Universidade Federal de Minas Gerais
Departamento de Ciência da Computação
Sistemas de Informação

#Coxinha

ou

#Petralha

Disciplina: Bancos de Dados Geográficos
Professor: Clodoveu Augusto Davis Júnior
Aluno: Enzo Roiz

1 - Introdução

Este trabalho tem como objetivo colocar em prática alguns dos conceitos vistos em sala de aula no durante o semestre letivo. Para este trabalho prático desta disciplina o tema escolhido foi “dados geográficos em redes sociais”. Dentro deste tema propõe-se então a aquisição e utilização de dados geográficos por meio de tweets adquiridos através de API’s fornecidas pela rede social Twitter.

Devido ao momento político do país, é possível notar certa divisão entre quem se considera de “esquerda” e quem se considera de “direita”. Levando em conta então que existem estes dois posicionamentos políticos, propõe-se então a análise de tweets relativos à situação política do país, relacionando-os à posição política do usuário baseado nas palavras que compõem seu tweet. Através da localização do tweet no espaço poderemos analisar qual a relação, se houver, entre a localização do usuário e sua posição política. Para este trabalho prático a aquisição de dados e análise foi restringida apenas à cidade de Belo Horizonte.

Nas seções a seguir serão vistas as etapas percorridas para a realização deste trabalho. Serão abordadas a coleta, o tratamento e o armazenamento dos dados obtidos, a visualização criada, além das dificuldades encontradas para o desenvolvimento deste trabalho prático.

2 - Coleta de Dados

Com o tema “dados geográficos em redes sociais”, foi escolhida então a rede social Twitter, por ser uma das mais conhecidas e utilizadas no mundo inteiro juntamente com o Facebook. Além disto o Twitter provê API’s que permitem acesso a tweets de usuários que por sua vez podem conter a localização do usuário no momento em que escreveu o tweet, permitindo associar o tweet à sua localização.

Para a aquisição de tweets foram utilizadas duas API's fornecidas pelo Twitter. A Twitter Search API e a Twitter Streaming API, melhor detalhadas a seguir. A aquisição dos tweets ocorreu entre os dias 20 de maio de 2016 e 13 de junho de 2016.

Twitter Search API: Faz parte da API REST do Twitter. Permite realizar "queries", consultas, aos tweets mais populares e tem comportamento semelhante à opção de "procurar" do Twitter, porém não é exatamente igual. A busca é realizada apenas dentre os tweets dos últimos 7 dias. Além disto a API foca em relevância e não em completude, com isto alguns tweets podem ser omitidos na busca por não serem considerados como sendo relevantes.

Twitter Streaming API: Permite acesso ao "stream" de dados do Twitter, ou seja, os clientes são "avisados" quando for "tweetado" algo de interesse. Assim através desta API pode-se adquirir tweets em tempo real. Ao contrário da Twitter Search API, é focada em completude uma vez que todos os tweets de interesse são retornados ao cliente.

Visando abstrair as dificuldades relacionadas ao acesso aos tweets, foi utilizada então a biblioteca para Python, **Tweepy**, que fornece métodos tanto para a Twitter Search API quanto para a Twitter Streaming API, facilitando a aquisição dos dados, permitindo filtros por lugares e palavras para o caso da Twitter Search API e por coordenadas geográficas no caso da Twitter Streaming API.

Para a Twitter Search API o script criado busca tweets em locais definidos por um "place", um código definido para aquele local. Para este trabalho prático foi definido que apenas os tweets em Belo Horizonte seriam de interesse, local que possui "place" *d9d978b087a92583*. Assim todos os tweets correspondentes a este place e que possuíssem alguma das palavras de interesse seriam retornados. Apesar de possuírem um "place" nem todos os tweets adquiridos tinham coordenadas geográficas, tendo que ser verificada a existência ou não após a aquisição do tweet.

Para a Twitter Streaming API o procedimento de aquisição de tweets é um pouco diferente. A Twitter Streaming API não permite que seja usado filtro por local e por palavra ao mesmo tempo. Com isto foi optado por retornar todos os tweets de Belo

Horizonte e realizar a filtragem “manualmente” verificando se no tweet havia ou não alguma das palavras de interesse e, caso houvesse, se este tweet teria ou não coordenada geográfica. Outro ponto que difere da Twitter Search API é o modo com que se define o local do qual deseja se receber tweets. Ao invés de usar um “place” como dito anteriormente, são passadas duas coordenadas geográficas, delimitando assim uma área retangular chamada de “bounding box”, como pode ser visto na figura abaixo. Assim todos os tweets com “places” dentro da área retangular definida são retornados ao cliente.

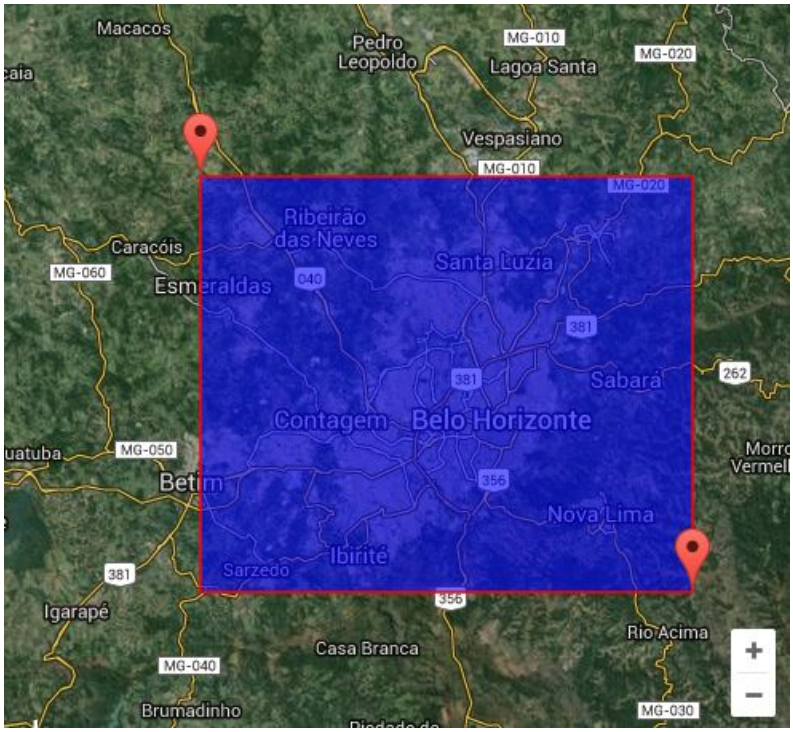

Figura 1 - Área retangular envolvendo a cidade de Belo Horizonte

Com os scripts para aquisição de tweets construídos, faltava definir então as palavras de interesse relacionadas aos seus respectivos posicionamentos políticos. Para tanto foram consideradas três diferentes classes de palavras.

Coxinha: Palavras cujo uso indicasse um posicionamento político de direita.

Petralha: Palavras cujo uso indicasse um posicionamento político de esquerda.

Potencial: Palavras cujo uso tivesse cunho político porém não indicasse um posicionamento político.

Dentre as diversas palavras utilizadas na aquisição de tweets, separando entre as classes citadas, podemos destacar algumas:

Coxinha	**Potencial**	**Petralha**
ForaDilma	Golpe	NãoVaiTerGolpe
TchauQuerida	Corrupção	ForaTemer
Petralha	Impeachment	FicaDilma
Esquerdista	Temer	VoltaQuerida
AvanteTemer	Dilma	Golpista

Com tais classes de palavras foi possível então filtrar os tweets com georreferência os separando por posição política no caso das classes coxinha e petralha e tweets a serem analisados posteriormente no caso da classe potencial.

3 - Tratamento de Dados

Com os tweets então separados por classes foi feito um tratamento “manual” que consistiu em verificar os tweets classificados nas classes “coxinha” e “petralha” analisando se o tweet era condizente ou não à posição política à qual foi classificado, o mudando de classe ou descartando caso necessário. De forma semelhante todos os tweets de potencial cunho político foram classificados em uma das classes “coxinha” ou “petralha” ou descartados, resultando na base de dados final. Esta classificação “manual” foi possível de ser feita devido ao não tão alto número de tweets adquiridos, assunto que será abordado na seção de dificuldades encontradas. A base de dados final teve um número total de 237 tweets, sendo 110 da classe “coxinha” e 127 da classe “petralha”.

4 - Armazenamento de Dados

Para o armazenamento de dados foi utilizado o banco de dados SQLite com uma estrutura bastante simples. Foi criada apenas uma entidade chamada de “Tweets” contendo o tweet, a posição geográfica através da latitude e longitude e a classe a qual o tweet pertencia, representado como um “booleano”, como mostra a imagem abaixo.

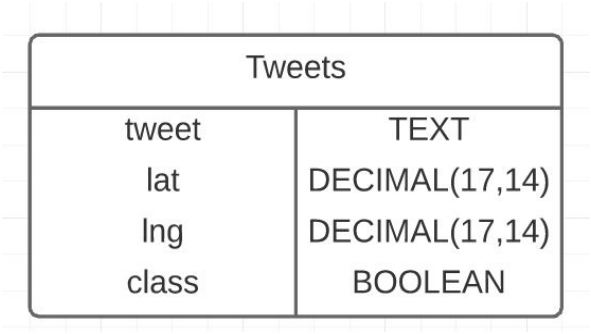

Figura 2 - Entidade Tweets

5 - Visualização

Para a visualização dos tweets, geolocalizados e classificados foi construída uma aplicação web utilizando-se o framework Django, que usa Python além da Google Maps API v3. Basicamente a aplicação consiste em plotar os tweets como pontos no mapa onde os marcadores em vermelho caracterizam posicionamento político de esquerda e os marcadores em azul caracterizam posicionamento político de direita. Ao clicar em um marcador é possível ver o tweet relacionado a ele. É possível também escolher o tipo de visualização: somente “coxinha”, somente “petralha” ou ambos através de uma caixa de seleção na parte superior direita da tela. A seguir podem ser vistas imagens da aplicação desenvolvida.

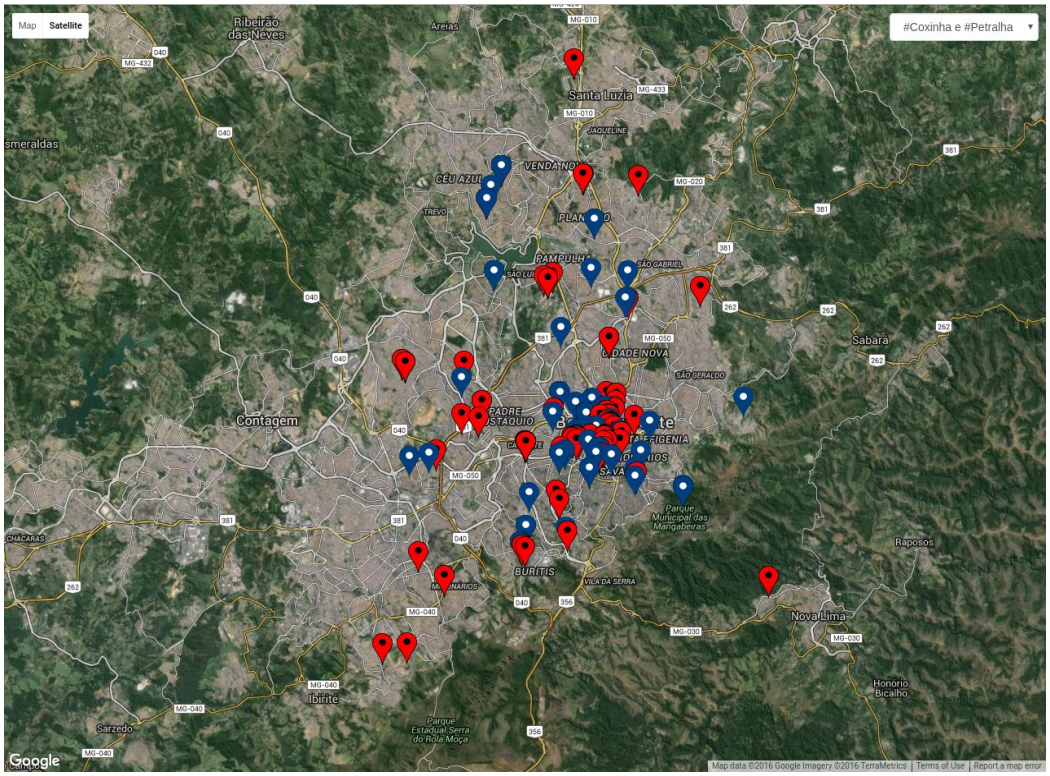

Figura 3 - Mapa de Belo Horizonte com todos os tweets

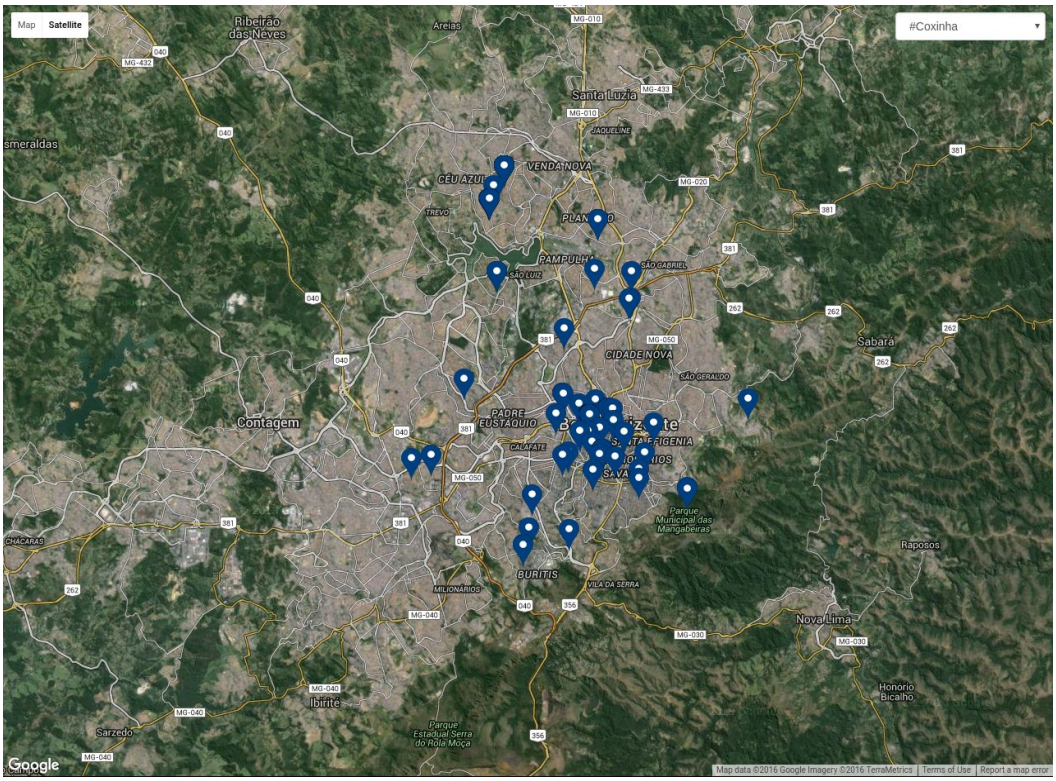

Figura 4 - Mapa de Belo Horizonte com tweets “coxinha”

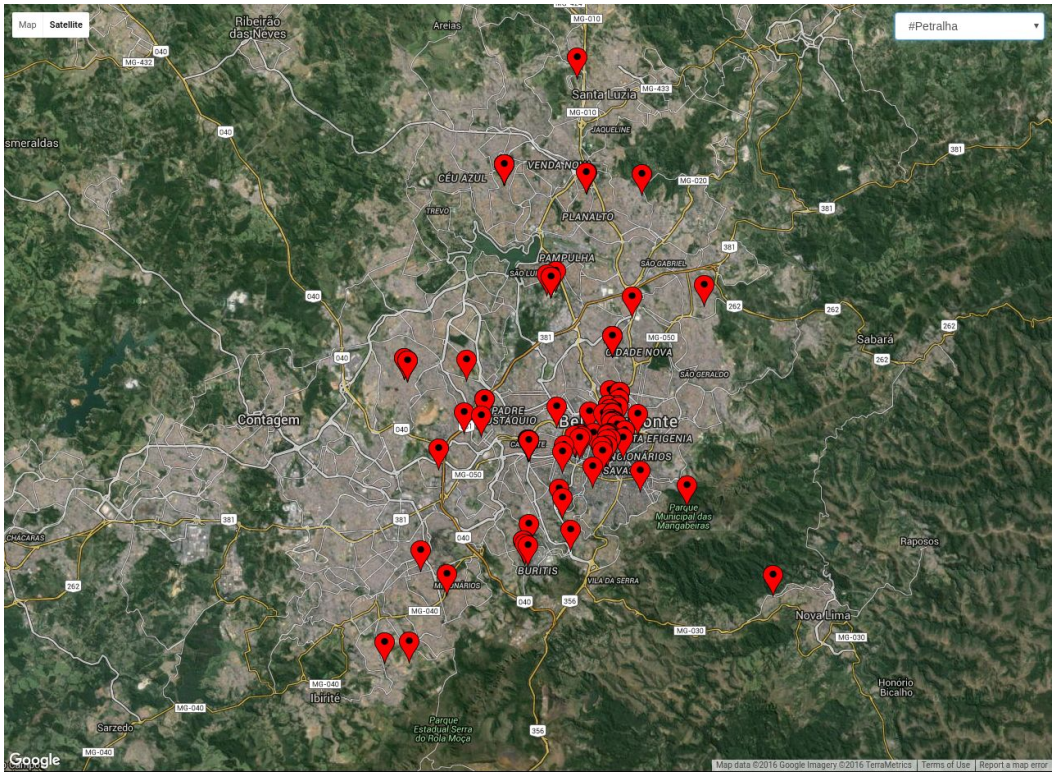

Figura 5 - Mapa de Belo Horizonte com tweets "petralha"

Figura 6 - Visualização de tweet ao clicar em marcador

6 - Dificuldades encontradas

Durante o desenvolvimento do trabalho surgiram alguns pontos que dificultaram a aquisição de dados e consequentemente uma possível análise a ser realizada. Entre eles podemos destacar os seguintes:

Utilização da Twitter Search API: A twitter search API busca tweets apenas dos últimos 7 dias. Com isto não foi possível extrair uma quantidade significante de tweets, dado que a aquisição de tweets começou em 20 de maio, ou seja, coletando apenas tweets do dia 13 de maio em diante. Não se tem, por exemplo, tweets em momentos como a votação para impeachment da, até então, presidente Dilma Rousseff, feito no dia 11 de maio, da manifestação “Fora Dilma” em 13 de março, ou da manifestação de apoio a Dilma Rousseff em 18 de março.

Um outro problema relacionado à Twitter Search API é quanto ao número de requisições realizadas. Por buscar por diversas palavras de interesse como visto anteriormente e obter um grande número de tweets como resultado, isto “ocupava” o servidor do Twitter que retornava à aplicação “Erro 429: Too many requests”, indicando que houve um número muito grande de requisições feitas. A aplicação era então banida temporariamente, ficando proibida de fazer requisições e coletar tweets, fazendo com que a busca se desse de maneira periódica, implicando em um tempo maior para a coleta dos tweets.

Outros problemas relativos à esta API foram os resultados coletados repetidamente, dada a interseção entre os intervalos de busca, tendo de ser tratada a duplicidade, além do foco desta API em relevância e não em completude, trazendo tweets relevantes porém não trazendo todos os correspondentes aos critérios de busca.

Twitter Streaming API: Esta API resolve o problema de excesso de requisições da Twitter Search API, já que é alertada quando acontece um tweet de interesse, bem como o de completude, uma vez que retorna todos os tweets de interesse. O grande

problema desta API é que como ela “escuta” o servidor o script deve permanecer rodando, com isto há de haver disponibilidade de recursos para manter o script rodando durante o tempo que se deseja coletar dados.

Quantidade de tweets georreferenciados: Um dos maiores problemas no desenvolvimento do trabalho foi a aquisição de tweets que tivessem localização geográfica. Isto pode ser visto pelo número “baixo” de tweets de interesse deste trabalho, 237 no total num período de 25 dias. Visando ter uma noção da relação entre este número, o número de tweets georreferenciados e do total de tweets, foi feita uma contagem para tweets coletados através da Twitter Streaming API. Os resultados obtidos foram:

- 348.634 tweets no total.
- 13.314 tweets georreferenciados. (4% do total)
- 57 tweets de interesse. (menos de 1% dos tweets georreferenciados)

Ilustrando o número baixo de tweets georreferenciados, apenas 4% do total coletado e um número menor ainda de tweets de interesse, menos de 1% dos georreferenciados.

7 - Análise e Conclusão

Através deste trabalho foi possível aplicar na prática alguns conceitos aprendidos na disciplina de Bancos de Dados Geográficos e entender a importância de disponibilizar informação associada à sua localização possibilitando diferentes análises. No caso deste trabalho, pelo pouco tempo de aquisição de informação e consequentemente um número baixo de tweets relacionados ao assunto escolhido, não foi possível gerar nenhuma análise relevante associando posicionamento político e localização na cidade de Belo Horizonte. Com um número maior de tweets poderia-se fazer análises como percentual de tweets de cada classe por região de Belo Horizonte, o IDH das regiões e o posicionamento político das pessoas, entre outras.

Por fim foi possível perceber a dificuldade na aquisição de informação através dos tweets que, apesar de o Twitter ser uma plataforma utilizada amplamente, raramente têm georreferência. Para este trabalho por exemplo apenas 4% (13.314) do total de tweets analisados (348.624) tinha georreferência e destes 4%, menos de 1% dos tweets (57) eram relacionados ao assunto de interesse. Os códigos utilizados para aquisição de tweets, e para gerar a visualização construída estão disponíveis em: https://github.com/enzoroiz/BDG-TP .

8 - Referências

dev.twitter.com. (2016). *The Search API | Twitter Developers*. [online] Disponível em: https://dev.twitter.com/rest/public/search [Acessado em 26 Jun. 2016].

dev.twitter.com. (2016). *The Streaming APIs | Twitter Developers*. [online] Disponível em: https://dev.twitter.com/streaming/overview [Acessado em 26 Jun. 2016].

djangoproject.com. (2016). *The Web framework for perfectionists with deadlines | Django*. [online] Disponível em: https://www.djangoproject.com/ [Acessado em 26 Jun. 2016].

Google Developers. (2016). *Google Maps JavaScript API | Google Developers*. [online] Disponível em: https://developers.google.com/maps/documentation/javascript/ [Acessado em 26 Jun. 2016].