# Análise de Redes Sociais com Aprendizado de Máquina para Prever o Fluxo do Tráfego de Veículos em Zonas Urbanas

IME-USP

Apoio: FAPESP USP

Aluno: **Lucas de Carvalho Dias**

Orientador: **Roberto Marcondes Cesar Junior**

Departamento de Ciência da Computação

Página: https://www.linux.ime.usp.br/~luketis/mac0499/

## Introdução

Em 2015, eram publicados cerca de 350 mil *tweets* por minuto. Estes continham informações sobre eventos e fatos que ocorreram no cotidiano dos usuários. Por incluírem parâmetros geográficos e temporais, essas publicações podem ser relacionadas com dados de trânsito, de modo a extraírem padrões que relacionem os dois tipos de parâmetros indicados. O objetivo deste projeto é desenvolver e implementar um método que usa a fusão desses dados em algoritmos de aprendizado de máquina, para, dessa forma, identificar automaticamente se um *tweet* está informando sobre o trânsito e, caso o esteja, a que tipo de evento o texto da publicação diz respeito. Em um período de duas semanas, foram coletados cerca de 300 Gigabytes de eventos de tráfego de veículos na cidade de São Paulo, e 10 Gigabytes de publicações do Twitter geo-localizadas na mesma cidade. Os dois tipos de dados foram filtrados e formatados para poderem ser fundidos. No resultado da fusão, foram empregadas técnicas de processamento de linguagem natural, com objetivo de prepará-lo para alimentar as rotinas de aprendizado de máquina implementadas.

## Objetivos

O objetivo deste projeto é estudar e desenvolver técnicas de mineração de textos (*text mining*) e aprendizado de máquina capazes de relacionar dados urbanos de fontes heterogêneas, bem como obter tal conjunto de dados.

## Metodologia

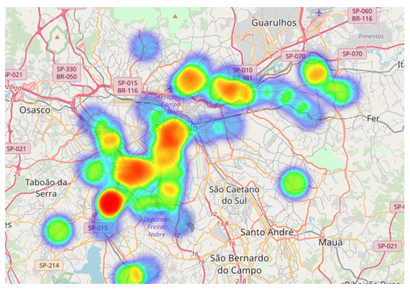


**Figura 1:** *Heatmap* dos incidentes que ocorreram as 7:43:00 no dia 12/06/2017 - Segunda-Feira

O processo proposto neste trabalho é dividido em duas partes: Coleta de Dados e Análise dos Dados coletados. Para realizar a primeira etapa do trabalho foram implementados, para cada fonte de dados, rotinas que ficariam em execução durante o período de coleta enviando requisições (usando as *API's*:*Streaming API [3], Here Traffic Flow e Traffic Incidentes [1] e Openweathermap [2]*) em intervalos de tempo para estas fontes e armazenando as respostas. O conjunto gerado foi então submetido a um processo de tratamento que envolve filtragem e formatação. Como as requisições retornavam arquivos JSON, o *dataset* primeiro foi convertido para um formato de tabela CSV. Após a conversão as entradas duplicadas foram eliminadas, e os meta-dados que não seriam usados foram descartados.

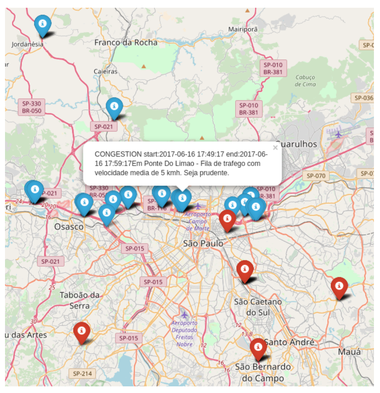


**Figura 2:** Mapa com marcadores dos Incidentes das classes Acidente e Congestionamento as 17:42:04 no dia 12/06/2017 - Segunda-Feira

Para completar o conjunto de dados dos *tweets* com parâmetros de localização precisos, foram empregados dois métodos para obter os endereços dos textos dos *tweets* e os converter para coordenadas geográficas. O método que apresentou melhor resultado foi empregado para completar o *dataset*. O papel dos dados de trânsito é servir de rótulo para os *tweets*. Para isso é feita uma fusão dos *datasets* de *tweets* e incidentes. Cada *tweet* é emparelhado com o incidente ao qual corresponde melhor. Com o *dataset* completo, transforma-se os dados textuais em numéricos usando *word2vec* [7]. Então é feito um processo de codificação e extração de características. Enfim, o *dataset* finalizado seria usado para treinar um classificador de *tweets* de forma supervisionada.

## Enriquecendo o Conjunto de Dados

Para adaptar o problema de extrair um endereço como um problema de aprendizagem de máquina, pode-se modelá-lo da seguinte forma: para cada pedaço de texto com n *tokens*, o classificador deve ser capaz de identificar se este contém um (ou parte de um) do começo, meio ou fim de um endereço. Foram mineradas 3000 postagens de usuários que só postam avisos de trânsito. Em cada postagem, foi feita a separação em 5-gramas, resultando em 38007 5-gramas e, então, empregada uma pré-rotulação usando uma Expressão Regular:

```
(\p{Lu}\p{L}+(?:\s\p{Lu}\p{L}+|\s\p{Ll}{1,2}(?=\s\p{Lu}))+
(?: comma\s\d+)?)
```

Após executada a extração com a *regex*, os 5-gramas relacionados a cada endereço eram processados para verificar se continham o começo, meio, fim do endereço e, então, o respectivo (pré-)rótulo era dado. Feita a pré-rotulação, a rotulação de fato foi apenas uma questão de passar pela tabela, corrigindo os eventuais erros.

## Classificação

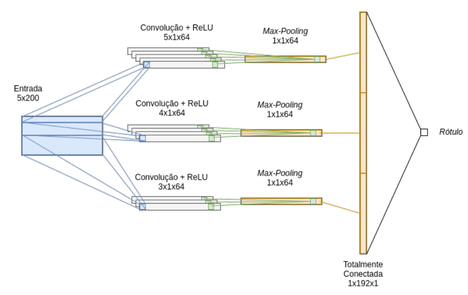


**Figura 3:** Detalhes da CNN usada no experimento de extração de endereços

A classificação dos incidentes relatados nos *tweets* é feita com dois métodos para comparar duas técnicas comumente usadas em problemas de classificação. A primeira técnica utiliza uma CNN [5] com uma camada de convolução, seguida de uma de *max-pooling* e uma totalmente conectada (*fully-connected*). O resultado da predição é obtido aplicando a função exponencial normalizada(*softmax*) no resultado da ultima camada. A função de custo utilizada para avaliar a atualização dos pesos da rede é *cross-entropy*. E o algoritmo usado para atualização dos pesos é o Adam [4].

Já a segunda técnica é uma Máquina de Vetor de Suporte (SVM - *Support Vector Machine*). É feita uma busca em grade (*GridSearch*), com seu processo explicado no diagrama da Figura 4 para encontrar os melhores valores para $\gamma$ entre $[1, 5, 10, 15]$ e $C$, com os mesmos valores testados. Todos os valores foram testados com o *kernel* linear e o RBF(*Radial Basis Function*).

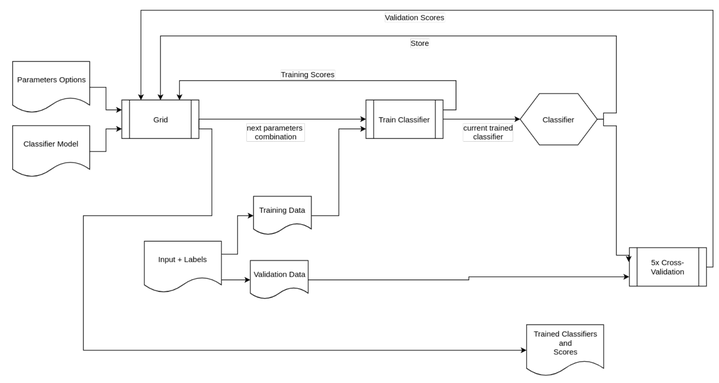


**Figura 4:** Diagrama explicando o processo de treinamento de um classificador com *GridSearch*.

## Resultados

| | Twitter | Trânsito (fluidez e incidentes) | Meteorológicos |
|---|---|---|---|
| 1ª Coleta | 36GB | 690GB e 3.5GB | 6.3GB |
| 2ª Coleta | 3.4GB | 235GB e 42GB | 7.1GB |
| Tratados | 405.7 MB | 10.7 MB(só incidentes) | 639 MB |
| Entradas | 1263289 | 15210(só incidentes) | 6080001 |

**Tabela 1:** Volume dos dados coletados

Após o treinamento da CNN, a predição no conjunto de teste obteve uma taxa de acerto de cerca de 92%, e com sensibilidades (*recall*): 97%, 78% e 81%. A performance da SVM linear e da SVM com núcleo RBF atingiram cerca de 90% e 94% de taxa de acerto, respectivamente, e obtendo sensibilidades 96%, 74% 82%(*kernel* linear) e 98%, 89% e 90%(*kernel* RBF). Os valores para os hiper-parâmetros selecionados pelo *GridSearch* foram todos 15, para o $\lambda$ e para o $C$ dos dois classificadores. No entanto, ao usar o classificador com o conjunto de dados de todas as postagens para extrair os endereços notou-se que este achava que havia endereço onde não deveria com muita frequência. Supõe-se que isso se deve ao fato do conjunto de postagens usado no treinamento apenas conter textos cujo assunto é mobilidade urbana. Uma solução para esse problema seria incluir entradas no conjunto de treinamento sobre outros assuntos, com rótulo 0.

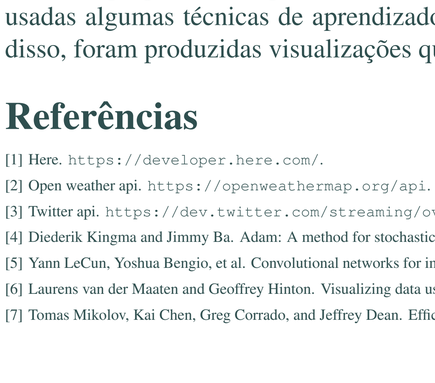


**Figura 5:** t-SNE [6] das primeiras palavras do conjunto de treinamento para extração de endereços

## Conclusões

O objetivo inicial do projeto era produzir um *dataset* a partir de coletas de dados de fontes heterogêneas, submetê-lo a um tratamento de filtragem e organização e com eles realizar análises usando aprendizado de máquina que se aproveitasse da fusão do conjunto. Conclui-se que o objetivo foi parcialmente cumprido, já que foi coletado um grande conjunto de dados de três fontes diferentes — *Twitter*, Trânsito e Meteorológico — nos quais foi feito um intensivo tratamento, e os quais também foram empregados no experimento de extração de endereços dos textos dos *tweets*, para o qual foram usadas algumas técnicas de aprendizado de máquina e processamento de linguagem natural. Além disso, foram produzidas visualizações que enriquecem a análise destes dados coletados.

## Referências

[1] Here. https://developer.here.com/.

[2] Open weather api. https://openweathermap.org/api.

[3] Twitter api. https://dev.twitter.com/streaming/overview.

[4] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[5] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[6] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.