro neste caso no formato especifico *csv* suportado pela aplicação e construído a partir da base de dados do projecto. Após importarmos o ficheiro podemos observar os dados importados, onde se verificam numa tabela em que a primeira coluna é o nome das variáveis e as restantes os vários valores destas. Confirmando os dados é preciso separar uma percentagem do *dataset* em treino e teste neste caso as percentagens utilizadas foi aproximadamente 70%/30% respectivamente.

De seguida a próxima operação a ser feita é a escolha do algoritmo e seus parâmetros mínimos, sendo estes a escolha da variável dependente e a escolha do *dataset* de treino, dado que para que este possa produzir resultados pode se assumir como variáveis independentes todas as que são dadas como colunas do ficheiro inicialmente importado à excepção da que foi escolhida como variável dependente, caso o utilizador queira pode excluir algumas destas considerando as restantes como variáveis independentes.

O algoritmo escolhido foi o algoritmo supervisionado AutoML, começou-se por ter de identificar vários parâmetros como o *dataset* de treino e teste a usar, a coluna respectiva à variável dependente do modelo, coluna que corresponde ao valor a ser previsto neste caso `pm2_5_average_geo`, as variáveis independentes seleccionando das colunas importadas quais as ignoradas utilizando as restantes, sendo que neste caso as usadas foram `speed`, `road_type`, `level`, `length`, `delay`. Caso o utilizador ache necessário pode seleccionar, ou não, se pretende excluir algum algoritmo suportados pelo H_2O para serem usados na obtenção do melhor modelo.

Posto isto, com a execução desta operação, o H_2O vai correr uma tarefa que irá produzir uns quantos modelos, dependendo do valor definido pelo atributo `max_models`. Este é especificado antes da execução deste algoritmo, para que se verifiquem os modelos assim que a tarefa de executar o algoritmo esteja concluída esta permite executar uma nova operação de seu nome `getLeaderBoard` que apresenta sobre a forma de tabela os melhores modelos produzidos segundo uma ordem decrescente do desvio médio. Na Tabela 5.1 observa-se assim os modelos obtidos assim como as métricas de avaliação de modelos de regressão como `Mean-residual-deviance`, `RMSE`, `MSE`, `MAE`, `RMSLE`.

Como se observa na Tabela 5.1 o modelo `StackedEnsemble_AllModels` apresenta melhor resultado para o desvio médio, no entanto irá ser analisado em melhor detalhe estas métricas em termos de *dataset* de treino teste e *cross-validation*, decidindo assim que modelo apresenta os melhores resultados para ser usado.

model_id	mean_residual_deviance	rmse	mse	mae	rmsle
StackedEnsemble_AllModels	8.3544	2.8904	8.3544	1.9208	0.1551
StackedEnsemble_BestOfFamily	8.3602	2.8914	8.3602	1.9189	0.1552
XGBoost_grid__1	8.3800	2.8948	8.3800	1.9178	0.1556
XGBoost_2	8.6609	2.9429	8.6609	1.9674	0.1580
XGBoost_1	8.8133	2.9687	8.8133	2.0137	0.1598
XGBoost_3	10.2916	3.2081	10.2916	2.2805	0.1728
GLM_1	24.8024	4.9802	24.8024	3.9569	0.2683

Tabela 5.1: Tabela com os modelos produzidos ordenados crescentemente consoante o desvio médio relativo à variável dependente.

Escolhendo cada um dos modelos obtidos conseguimos confirmar não só alguns parâmetros que foram previamente seleccionados como o `dataset` de treino e de teste, a variável dependente, o número de partições a usar no `K-fold-cross-validation` por omissão K=5, método de validação do modelo que consiste em dividir o conjunto total de dados em k subconjuntos mutuamente exclusivos do mesmo tamanho e, a partir daí, um subconjunto é utilizado para teste e os k-1 restantes são utilizados para a estimativa dos parâmetros, fazendo-se o cálculo do erro estimado do modelo. Para além deste parâmetros existe um outro de seu nome `base_models` que consiste para o caso em que o modelo usou o algoritmo *Stacked Ensembles*. Observa-se assim os tais parâmetros referidos e confirma-se que para o primeiro modelo seleccionado os algoritmos usados no *Stacked Ensembles* são o *XGBoost* 4 modelos e por fim um modelo de *Generalized Linear Model* (GLM).

Como *output* cada modelo permite observar as métricas de avaliação para o *dataset* de treino, teste e *cross-validation*, Tabelas 5.1(a), 5.1(b) e 5.1(c) respectivamente, neste caso as métricas apresentadas correspondem às de regressão dado o tipo da variável de resposta.

Como se pode observar nestes resultados, os que foram aplicados ao *dataset* de teste foram mais favoráveis, por exemplo o coeficiente de determinação (R^2) ronda os 72% sendo que num modo geral o modelo que apresenta o melhor valor deste é o StackedEnsemble_BestOfFamily. Tendo se observado que este mesmo modelo não só apresenta os melhores resultados obtidos em relação a esta métrica mas também relativos às restantes em comparação com os outros modelos e em cada um dos *datasets* analisados, este foi o modelo escolhido na previsão do nosso caso de estudo, prever o valor para o poluente PM2.5.

(a) Tabela de resultados relativos ao *dataset* de treino

Training Metrics	**MSE**	**RMSE**	**R^2**	**MAE**	**RMSLE**
StackedEnsemble_AllModels	5.6908	2.3855	0.8020	1.49	0.1264
StackedEnsemble_BestOfFamily	5.6509	2.3772	0.8034	1.4717	0.1258
XGBoost_grid__1	5.6766	2.3826	-	-	-
XGBoost_2	6.2626	2.5025	0.7821	1.6235	0.1333

(b) Tabela de resultados relativos ao *dataset* de teste

Validation Metrics	**MSE**	**RMSE**	**R^2**	**MAE**	**RMSLE**
StackedEnsemble_AllModels	8.0565	2.8384	0.7206	1.8920	0.1527
StackedEnsemble_BestOfFamily	8.0553	2.8382	0.7206	1.8902	0.1527
XGBoost_grid__1	8.0821	2.8429	0.7197	1.8902	0.1532
XGBoost_2	8.3787	2.8946	0.7094	1.9347	0.1559

(c) Tabela de resultados relativos ao *dataset* de *cross validation*

Cross Validation Metrics	**MSE**	**RMSE**	**R^2**	**MAE**	**RMSLE**
StackedEnsemble_AllModels	8.3544	2.8904	0.7093	1.9208	0.1551
StackedEnsemble_BestOfFamily	8.3602	2.8914	0.7091	1.9189	0.1553
XGBoost_grid__1	8.3801	2.8948	0.7084	1.9178	0.1556
XGBoost_2	8.6609	2.9429	0.6987	1.9674	0.1580

Tabela 5.2: Tabelas com as respectivas métricas e seus resultados para os modelos relativos aos *datasets* treino, teste e *cross validation*

5.2 Serviço de previsão para o valor de PM2.5

Com os resultados obtidos pela análise feita com o H_2O, foi necessário arranjar forma de conseguir usar o modelo escolhido, sendo aquele que fornece os melhores resultados obtidos, para tentar prever o melhor valor para o PM2.5. Com o facto de as variáveis independentes do modelo poderem ser provenientes de vários ficheiros de entrada, consoante vários registos que se recolham neste caso em concreto para os dados dos trânsito, era necessário ter uma solução robusto e de certa forma escalável, ao ponto que num trabalho futuro se consiga prever usando outras fontes de dados que não só do trânsito. Sendo assim, desenvolvido o modelo na aplicação web do

H_2O, foi necessário exportá-los num formato comprimido. Para que os modelos possam ser utilizados externamente, o H_2O dispõe de uma biblioteca (.Jar) de seu nome `h2o-genmodel.jar`[5]. Este contém vários *packages*, com várias funcionalidades desde analisar as variáveis independentes e suas importâncias, como analisar as métricas para os *dataset* treino, teste e *cross-validation*, calcular o *score* para o algoritmo associado ao modelo e fazer as predições associadas ao tipo de modelo, quer seja binomial, multinomial, ordinal, regressão, entre outros. Uma das funções usada neste processo é a importação do conteúdo dos modelos para objectos *Java*, *Plain Old Java Object* (POJO) ou *Model ObJect, Optimized* (MOJO)[6].

Um MOJO é uma alternativa ao POJO do H2O. Assim como acontece com os POJOs, o H2O permite converter os modelos construídos em MOJOs, que podem ser *deployed* para *scoring* em tempo real.

A partir destes objectos consegue-se obter uma estrutura com vários objectos de acordo com o tipo de modelo utilizado e o tipo de previsão adequada a este. No entanto em relação a estes objectos um deles apresenta mais vantagens em relação ao outro, neste caso o MOJO é mais vantajoso principalmente quando se trata de lidar com uma maior quantidade de dados, uma vez que este também suporta todos os modelos gerados pelo AutoML ao contrário do POJO que não suporta GLRM, Stacked Ensembles, e Word2Vec.

Posto isto, foi desenvolvido um serviço em Java, denominado `Predict Traffic`. Este serviço tira partido das funcionalidades mencionadas para receber um conjunto de dados num formato específico, neste caso *csv*, sendo que este ficheiro contem a informação das características do modelo, e consiga através destas prever o classificador, valor de PM2.5, para um determinado dia, reproduzindo como *output* um novo ficheiro, do mesmo formato com cada um desses valores relativos a cada um dos dados do trânsito. A Listagem 5.3 descreve as funcionalidades utilizadas provenientes da biblioteca `h2o-genmodel.jar`, em que se pode verificar nas linhas 1 a 4,o carregamento do modelo, neste caso o StackedEnsemble_BestOfFamily para um objecto MOJO. Nas linhas de de 5 a 10 verificamos que as características recolhidas do ficheiro de *input* são introduzidas num outro objecto, objecto este que como se verifica na linha 11 é usado como parâmetro a uma função do objecto representativo do modelo para fazer uma previsão segundo uma regressão. Dai resulta um objecto com um conjunto de propriedades sendo uma delas o valor obtido da previsão (linha 12).

Com este processo concluído, para cada *input* obteve-se assim um novo ficheiro com a informação relativa ao trânsito e uma nova informação relativa ao valor previsto para

[5]http://docs.h2o.ai/h2o/latest-stable/h2o-genmodel/javadoc/index.html
[6]http://docs.h2o.ai/h2o/latest-stable/h2o-docs/mojo-quickstart.html

```
1     EasyPredictModelWrapper.Config config = new EasyPredictModelWrapper.Config()
2             .setModel(MojoModel.load("StackedEnsemble_BestOfFamily_AutoML_20200710_094937.zip"
                  ))
3             .setEnableContributions(true);
4     EasyPredictModelWrapper model = new EasyPredictModelWrapper(config);
5         RowData row = new RowData();
6         row.put("speed", record.get("speed"));
7         row.put("road_type", record.get("road_type"));
8         row.put("level", record.get("level"));
9         row.put("length", record.get("length"));
10        row.put("delay", record.get("delay"));
11        RegressionModelPrediction p = model.predictRegression(row);
12        l.add(String.valueOf(p.value));
```

Listagem 5.3: Uso da biblioteca `h2o-genmodel.jar` para prever o valor de PM2.5

PM2.5. Esta necessita de ser importada para a base de dados da aplicação de modo a que esta possa ser usada pela aplicação *dashboard* de modo a consultar e representar estes valores. Também para que se possa representar esta informação ao nível das freguesias e suas subsecções estatísticas é necessário cruzar esta informação.

```
1 \copy public.traffic_predicted (uuid,street,speed,road_type,pub_millis,level,length,end_node,
      delay,country,city,type,"ID",turn_type,bbox,geo,"DT",geom_ml,geom_ml_buffer,
      pm2_5_average_geo,pm2_5_values_geo,pm2_5_average_geo_predict) FROM 'C:\Users\rta\Documents
      \tese_repo\predictTraffic\traffic_2020-03-19WMV_predict.csv' DELIMITER ',' CSV HEADER;
2 CREATE OR REPLACE FUNCTION calc_avg_pm2_5_predicted_subseccoes(data_date timestamp)
3   RETURNS void as
4   $BODY$
5   DECLARE
6   BEGIN
7     UPDATE subseccoes_lisboa
8     SET pm2_5_average_predicted = (SELECT AVG(T.pm2_5_average_geo_predict)
9               FROM traffic_predicted AS T INNER JOIN subseccoes_lisboa AS S
10                ON (ST_Intersects(S.wkb_geometry, T.geom_ml_buffer) AND S.ogc_fid =
                      subseccoes_lisboa.ogc_fid) WHERE date(T."DT") = date(data_date));
11  END;
12  $BODY$
13 LANGUAGE 'plpgsql';
```

Listagem 5.4: Importação dos dados obtidos para a base de dados

A Listagem 5.4 descreve este processo de colocar a informação na base de dados de maneira a que esta se possa representar na aplicação. Este processo é desencadeado sempre que um novo modelo seja gerado e se pretenda que uma nova previsão seja feita utilizando este, ou caso se queira passar um novo conjunto de características, neste caso as correspondentes às variáveis independentes do modelo para se obterem novas previsões para o melhor modelo obtido. A linha 1 descreve a passagem do ficheiro resultante e sua informação para a base de dados. Este ficheiro é incluído com a instalação do software indicando apenas o caminho relativo deste. As linhas 2 e 3 a adição de uma nova coluna para representar o valor previsto do PM2.5 na informação relativa às regiões e suas subsecções-estatísticas. As linhas 6 a 17 descreve uma

função que tem como propósito para uma data especifica calcular o valor médio previsto para o PM2.5 para as subsecções-estatísticas, para tal a informação proveniente da previsão que contém cada um dos troços, representados por dados geográficos, relativos ao trânsito para uma certa data é intersectada com a função `ST_Intersects` sobre a informação geográfica relativa à subsecção-estatística. Para o caso das freguesias, é criada uma função que pega em cada um dos valores previstos de PM2.5 para as subsecções e é calculada a média desses à freguesia que pertence.

Dado este processo, no momento a base de dados contém a informação tanto a nível diário do trânsito e da poluição como os dados que correspondem a uma previsão do valor de poluição. Assim sendo, todos os processamentos descritos pelos serviços e aplicações dão o suporte e manutenção aos dados necessários para a aplicação. Posto isto é necessário definir as acções que transportem os dados entre a aplicação e a base de dados, e como os dados são representados dentro da aplicação. Com tudo isto é possível tirar partido das funcionalidades do *dashboard* tornando-o interactivo e responsivo às acções impostas pelo utilizador.

6

Dashboard interactivo

Todo o processamento dos dados necessários, desde a obtenção destes como todo o tipo de tratamento para que estes possam ser guardados na base de dados, ou utilizados em processos de mineração de dados, deve-se ao facto de estes serem o requisito principal para acrescentar usabilidade à aplicação cliente final. Como tal, estes têm de ser obtidos segundo as acções definidas na interface da aplicação conjuntamente com a usabilidade que o utilizador final requerer destas, fazendo com que a informação obtida possa ser filtrada, obtendo parte dos dados.

O objectivo do ponto de vista da construção da aplicação, era que esta contivesse um *dashboard* interactivo, de fácil utilização e que cumprisse com o objectivo principal em que os utilizadores finais, neste caso os operacionais da CML capazes de tomar decisões acerca da poluição no ar, possam para o poluente em análise ter uma clara apreciação das áreas envolventes e com vários níveis de detalhe para que os dados fossem compreendidos da melhor forma.

Sendo assim surgiu a PoluLx aplicação cliente, onde é possível verificar o mapa da cidade de Lisboa e realizar interacções sobre este de modo a consultar os dados presentes na base de dados e posteriormente os dados que irão ser modelados. Aqui irá ser possível verificar os resultados que irão ser obtidos da modelação, onde estes serão comparados com os dados obtidos num espaço temporal (presente e passado).

A Figura 6.1 representa a arquitectura da aplicação PoluLx, que usa uma arquitectura Redux dado que a aplicação foi construída usando *React Javascript*. Consiste em ter 4

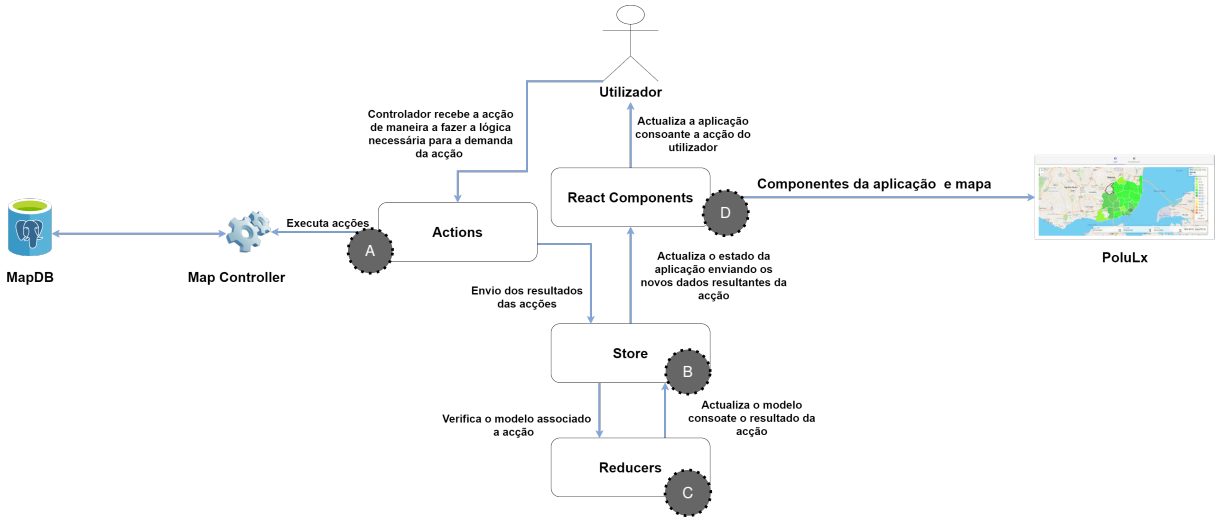


Figura 6.1: Arquitectura da aplicação cliente

peças fundamentais, as componentes da aplicação (Figura 6.1-Ⓓ), a *store* (Figura 6.1-Ⓑ) responsável por guardar o estado da aplicação e actualizá-lo consoante as acções do utilizador, as *actions* (Figura 6.1-Ⓐ) definidas para responder aos pedidos do utilizador e recolher os dados da base de dados se necessário e por fim *reducers* (Figura 6.1-Ⓒ) responsáveis por definir grande parte da lógica aplicacional, mais concretamente, vindo do tratamento das acções.

6.1 *Proxy* das acções do utilizador dentro da aplicação

Dado que a aplicação cliente necessita de consultar a base de dados é necessário ter um serviço intermediário (*proxy*) entre ambos, este definido usando a tecnologia e arquitectura do *Java Spring*[1].

Map Controller é o nome escolhido para o *proxy* entre as acções definidas na aplicação e a base de dados para dar suporte a esta. Este *proxy* tem um conjunto de *endpoints* que definem as tais acções que podem ou não receber parâmetros de modo a filtrar os dados pretendidos, estas acções em certos casos podem retornar dados de modo a que depois a aplicação saiba lidar com estes, ou a acção pode não retornar dados mas disputar uma outra acção mas do lado da base de dados, para que se processe os dados consoante o pedido vindo da aplicação, e contenha os dados actualizados perante um novo pedido que surja.

[1] https://spring.io/

Nome do Método	URL	Método HTTP
getGrid	/grid	GET
getAll	/map_point	GET
getAllByRegion	/map_point_by_region	GET
getAllBySection	/map_point_by_section	GET
getMinMaxPm	/get_max_min_pm	GET
getMaxPmRegion	/get_max_pm_region	GET
updateRegions	/map_point_by_region	PUT
updateSubsections	/map_point_by_section	PUT
updateRegionsPredicted	/map_point_by_region_predicted	PUT
updateSubsectionsPredicted	/map_point_by_section_predicted	PUT
getTrafficJams	/traffic_jams	GET

Tabela 6.1: *Endpoints* associados às acções dentro da aplicação

A Tabela 6.1 descreve a assinatura dos *endpoints* definidos no proxy. Os dois primeiros foram usados como auxílio da progressão da aplicação do ponto de vista em que esta mostra os dados, ou seja, a função `getGrid` serve para mostrar a área delimitadora sobre a cidade de Lisboa e cada um dos polígonos da grelha para obtenção dos dados iniciais, esta retorna para a aplicação os dados no formato GeoJSON. Já a função `getAll` obtém dos polígonos da grelha os seus respectivos pontos centrais e retorna para a aplicação apenas os que registaram valores de PM2.5. A linha 3 e 4 representam as funções para apresentar os valores de PM2.5 no mapa para as freguesias e subsecções, relativamente às subsecções esta recebe um `id` que corresponde ao identificador da freguesia retornando apenas as informações das subsecções correspondentes, ambas as funções retornam os dados em texto com a descrição do GeoJSON associado. As funções `getMinMaxPM` e `getMaxPmRegion` representadas nas linhas 5 e 6, serve respectivamente para obter o valor mínimo e máximo de PM2.5 entre um período de datas definido pelo utilizador, e para obter o valor máximo de PM2.5 em relação às freguesias. Para actualizar o valor de PM2.5 ao nível das freguesias e suas subsecções-estatísticas, tanto para os valores reais como os previstos, temos as funções `updateRegions, updateSubsection` e `updateRegionsPredicted e updateSubsectionsPredicted` representados pelas linhas 7 a 10, onde estas recebem uma data que corresponde à filtragem dos dados pela mesma, à excepção da função de actualização das freguesias com o valor previsto, dado que primeiramente é calculado ao nível das subsecções e só depois para as freguesias considerando só os valor pertencentes a estas. Por fim existe ainda uma função `getTrafficJams` que consulta os dados obtidos do Waze e guardados na base de dados, de maneira a que se possa representar na aplicação os vários troços consoante o nível de trânsito para uma

determinada data.

6.2 Aplicação cliente *dashboard* e sua evolução

Relativamente à aplicação em termos de ferramentas foi utilizado um aplicação em *React Javascript* definindo uma arquitectura (Figura 6.1) suportada para aplicações construídas em *React* e a biblioteca *Leaflet*[2] no que diz respeito aos mapas. Com isto e o que está definido nas base de dados é nos possível representar por freguesias ou se pretendido por um maior nível de zoom por sub-secções estatísticas os valores obtidos para as partículas PM2.5 para cada ponto contido na região.

```
1   var mymap = L.map("mapid")
2   var tileLayer = L.tileLayer('https://api.mapbox.com/styles/v1/{id}/tiles/{z}/{x}/{y}?
        access_token={accessToken}', {
3       attribution: 'Map data &copy; <a href="https://www.openstreetmap.org/">
            OpenStreetMap</a> contributors, <a href="https://creativecommons.org/licenses/
            by-sa/2.0/">CC-BY-SA</a>, Imagery © <a href="https://www.mapbox.com/">Mapbox</
            a>',
4       maxZoom: 18,
5       id: 'mapbox/streets-v11',
6       // id: 'mapbox/satellite-streets-v11',
7       tileSize: 512,
8       zoomOffset: -1,
9   });
10  tileLayer.addTo(mymap);
11  var mymap = mymap.fitBounds([
12      [38.69139935, -9.22983565],
13      [38.69139935, -9.08633286],
14      [38.79675837, -9.08633286],
15      [38.79675837, -9.22983565],
16      [38.69139935, -9.22983565]
17  ]);
18  var bounds = mymap.getBounds();
19  mymap.setView(bounds.getCenter(), 12);
```

Listagem 6.1: Carregamento do mapa dentro da componente deste na aplicação

A arquitectura da aplicação foi construída não só usando o modelo para a ferramenta utilizada como para dar suporte às acções de utilização e lidar com a representação do mapa. Esta segue o fluxo próprio começando por definir a componente do mapa esta que tira partido de funcionalidades do *Leaflet*. A Listagem 6.1 apresenta o carregamento do mapa através do *Leaflet*. Como se pode verificar este obtém o mapa de uma api que fornecida pelo mapbox[3]. Esta API necessita de um registo prévio de modo a obter-se uma chave de acesso e um conjunto de outros parâmetros necessários como o tipo de representação (e.g ruas, satélite), e o tamanho inicial do mapa.

[2] https://leafletjs.com/
[3] https://docs.mapbox.com/api/overview/

Para além do mapa outras componentes (Figura 6.1-D) foram definidas como a informação do valor do PM2.5 e freguesia seleccionada (Figura 6.5.B), legenda das cores do mapa(Figura 6.5.B), escolha temporal da representação do mapa(Figura 6.5.D) e valores mínimos e máximos de PM2.5 (Figura 6.5.E). Para ajudar na realização destas componentes, foi usado o *Material-UI*[4]. Este fornece algumas componentes desenvolvidas em *React*, de modo a que dentro da aplicação apenas se tenha de as importar e definir um conjunto de propriedades que estas apresentem, como por exemplo a definição de acções destas, estilos e tamanho.

```
1  var dateRange = L.control({position: 'bottomright', display: 'inline-flex'});
2  dateRange.onAdd = (map) => {
3      this._div = L.DomUtil.create('div', 'dateRange'); // create a div with a class "
           info"
4      return this._div;
5  };
6  // method that we will use to update the control based on feature properties passed
7  dateRange.update = () => {
8      ReactDOM.render(
9          <MuiPickersUtilsProvider utils={DateFnsUtils}>
10           <Grid container justify="space-around" >
11             <KeyboardDatePicker
12               style={{marginRight:8}}
13               margin="dense"
14               id="date-picker-dialog"
15               label="Map Date"
16               format="yyyy/MM/dd"
17               value={this.state.value.mapDate}
18               onChange={this.handleMapDateChange}
19               onAccept={this.acceptMapDateChange}
20               KeyboardButtonProps={{
21                 'aria-label': 'change date',
22               }}
23             />
24           </Grid>
25         </MuiPickersUtilsProvider>
26         , this._div)
27 };
28 dateRange.addTo(mymap);
```

Listagem 6.2: Definição da componente da escolha da data do mapa

A Listagem 6.2 apresenta uma das componentes construídas neste caso define a componente de escolha da data do mapa esta tira partido de componentes provenientes do *Material-UI* como `KeyboardDatePicker` passando-lhe as propriedades necessárias, esta no fim é adicionada ao mapa (linha 28) de maneira a que esta componente fique sobre o mapa numa das suas margens possíveis.

Dado que as componentes são fundamentais para a interacção do utilizador na aplicação estas definem acções (e.g Listagem 6.2 linhas 18 e 19), que podem ser apenas

[4]https://material-ui.com/

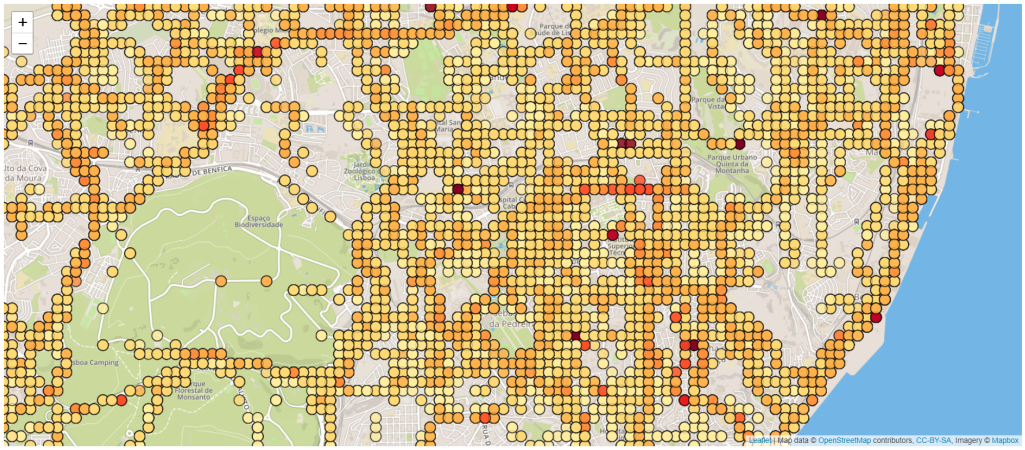


Figura 6.2: Representação do mapa por pontos coloridos consoante o valor de PM2.5

para actualizar o estado da aplicação sobre alguma lógica aplicada nestas ou necessitarem de consultar e actualizar dados presentes na base de dados(Figura 6.1-Ⓐ). Por exemplo a acção `acceptMapDateChange` (Listagem 6.2 linha 19) deve-se ao facto de o utilizador escolher uma data para visualizar os dados no mapa, para tal esta acção define comunicações ao proxy `MapController` para obter os dados para as freguesias, sub-secções estatísticas e trânsito estas para a data escolhida. Quando se obtém os dados provenientes destes acessos, é necessário definir uma lógica associada a cada uma destas acções. Mas, antes disso, os dados passam pela *Store* da aplicação (Figura 6.1-Ⓑ) para transportar o estado da aplicação antes e depois da acção, e é nos *Reducers* (Figura 6.1-Ⓒ) que é definida a lógica necessária para que o estado seja actualizado consoante os novos dados. Por fim a *Store* é informada deste novo estado e informa as componentes que têm de ser renderizadas de modo a que o utilizador veja a sua acção a ter sucesso.

A aplicação final cliente foi sofrendo alterações a nível da interface e tal deve-se ao facto de à medida que se construía as base de dados a maneira como os dados eram mostrados foram sofrendo algumas alterações, isto também sempre a pensar na melhor experiência de utilização. Primeiramente sobre a área delimitadora da cidade de Lisboa e os respectivos polígonos representados a partir da grelha, como se verifica na Figura 4.2 eram recolhidos os valores para os pontos centrais e eram representados sobre o mapa com uma escala de cores representando a intensidade destes. Como se pode verificar na Figura 6.2, inicialmente estes pontos foram representados com pequenas circunferências dentro da aplicação mostrando uma quantidade elevada destas com uma certa distinção nos seus valores e de certa forma alguma dificuldade de analisar estas ao pormenor de um ponto vista geográfico.

Contudo, a segunda abordagem resultou na representação dos pontos por freguesias

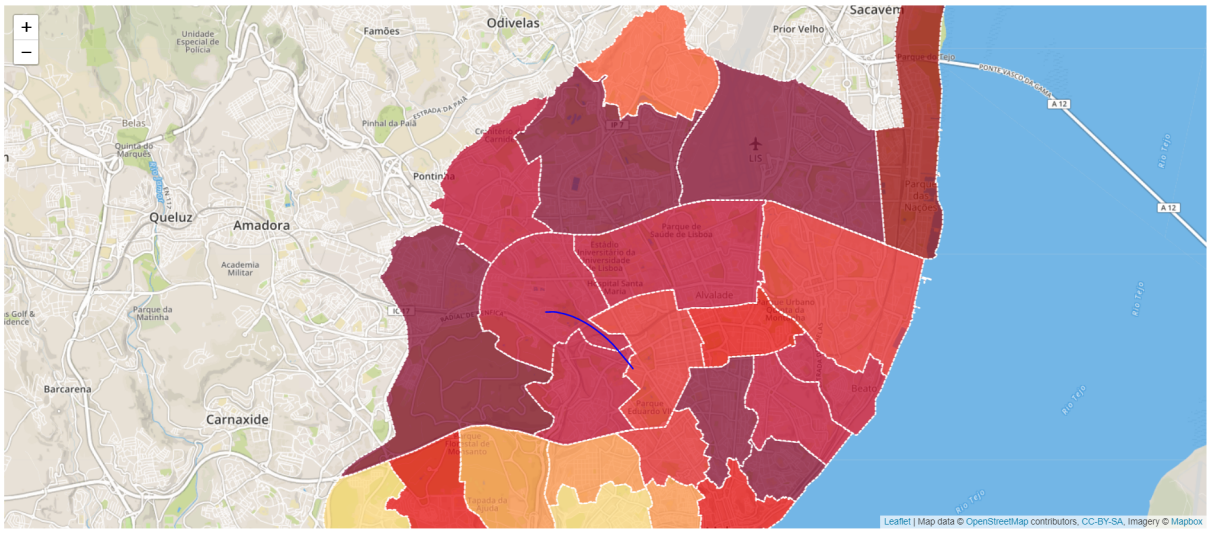


Figura 6.3: Representação do mapa por freguesias

como se pode verificar na Figura 6.3, ou seja, do modo de representação deixaríamos de ter pontos e passaríamos a ter zonas no mapa com uma certa cor. Para tal era necessário conhecer os limites de cada freguesia e, para os pontos já calculados, verificar quais estavam contidos nestes. Assim, aplicava-se uma métrica para cada zona, neste caso aplicou-se uma média dos valores dos pontos contidos na região de interesse.

Com a representação ao nível das freguesias verificou-se que esta poderia não ser a forma de representação mais pormenorizada, visto que dentro de uma freguesia com um certo nível elevado de área poderia não haver pontos, pois estes podem estar mais representados numa zona dentro da freguesia que outra. De modo a que se possa verificar este facto, quando o utilizador pretende observar uma certa freguesia, o mapa aumenta o zoom e a representação passa a ser por sub-secções estatísticas [12]. Na Figura 6.4 comprova-se a visualização para uma freguesia, está dividida nas suas sub-secções estatísticas.

A escala de cores representada foi definida pela agência europeia do ambiente segundo o *EU Air Quality Index* (EAQI) [6], que define 6 bandas de cores: 1. Good (Green) 2. Fair 3. Moderate 4. Poor (Yellow) 5. Very poor 6. Extremely poor (Red)

A Figura 6.5 representa o estado final da aplicação não só com o mapa e seus níveis de visualização como também várias acções utilitárias para ajudar a análise do utilizador. A Figura 6.5.Ⓐ representa a área do mapa onde se pode observar dentro da cidade de Lisboa os valores do poluente para cada freguesia. Estes valores podem ser observados pela acção do utilizador ao passar com o rato por cima destas como se pode observar na Figura 6.5.Ⓕ, ao mesmo tempo que é feita esta acção é observado pela Figura 6.5.Ⓑ

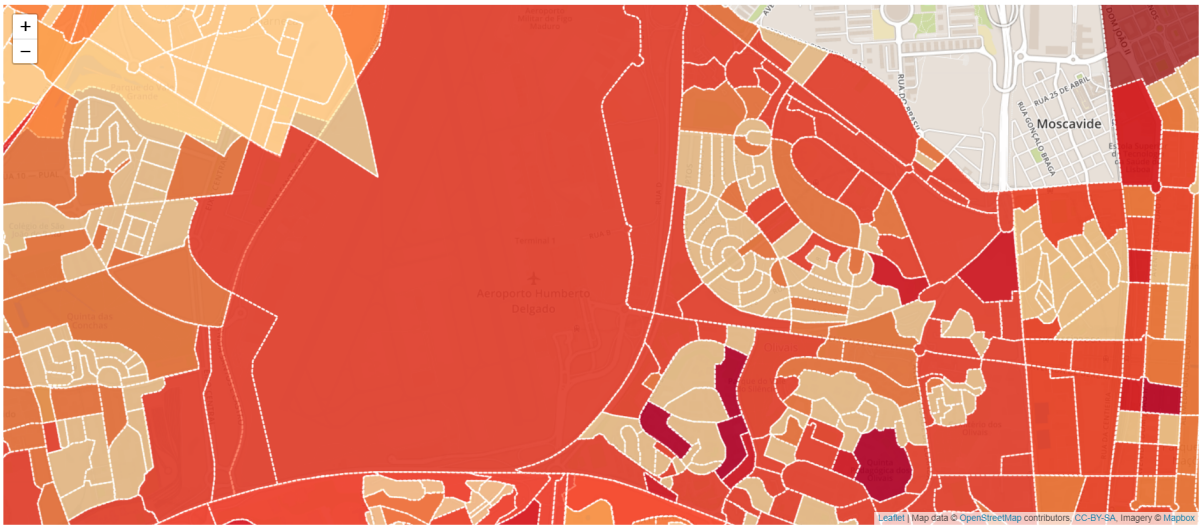


Figura 6.4: Representação do mapa por sub-secções estatísticas

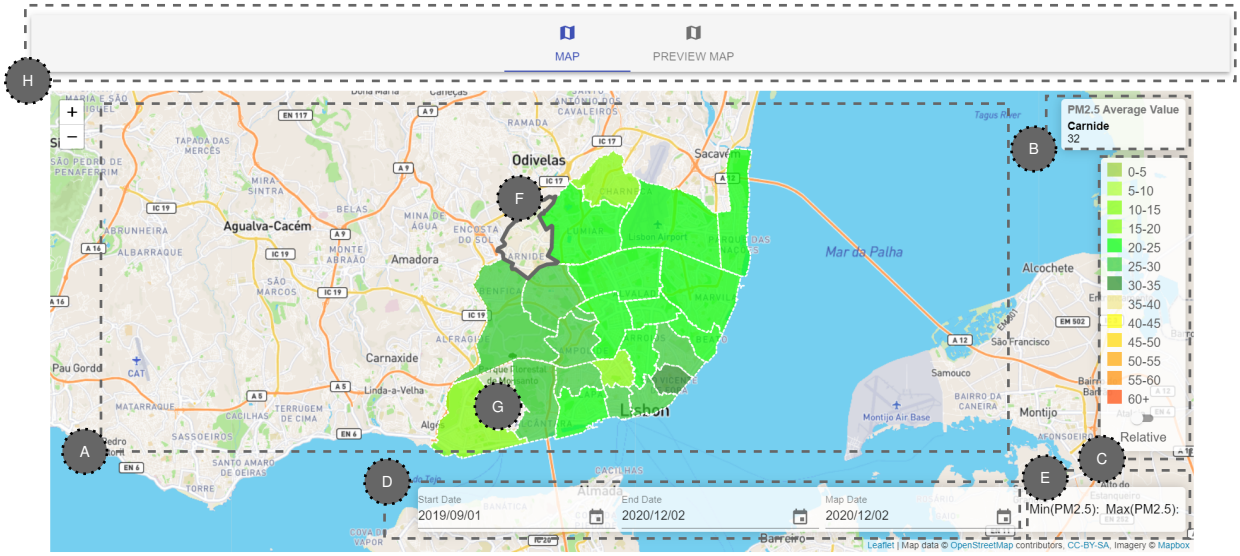


Figura 6.5: Interface do utilizador do mapa interactivo. As áreas destacadas representam os principais componentes para a interacção e para obter *feedback* visual com base na interacção

o nome da freguesia em questão e o seu valor. Segundo o que é validado pelos dados e posteriormente calculado para se obter o valor do poluente para um certa freguesia, pode acontecer o caso em que esta para uma certa data não apresente registos, sendo assim, a sua representação é nula do ponto de vista a aplicar uma cor da escala, este facto observa-se pela Figura 6.5.G. Como se observa na região do mapa este segue uma escala e esta mesma está presente na aplicação, Figura 6.5.C, não só para validar os valores, mas com a particularidade de se poder alternar entre uma outra escala

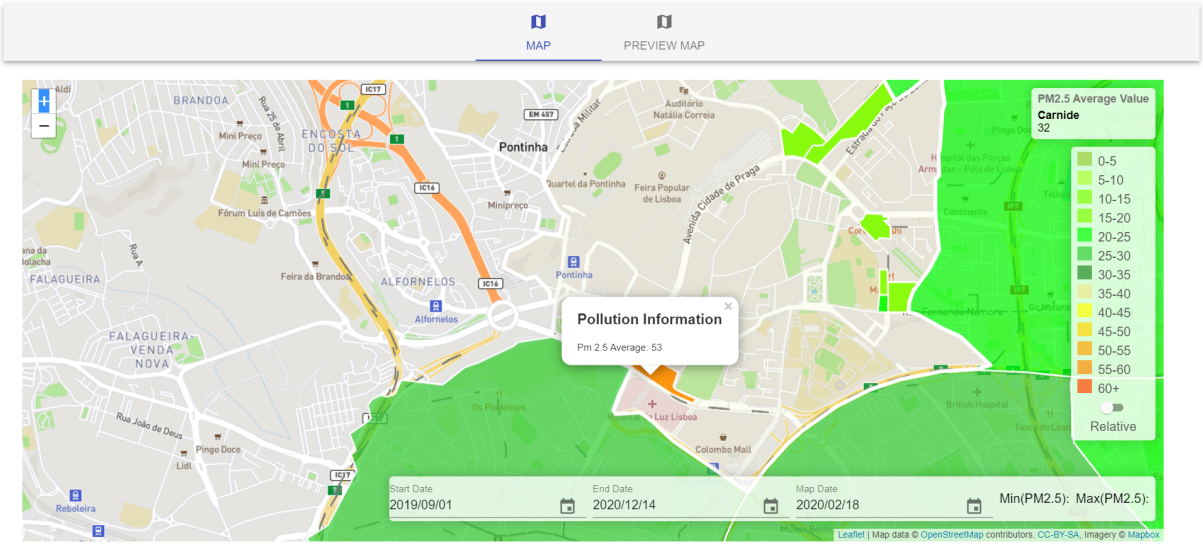


Figura 6.6: Representação do mapa ao nível das subsecções mostrando o valor do poluente para uma subsecção

que é calculada segundo o maior valor registado de PM2.5, podendo auxiliar o utilizador que apesar de por norma os valores aparentam ser bons segundo a escala europeia, dentro destes quais os mais preocupantes e que requerem alguma atenção. Do ponto de vista temporal, Figura 6.5.Ⓓ, a aplicação fornece a opção de escolher a data do mapa que se pretende analisar os resultados, e ou, escolher a data mínima e máxima de modo a se obter os valores mínimos e máximos desse período, Figura 6.5.Ⓔ. De maneira a incorporar a análise de resultados segundo os valores previstos para o PM2.5, acrescentou-se no topo da aplicação a possibilidade de alternar entre mapas, Figura 6.5.Ⓗ, neste caso a representação na figura é em tempo real, mudando para o *preview map* obtém-se um novo mapa apresentando os resultados previsto para o dia(s) seguinte ao que foi definido no mapa em tempo real.

Na Figura 6.6 observa-se a visualização do mapa ao nível das sub-secções estatísticas, após a acção do utilizador ao clicar na freguesia (Figura 6.5.Ⓕ) pretendida esta é expandida aumentando o *zoom* do mapa aparecendo as secções com valores registados e ao clicar sobre uma delas pode-se observar o valor de PM2.5 desta. Para além da informação relativa ao poluente, ao nível das sub-secções estatísticas também é possível observar os troços relativos ao trânsito para o mesmo dia (Figura 6.7), estes que são representados por linhas consoante o atributo do trânsito ,`level`, que caracteriza a gravidade do troço. O valor deste atributo varia entre 0-5 sendo que este valor contribui na sua representação sendo que o 0 representa um troço picotado com uma grande distância, diminuindo à medida que o valor aumenta sendo que o 5 representa uma

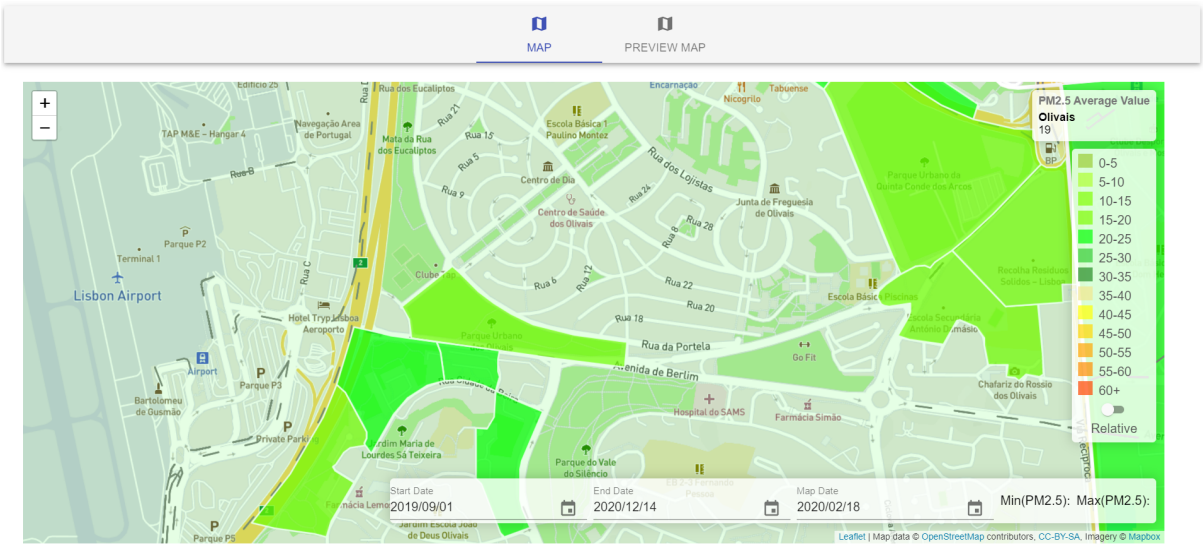


Figura 6.7: Representação do mapa ao nível das subsecções mostrando vários troços segundo o nível médio do trânsito registados nesse dia

linha completa.

7

Conclusões

O objectivo desta dissertação era desenvolver uma aplicação que representá-se um *dashboard* interactivo sobre um mapa, que permite disponibilizar informação e monitorização ao município de Lisboa, sobre o estado actual ou passado histórico do poluente, neste caso o PM2.5, sobre as freguesias e suas sub-secções estatísticas da cidade. Um dos outros objectivos da aplicação é permitir a previsão destes valores, tirando partido de outros dados como o transito e dados meteorológicos. O primeiro passo para a concretização da aplicação foi o desenvolvimento das componentes e serviços que permitem obter, processar e guardar os dados relativos ao poluente. Nesta parte utilizou-se uma base de dados com uma solução para guardar dados geográficos. A junção desta base de dados com um controlador que permite-se fazer a gestão dos dados a enviar para a aplicação permite ver, do ponto de vista aplicacional, a representação dos dados no mapa e todo o tipo de interactividade sobre este. Os objectivos relativamente a esta troca de dados foram cumpridos, na medida em que os utilizadores conseguem realizar todo o tipo de funcionalidade dentro da aplicação, desde definir as datas de visualização do mapa, a escolha de freguesia e subsecção estatística, a escolha da legenda do mapa, e escolher entre visualizar o mapa em tempo real, ou dados previstos.

Depois deste desenvolvimento e teste de usabilidade, seguiu-se o processamento de mais fontes de dados, neste caso de trânsito, que nos permitisse juntar aos demais, criando assim um modelo capaz de prever, dado um conjunto de características relativas ao trânsito o valor do poluente para um certo dia, no período máximo de 1 semana.

Assim sendo a resposta à questão "Existe uma relação causa efeito entre tráfego e condições atmosféricas e o valor de concentração de PM2.5 em determinadas zonas geográficas da cidade de Lisboa?", comprova-se pelo resultado da dissertação e seus resultados obtidos, apesar de estes não conterem a informação das condições atmosféricas logo a relação existente é apenas entre o tráfego e o valor de concentração de PM2.5.

A aplicação, com um `dashboard` interactivo sobre um mapa, é uma peça fundamental da solução pois permite ao utilizador interagir com toda a informação. Esta aplicação *Web* que assenta na API definida no *proxy* de controlo do mapa permitindo interacções neste lidando com todos os dados utilizados.

Como resultado do trabalho desenvolvido, foi escrito e apresentado um artigo na conferência internacional *IV2020 - 24th International Conference on Information Visualisation*, intitulado "Exploring Air Quality Using a Multiple Spatial Resolution Dashboard - A Case Study in Lisbon" [27].

7.1 Trabalho futuro

O modelo resultante com base nos dados recolhidos apresenta bons resultados, no entanto apenas os dados da poluição conjuntamente com os do trânsito (WAZE) pode demonstrar algum grau de incerteza nalgum dos pontos em análise mais críticos sobre o mapa. Uma futura abordagem seria o estudo de mais fontes de dados que possam contribuir para este problema em análise, e por consequência novos modelos a surgir resultantes dessa análise.

Um outro aspecto a explorar e adaptável à solução do *dashboard* interactivo seria a possibilidade de escolha e análise de outros poluentes como o PM10, ozono (O_3) e dióxido de azoto (NO_2).

A fonte de dados usada relativa à captura dos valores do poluente aparenta uma única granularidade temporal (média deslizante dos últimos 7 dias), dado o nível de detalhe que o utilizador final pretende analisar, seria bom explorar outras fontes ou ferramentas que possam fornecer estes dados com uma margem temporal mais reduzida.

Referências

[1] Fatma Bouali, Abdelheq Guettala & Gilles Venturini, "VizAssist: an interactive user assistant for visual data mining", *The Visual Computer*, vol. 32, n.º 11, páginas 1447–1463, 2016.

[2] Leo Breiman, "Bagging predictors", *Machine learning*, vol. 24, n.º 2, páginas 123–140, 1996.

[3] ——, "Stacked regressions", *Machine learning*, vol. 24, n.º 1, páginas 49–64, 1996.

[4] Tianqi Chen & Carlos Guestrin, "Xgboost", *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. DOI: 10.1145/2939672.2939785. URL: http://dx.doi.org/10.1145/2939672.2939785.

[5] Sourangsu Chowdhury & Sagnik Dey, "Cause-specific premature death from ambient PM2.5 exposure in India: Estimate adjusted for baseline mortality", *Environment International*, vol. 91, páginas 283–290, 2016.

[6] European Comission, *European Air Quality Index*, online: https://www.eea.europa.eu/themes/air/explore-air-pollution-data, 2020.

[7] *Air quality in europe 2019*, 2019. URL: https://www.eea.europa.eu//publications/air-quality-in-europe-2019.

[8] Yoav Freund, Robert Schapire & Naoki Abe, "A short introduction to boosting", *Journal-Japanese Society For Artificial Intelligence*, vol. 14, n.º 771-780, pág. 1612, 1999.

[9] *H20*. URL: https://www.h2o.ai/.

[10] Erin LeDell & Sebastien Poirier, "H2O AutoML: Scalable automatic machine learning", *7th ICML Workshop on Automated Machine Learning (AutoML)*, 2020. URL: https://www.automl.org/wp-content/uploads/2020/07/AutoML_2020_paper_61.pdf.

[11] Mattias Hallquist, John Munthe, Min Hu et al., "Photochemical smog in China: Scientific challenges and implications for air-quality policies", *National Science Review*, vol. 3, n.º 4, páginas 401–403, 2016.

[12] Instituto Nacional de Estatistica, *Base Cartográfica*, online: https://censos.ine.pt/xportal/xmain?xpid=CENSOS&xpgid=censos_base_cartogr, 2014.

[13] *Instituto Português do Mar e da Atmosfera (IPMA): Previsão 10 dias, horária diária, localidade*. URL: https://www.ipma.pt/pt/otempo/prev.localidade.hora/.

[14] Ravi Kishore Kodali, Snehashish Mandal & S Shahrukh Haider, "Flow based environmental monitoring for smart cities", em *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, 2017, páginas 455–460.

[15] Abel Martins, Mario Cerqueira, Francisco Ferreira, Carlos Borrego & Jorge H Amorim, "Lisbon air quality: Evaluating traffic hot-spots", *International Journal of Environment and Pollution*, vol. 39, n.º 3-4, páginas 306–320, 2009.

[16] Sandra Maria Pereira Mesquita, "Modelação da distribuição espacial da qualidade do ar em lisboa usando sistemas de informação geográficas", Tese de Doutoramento, 2010.

[17] Mario J Molina & Luisa T Molina, "Megacities and atmospheric pollution", *Journal of the Air & Waste Management Association*, vol. 54, n.º 6, páginas 644–680, 2004.

[18] *Red*. URL: https://nodered.org/.

[19] Serviço Nacional de Saúde, *Organização Mundial de Saúde (OMS): Poluição atmosférica*. URL: https://www.sns.gov.pt/noticias/2018/05/02/oms-poluicao-atmosferica/.

[20] Noemí Pérez, Jorge Pey, Michael Cusack, Cristina Reche, Xavier Querol, Andrés Alastuey & Mar Viana, "Variability of particle number, black carbon, and pm10, pm2. 5, and pm1 levels and speciation: Influence of road traffic emissions on urban air quality", *Aerosol Science and Technology*, vol. 44, n.º 7, páginas 487–499, 2010.

[21] *Qualidade do ar em paris*. URL: https://capgeo.sig.paris.fr/Apps/QualiteAirParis/.

[22] Ignacio Ruiz-Guerra, Valentín Molina-Moreno, Francisco J Cortés-García & Pedro Núñez-Cacho, "Prediction of the impact on air quality of the cities receiving cruise tourism: The case of the port of barcelona", *Heliyon*, vol. 5, n.º 3, e01280, 2019.

[23] Wolfram Schlenker & W Reed Walker, "Airports, air pollution, and contemporaneous health", *The Review of Economic Studies*, vol. 83, n.º 2, páginas 768–809, 2016.

[24] António Serrador, João Tremoceiro, Nuno Cota, Nuno Cruz & Nuno Datia, "iLX - A Success Case in Public Tender Methodology", em *ProjMAN 2018 - International Conference on Project MANagement*, 2018.

[25] *Sqlite*. URL: https://www.sqlite.org/index.html.

[26] Mark J. van der Laan, Eric C Polley & Alan E. Hubbard, "Super learner", *Statistical Applications in Genetics and Molecular Biology*, vol. 6, n.º 1, 16 Sep. 2007. DOI: https://doi.org/10.2202/1544-6115.1309. URL: https://www.degruyter.com/view/journals/sagmb/6/1/article-sagmb.2007.6.1.1309.xml.xml.

[27] M.P.M. Pato João Moura Pires Ruben Taborda Nuno Datia, "Exploring air quality using a multiple spatial resolution dashboard - a case study in lisbon", em *IV2020 - 24th International Conference on Information Visualisation*, 2020.

[28] The World Air Quality Index project, *Poluição Atmosférica Mundial: Índice de Qualidade do Ar em tempo real*. URL: https://waqi.info/pt/.

[29] *Trafair air quality dashboard*. URL: https://trafair.eu/airquality/.

[30] David H. Wolpert, "Stacked generalization", *Neural Networks*, vol. 5, páginas 241–259, 1992.

[31] Kanit Wongsuphasawat, Dominik Moritz, Anushka Anand, Jock Mackinlay, Bill Howe & Jeffrey Heer, "Voyager: Exploratory analysis via faceted browsing of visualization recommendations", *IEEE transactions on visualization and computer graphics*, vol. 22, n.º 1, páginas 649–658, 2015.

[32] YF Xing, YH Xu, MH Shi & YX Lian, "The impact of PM2.5 on the human respiratory system.", *Journal of Thoracic Disease*, vol. 8, n.º 1, E69–74, 2016.