Análise de *Tweets* com MongoDB

Amanda Karina Lopes de Oliveira[1]

[1]Pontifícia Universidade Católica de Minas Gerais (PUCMINAS) - Unidade Praça da Liberdade
R. Cláudio Manoel, 1205 - Funcionários, Belo Horizonte - MG, 30140-100

am.karina@hotmail.com[1]

Resumo. *O trabalho propõe a análise de uma base de dados coletada utilizando NodeJs e importada para o MongoDB. NodeJs é um mecanismo JavaScript leve e eficiente orientado a eventos além de ser uma plataforma aberta que dispõe de inúmeras bibliotecas[4], entre elas a biblioteca do Twitter que serve para coletar tweets em tempo real e salvá-los em uma estrutura no MongoDB. Foram coletados tweets sobre séries de TV ao longo de uma semana e foram extraídos os termos mais frequentes, o volume de tweets por dia e hora.*

Palavras-chave: nodejs, mongodb,*tweets*, análise de dados

1. INTRODUÇÃO

Desde 2012, cerca de 2,5 exabytes de dados são criados por dia, e esse número dobra a cada 40 meses. A quantidade de dados trafegados por segundo na Internet, atualmente, é maior que toda a Internet era há 20 anos[12]. O volume gigantesco de dados junto a aplicações *real-time*, que geram informações em tempo real sobre algo - sensores, smartphones, video-games, etc - nos dão um mundo de possibilidades sobre análise de dados. É possível então, ter acesso a informações além do ecossistema da empresa, onde se possui apenas os dados em seus servidores. É possível realizar consultas em redes sociais para entender o que o consumidor achou de determinado serviço ou produto e tomar decisões em cima dessas análises.

Todo esse volume de dados gerados em uma velocidade e variedade imensa necessita de uma tecnologia específica, e hoje, há uma variedade de ferramentas baratas, em grande parte livre, que oferecem o suporte necessário e com comunidade ativa. Dentre as diversas tecnologias temos o MongoDB, que além de ser livre e ter uma comunidade bem ativa, permite a construção de sistema online e offline do forma rápida, barata e em conjunto com outras tecnologias, por exemplo o Hadoop, para propósitos específicos ou não[1].

Neste trabalho, foi utilizado apenas o MongoDB para armazenamento de *tweets* coletados em tempo real e posteriormente, analisados. Nas próximas seções, descritas abaixo, serão descritas as configurações e procedimentos realizados para a execução deste.

Este artigo está organizado em 6 seções. A seção 2 apresenta o referencial teórico, dividido em Twitter, Dados estruturados e não estruturados, MongoDB, NodeJS. A seção 3 apresenta os trabalhos relacionados a este tema. A seção 4 apresenta a metodologia seguida durante a experimentação. A seção 5 descreve o processo da experimentação e seus resultados. A seção 6 apresenta as conclusões obtidas e possíveis continuações deste trabalho.

2. REFERENCIAL TEÓRICO

Para melhor entendimento deste trabalho, são necessários conhecimentos sobre alguns assuntos que serão apresentados neste seção.

2.1. Twitter

O Twitter é uma rede social de microblogging que permite aos usuários enviar e receber atualizações pessoais de outros contatos, por meio do website do serviço, por SMS e por softwares específicos de gerenciamento. A rede possui cerca de 313 milhões de usuários ativos por mês e conta com mais de 1 bilhão de visitas mensais únicas a sites com *Tweets* Embedado, que é tipo de tag HTML para media, usada para incorporar arquivos multimedia de áudio e vídeo.[6].

Embora todos usem o Twitter de maneiras diferentes, segundo [7] é possível notar alguns usos da rede.

- **Notícias e política:** Leia as últimas notícias locais e mundiais, assista a eventos políticos enquanto eles acontecem e participe de comunidades e acontecimentos sociais.
- **Esportes:** Dos Jogos Olímpicos à Série A, receba atualizações em tempo real, saiba o que os principais jogadores têm a dizer e conecte-se com fãs de todo o mundo.
- **Cultura pop:** Para ver o que suas celebridades favoritas estão fazendo, siga-as no Twitter e entre na conversa.
- **Influenciadores:** Saiba o que formadores de opinião e especialistas em seu segmento estão dizendo e conecte-se com celebridades, artistas e criadores.
- **Serviços públicos:** Obtenha informações e atualizações para viajantes, receba suporte e atualizações sobre desastres e entre em contato com agentes de atendimento ao cliente sem precisar fazer uma ligação.

O Twitter também conta com alguns jargões que são importantes para o entendimento do trabalho:

- **Tweet:** É a postagem feita, pode ser um texto de até 40 caracteres associado ou não a uma imagem ou vídeo.
- **Retweet:** É o compartilhamento com a sua rede de outro Tweet.
- **Seguir:** Possibilita ver os *Tweets* em tempo integral de outras contas.
- **Busca:** Permite saber mais sobre determinado assunto, fazendo uma busca e acompanhe a conversa em tempo real.
- **Hashtag:** As hashtags conectam *Tweets* que falam sobre o mesmo assunto em um só lugar. É possível acessar todos os *tweets* que contenham a hashtag apenas clicando nela.

2.2. Dados estruturados e não estruturados

Os dados podem ser divididos em Estruturados, Semi-Estruturados e Não Estruturados.

Os dados estruturados possuem uma estrutura prévia, são organizados em grupos semânticos, que possuem agrupamento e descrições além de definição do tipo do dado, números, data, texto, etc.

Os dados semi-estruturados são dados onde o esquema de representação está presente (de forma explícita ou implícita), é auto-descritivo. Os esquemas são definidos após a existência dos dados, não há um padrão e apenas uma parte dos dados pode apresentar estrutura definida. Exemplo: XML, RDF, OWL

Os dados não-estruturados não possuem estrutura definida, podem ser textos, documentos, imagens, vídeos, etc. Não há estrutura descrita nem implicitamente. Grande parte dos dados da WEB se enquadram nesse tipo.

2.3. MongoDB

MongoDB (do inglês hu**mongo**us, "gigantesco") é uma aplicação de código aberto, de alta performance, sem esquemas, orientado a documentos. Foi escrito na linguagem de programação C++[1]. Além de orientado a documentos, é formado por um conjunto de documentos JSON. Muitas aplicações podem, dessa forma, modelar informações de modo muito mais natural, pois os dados podem ser aninhados em hierarquias complexas e continuar a ser indexáveis e fáceis de buscar.

O desenvolvimento de MongoDB começou em outubro de 2007 pela 10gen. A primeira versão pública foi lançada em fevereiro de 2009.[2]

2.4. Map-Reduce

Map Reduce é um paradigma da Google para promover o uso de clusters, permitindo a paralelização de tarefas. A primeira etapa consiste em mapear (map) os dados e deixá-los em memória. Assim que tudo tenha sido devidamente mapeado, a segunda etapa é iniciada, onde os dados são unificados (reduce)[9].

Cada algoritmo de Map-Reduce pode realizar diferentes ações, por exemplo: o Map pode mapear dados, atribuir um valor para determinado dado. Já o reduce irá agrupar todos os dados mapeados, por exemplo, somando o valor atribuído agrupando por alguma chave definida.

2.5. NodeJS

NodeJS é uma aplicação orientada a eventos assíncronos, construída utilizando o Chrome V8, que é uma ferramenta de alta performance de JavaScript escrita em C++. O NodeJS foi desenhado para aplicações escaláveis, suportando múltiplas conexões concorrência[3].

O NodeJS conta com diversas bibliotecas em seu ecossistema NPM, fazendo com que haja suporte a API do Twitter, conexão com diversos bancos de dados, incluindo o MongoDB, escrita e leitura de arquivos, otimização em clusters e muitos mais. É fácil instalação e utilização.

3. TRABALHOS RELACIONADOS

A quantidade de informação gerada diariamente em redes sociais abriu espaço para diversos trabalhos que realizam a sua análise, desde a polaridade dos textos postado até sistemas mais complexo de previsão de trânsito e criminalidade. Nesta seção discutiremos alguns dos trabalhos relacionados a este trabalho.

Em 2014, [10] abordou a análise de sentimentos de *Tweets* sobre a Copa do Mundo no Brasil para comparar com as estatísticas divulgadas oficialmente. Os *tweets* coletados foram salvos no MongoDB e posteriormente, os textos passaram pela análise de sentimento, identificando a polaridade dos *tweets* em neutro, positivo e negativo. Foi possível identificar a reação dos visitantes e dos nativos durante a copa e os resultados foram considerados confiáveis.

Também em 2014, [8] propôs uma ferramenta que pesquisadores poderiam coletar e extrair conhecimento de *tweets*. Foram utilizados dois estudos de casos e os *tweets* coletados foram salvos em MongoDB. Foram analisados *tweets* em espanhol que abordavam dois assuntos: 1) Ataque Terrorista em Boston 2)Atividade política regular. O estudo de caso 1 mostra que a geolocalização e timezone são importantes nesse cenário, uma vez que os países que utilizam o espanhol ou são europeus ou latino-americanos e não dividem a mesma timezone. Foi possível identificar o comportamento perante a um evento inesperado contudo, não foi possível verificar o sentimento dos *tweets*. Já em seu segundo caso, foi possível fazer a classificação de sentimento e foi notado que muitos dos *tweets* eram neutros, onde ou o texto não era entendível ou era apenas informativo, além disso, a proporção de negativo/positivo ficou bem desbalanceada. O autor acredita que com uma base de dados maior isso poderia ser corrigido.

4. METODOLOGIA

Para realizar o experimento proposto, foram seguidas as seguintes etapas:

4.1. Coleta de dados

Foi feito um script em NodeJS para coletar *tweets* em *real-time* que contivessem nomes de séries de TV e outras palavras relacionadas. Também foram filtrados *tweets* em português e a geolocalização do Brasil. Foram coletados *tweets* durante 1 semana em diversos horários. Esses *tweets* foram coletados e salvos diretamente no MongoDB sem tratamento prévio.

4.2. Tratamento de dados

Para realizar a análise, foram removidas stop-words, *tweets* relacionados ao campeonato brasileiro, em específico a queda do Internacional para a série B.

Então, as datas que foram salvas como texto foram convertidas devidamente para o tipo certo, data, no banco.

4.3. Análise dos dados

Foram feitas três análises, já pré-determinadas: os termos mais frequentes, o volume de *tweets* por dia e o volume de *tweets* por hora do dia.

4.4. Visualização dos resultados

Os resultados das análises foram analisados, verificando os termos frequentes e o volume identificando possíveis causas. Os resultados foram dispostos em gráficos para melhor visualização.

5. EXPERIMENTO

Dado o modelo descrito na seção anterior, segue o detalhamento de cada etapa realizada e seus resultados.

5.1. Coleta dos Dados

O tema série foi estabelecido pelo autor a fim de filtrar *tweets* sobre determinado assunto e posteriormente entender a reação as diversas séries da atualidade. As palavras definidas para a coleta são ou o nome da série ou a **hashtag** relacionada a ela. Também foram utilizados os nomes dos canais que disponibilizam estas séries.

A coleta iniciou no dia 12/12/2016 às 00:00 e terminou do dia 21/12/2016 ás 03:00. Foram coletados cerca de 700 mil *tweets* durante esse período, totalizando xxmb de dados, considerando todas as informações que o Twitter retorna (data, texto, retweets, metions, etc).

Foi desenvolvido no script 1 em NodeJS utilizando a biblioteca Twitter e MongoDb para coletar e salvar os *tweets*.

```
1  //importa a biblioteca do Twitter
2  var Twitter = require('twitter');
3  //define as configurações para o Twitter
4  var client = new Twitter({
5    consumer_key: 'VqSkPBVPQtSSSUeSd6F5Bztdu',
6    consumer_secret: 'RuLNhkJRhJjcYeVIV9GrinsqNL9DsI458KaGoaCjlUIS5jnCRM'
       ,
7    access_token_key: '229050848-oS40x5NbJpvRqrFk0o1iKiWb5zWxpcG2lCAy8Hk1
       ',
8    access_token_secret: 'tQVWwE0Z7GAEBv3TUEEohpzh15j6LDsanxam6HOTfC0Ml'
9  });
10 //instancia o objeto de conexão com o Mongo
11 var MongoClient = require('mongodb').MongoClient
12   , format = require('util').format;
13
14 //string com as palavras a serem consideradas.
15 var track = "netflix,got,twd,seriado,série,himym,how i meet your mother
     ,3porcento,gilmore,breakinbad,house,suits,";
16 track+= "black mirror,friends,vikings,gray,flash,arrow,gotham,the100,
     american horror history,blacklist,stranger things,crown,";
17 track+="htgawm,how to get away with murder,marvel,criminal minds, mr
     robot,vampire diaries,orange is the new black,";
18 track+="oitnb,homeland,the big bang theory,tbbt,luke cage,demolidor,
     dare devil,jessica jones,narcos,the originals,sense8,";
19 track+="bones,the fall,prision brake,gossip girls,csi,scandal,the good
     wife,dexter,house of cards, hoc,teen wolf";
20 track+="game of thrones,supernatural,pretty little liars,pll,westworld,
     blindspot,once upom a time,legends of tomorrow,shield";
21 //abre a conexão com a API do Twitter passando os filtro track (
     palavras), location e linguagem
22 var stream = client.stream('statuses/filter', {track:track ,location:'
     Brazil',language:'pt'});
23 //para cada retorno
24 stream.on('data', function(event) {
25   //abre conexão com o banco
26   MongoClient.connect('mongodb://127.0.0.1:27017/trabalho_pratico',
       function(err, db) {
```

```
27      //se erro ao conectar, sai
28      if(err){
29          throw err;
30      }
31      //salva o tweet
32      db.collection('tweets').save(event,{w: 1}, function(err, records) {
33        //se erro ao salvar, informa em tela
34        if (err) console.log("erro ao salvar tweet - " + event.text);
35        console.log("record added "+ records);
36        //fecha conexão
37        db.close();
38      });
39    });
40
41  });
42  //caso erro ao receber tweet, joga a exceção
43  stream.on('error', function(error) {
44    throw error;
45  });
```

Código Fonte 1. Coletor de tweets

5.2. Análise dos Dados

A análise dos dados foi divida em três partes, detalhadas nas próximas sub-seções.

5.2.1. Tratamento dos *tweets*

Foram coletados724251 *tweets* durante o período, resultando em 0.978GB de informação. Verificando os tweets coletados, foram encontrados temas divergentes do tema proposto, dentre eles, o rebaixamento do Internacional para a série B, sendo este um falso positivo. Ao final da remoção desses falsos positivos, restaram 680535 registros para análise. Foram também desconsideradas pontuações e stop-words durante a contagem de termos mais frequentes.

5.2.2. Termos mais frequentes

Para realizar esta análise, os textos dos *tweets* passaram pelo pré-tratamento para remover stop-words e pontuação. Os *tweets* de temas divergentes também foram eliminados da base.

Para realizar a contagem, foi desenvolvido o fonte 2 para o MongoDB com Map Reduce e os resultados exportados para outra collection(tabela) de termos.

Os termos foram salvos, considerando suas variações, erros de digitação ou ortografia, e então ordenados por sua frequência.

```
1  //define map
2  var map = function() {
3    //define stop words
4      var stopwords = "a, agora, ainda, alguém, algum, alguma, algumas,
           alguns, ampla, amplas, amplo, amplos, ante, antes, ao, aos, após,
           aquela, aquelas, aquele, aqueles, aquilo, as, até, através,
```

```
        cada, coisa, coisas, com, como, contra, contudo, da, daquele,
        daqueles, das, de, dela, delas, dele, deles, depois, dessa,
        dessas, desse, desses, desta, destas, deste, deste, destes,
        deve, devem, devendo, dever, deverá, deverão, deveria, deveriam
        , devia, deviam, disse, disso, disto, dito, diz, dizem, do, dos
        , e, é, ela, elas, ele, eles, em, enquanto, entre, era, essa,
        essas, esse, esses, esta, está, estamos, estão, estas, estava,
        estavam, estávamos, este, estes, estou, eu, fazendo, fazer,
        feita, feitas, feito, feitos, foi, for, foram, fosse, fossem,
        grande, grandes, há, isso, isto, já, la, lá, lhe, lhes, lo, mas
        , me, mesma, mesmas, mesmo, mesmos, meu, meus, minha, minhas,
        muita, muitas, muito, muitos, na, não, nas, nem, nenhum, nessa,
         nessas, nesta, nestas, ninguém, no, nos, nós, nossa, nossas,
        nosso, nossos, num, numa, nunca, o, os, ou, outra, outras,
        outro, outros, para, pra, pro,pela, pelas, pelo, pelos, pequena
        , pequenas, pequeno, pequenos, per, perante, pode, pude,
        podendo, poder, poderia, poderiam, podia, podiam, pois, por,
        porém, porque, posso, pouca, poucas, pouco, poucos, primeiro,
        primeiros, própria, próprias, próprio, próprios, quais, qual,
        quando, quanto, quantos, que, quem, são, se, seja, sejam, sem,
        sempre, sendo, será, serão, seu, seus, si, sido, só, sob, sobre
        , sua, suas, talvez, também, tampouco, te, tem, tendo, tenha,
        ter, teu, teus, ti, tido, tinha, tinham, toda, todas, todavia,
        todo, todos, tu, tua, tuas, tudo, última, últimas, último, ú
        ltimos, um, uma, umas, uns, vendo, ver, vez, vindo, vir, vos, v
        ós, rt".replace(" ","").split(',');
    var summary = this.text;
  //define função para verificar se a string não é uma stop word
    var contains = function(){
      for(var j=0; j<stopwords.length;j++)
        if (summary[i].trim() == stopwords[j].toLowerCase().trim())
          return true;
        return false;
    };
  //se o texto não é vazio
    if (summary) {
        //transforma para minúsculo e quebra a string por espaço
        summary = summary.toLowerCase().trim().split(" ");
        for (var i = summary.length - 1; i >= 0; i--) {
            // remove pontuação e espaços ao início e fim da string
            summary[i] = summary[i].replace(/[^\w\s]/gi, '').trim();
            if (summary[i] && !contains())  {      // confere se a
               palavra não é vazia e não faz parte das stop-words
              emit(summary[i], 1); // // salva o valor 1 para cada
                 palavra
            }
        }
    }
};
//define função reduce
var reduce = function( key, values ) {
    var count = 0;
  //para cada valor da chave, soma o value
    values.forEach(function(v) {
        count +=v;
    });
```

```
33      return count;
34  };
35  //varre a collectionde tweets solicitando o map e o reduce, sem filtro
        e salva a saída na collection termos
36  db.tweets.mapReduce(map,
37            reduce,
38            {query:{  }
39            ,out:"termos"}
40  );
41  //ler a collection volume_dia ordenando pela soma de forma descrescente
42  db.termos.find().sort({value : -1});
```

Código Fonte 2. Termos mais frequentes

Abaixo, na figura 1 mostram os 15 termos mais utilizados, desconsiderando as stop-words. Para a geração do gráfico foi utilizado um script 3 em Python para ler e gerar o gráfico.

```
1  # importando o pylab (da biblioteca matplotlib que instalamos)
2  import pylab
3  import pymongo
4
5  from pymongo import MongoClient
6  client = MongoClient()
7  client = MongoClient('mongodb://localhost:27017/')
8  db = client.trabalho_pratico
9  termos = db.termos
10 result = termos.find().sort("value", -1).limit(15)
11 palavras = []
12 valores = []
13 for doc in result:
14   palavras.append(doc['_id'])
15   valores.append(doc['value'])
16
17 cor = ('r', 'b', '#00FF33')
18 pylab.figure(1)
19 x = range(len(palavras))
20 pylab.xticks(x, palavras)
21 pylab.xlabel('Termos')
22 pylab.ylabel('Quantidade')
23 pylab.grid(True)
24 locs, labels = pylab.xticks()
25 pylab.setp(labels, rotation=45)
26 pylab.bar(x,valores,0.5,color=cor, yerr=5, align='center')
27
28 pylab.show()
```

Código Fonte 3. Script de geração de gráfico

5.2.3. Volume por dia

Para realizar esta análise, foi desconsiderada a hora do *tweet*, de forma que fosse possível identificar a quantidade gerada por dia.

O fonte 4 foi desenvolvido para MongoDB também utilizando Map Reduce para

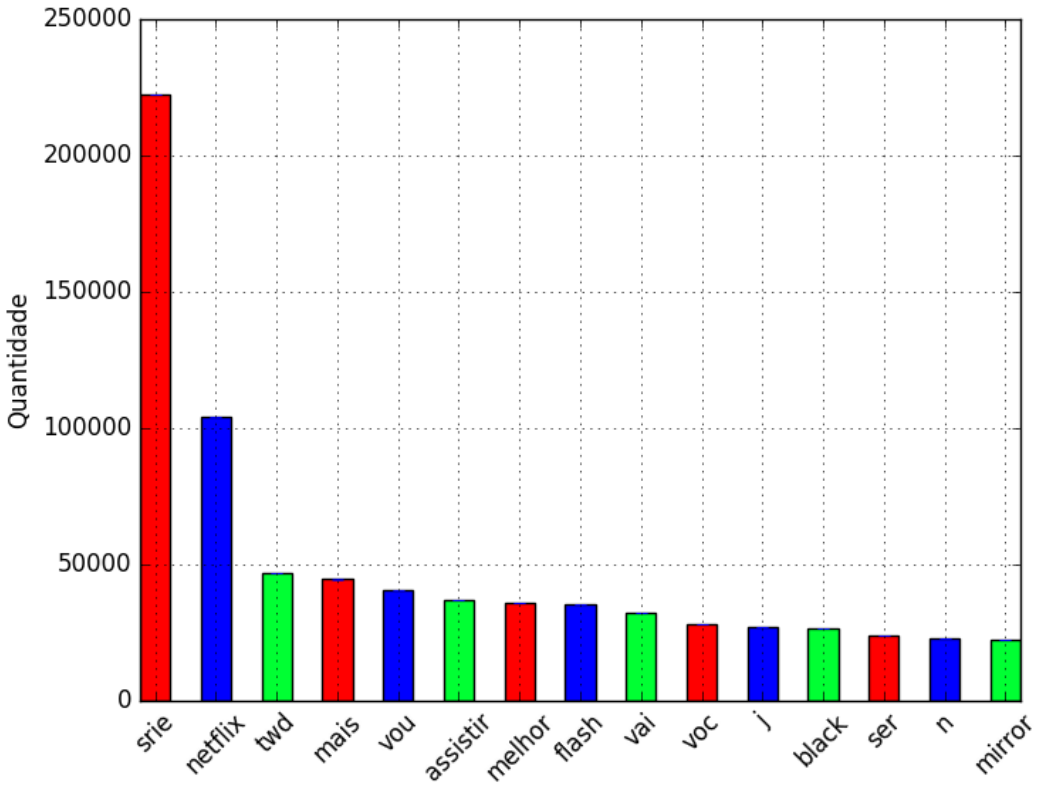

Figura 1. Os 15 termos mais frequentes

que a contagem fosse realizada e os resultados exportados para a uma nova collection contendo o dia e a quantidade de *tweets*/dia.

```
1  //define função map
2  var map = function() {
3    //utiliza a propriedade created_at
4    var summary = this.created_at;
5    //transformar a string data em Date
6    var montarData = function(){
7      //quebra a string por espaço
8      var data = summary.split(" ");
9      var mes;
10     // retorna uma data, montada no formato ano-mês-dia
11     return new Date(data[5]+"-"+getMonthFromString(data[1])+"-"+data
           [2]);
12
13   };
14   //retorno o mês de uma string (o mês de entrada pode ser por extenso)
         retornando o número
15   function getMonthFromString(mon){
16     return new Date(Date.parse(mon +" 1, 2012")).getMonth()+1;
17   }
18   if (summary) {
19     emit(montarData(), 1); // salva o valor 1 para cada data.
20   }
21
```

```
};
//define função reduce
var reduce = function( key, values ) {
  var count = 0;
  //para cada valor da chave, soma o value
  values.forEach(function(v) {
    count +=v;
  });
  return count;
};
//varre a collectionde tweets solicitando o map e o reduce, sem filtro
   e salva a saída na collection volume_dia
db.tweets.mapReduce(map,
          reduce,
          {query:{  }
          ,out:"volume_dia"}
 );
//ler a collection volume_dia ordenando pela soma de forma descrescente
db.volume_dia.find({},{value : -1})
```

Código Fonte 4. Volume de Tweets por dia

Abaixo, na figura 2 mostram os 15 dias que tiveram o maior volume de *tweets* coletados. Para a geração do gráfico foi utilizado um script 5 em Python para ler e gerar o gráfico.

```
# importando o pylab (da biblioteca matplotlib que instalamos)
import pylab
import pymongo

from pymongo import MongoClient
client = MongoClient()
client = MongoClient('mongodb://localhost:27017/')
db = client.trabalho_pratico
termos = db.volume_dia
result = termos.find().sort("value", -1).limit(15)
palavras = []
valores = []
for doc in result:
  palavras.append(doc['_id'].strftime("%Y-%m-%d"))
  valores.append(doc['value'])

cor = ('r', 'b', '#00FF33')
pylab.figure(1)
x = range(len(palavras))
pylab.xticks(x, palavras)
pylab.xlabel('Data')
pylab.ylabel('Quantidade')
pylab.grid(True)
locs, labels = pylab.xticks()
pylab.setp(labels, rotation=45)
pylab.bar(x,valores,0.5,color=cor, yerr=5, align='center')

pylab.show()
```

Código Fonte 5. Script de geração de gráfico

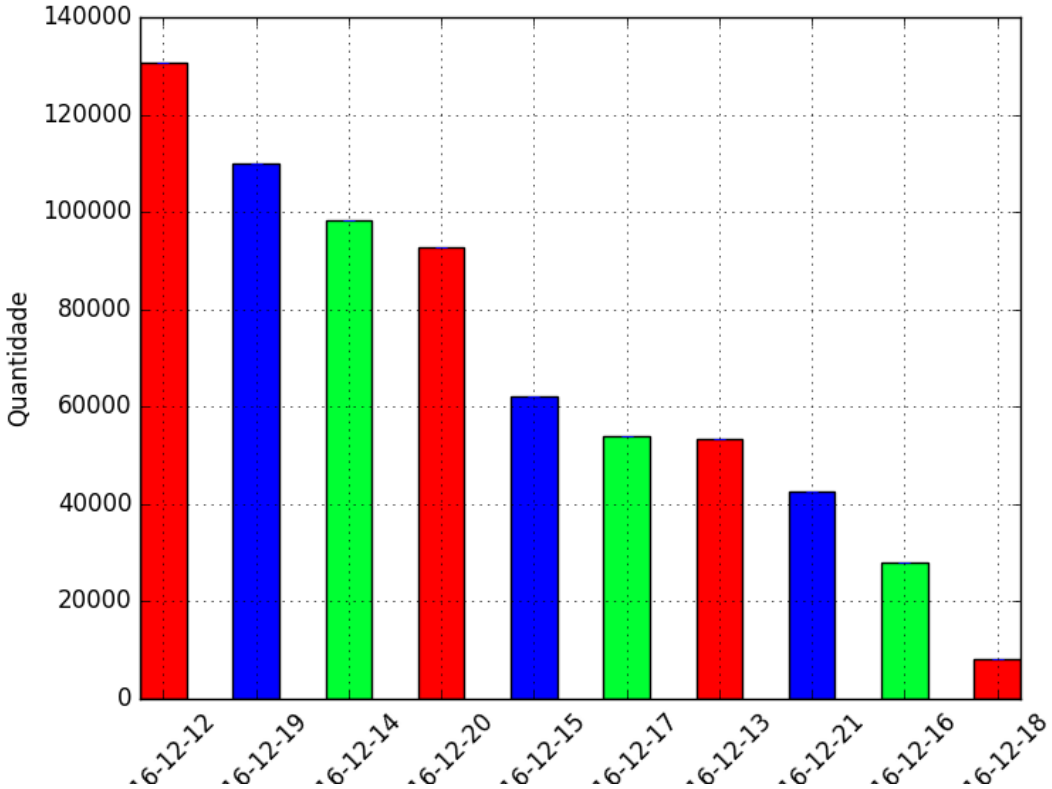

Figura 2. Os 15 dias com maior volume de *tweets*

5.2.4. Volume por hora do dia

Para realizar esta análise, foi desconsiderado o minuto e segundo do *tweet*, de forma que fosse possível identificar a quantidade gerada por hora-dia.

O script 6 foi desenvolvido para MongoDB também utilizando Map/Reduce para que a contagem fosse realizada e os resultados exportados para uma nova collection contendo o dia e a quantidade de *tweets*/hora/dia.

```
1  //define função map
2  var map = function() {
3    //utiliza a propriedade created_at
4    var summary = this.created_at;
5    //transformar a string data em Date
6    var montarData = function(){
7      //quebra a string por espaço
8      var data = summary.split(" ");
9      var mes;
10     // retorna uma data, montada no formato ano-mês-diaTHH:MI:SS
11     return new Date(data[5]+"-"+getMonthFromString(data[1])+"-"+data
           [2]+"T"+ data[3].split(":")[0] +":00:00Z");
12   };
13   //retorno o mês de uma string (o mês de entrada pode ser por extenso)
         retornando o número
14   function getMonthFromString(mon){
```

```
    return new Date(Date.parse(mon +" 1, 2012")).getMonth()+1;
  }
  if (summary) {
    emit(montarData(), 1); //salva o valor 1 para cada data.
  }
};
//define função reduce
var reduce = function( key, values ) {
    var count = 0;
  //para cada valor da chave, soma o value
    values.forEach(function(v) {
        count +=v;
    });
    return count;
};
//varre a collectionde tweets solicitando o map e o reduce, sem filtro
   e salva a saída na collection volume_hora_dia
db.tweets.mapReduce(map,
          reduce,
          {query:{  }
          ,out:"volume_hora_dia"}
);
//ler a collection volume_hora_dia ordenando pela soma de forma
   descrescente
db.volume_hora_dia.find({},{value : -1});
```

Código Fonte 6. Volume de tweets por hora no dia

Abaixo, na figura 3 mostram o top 15 hora/dia que tiveram o maior volume de *tweets* coletados. Para a geração do gráfico foi utilizado um script 7 em Python para ler e gerar o gráfico.

```
# importando o pylab (da biblioteca matplotlib que instalamos)
import pylab
import pymongo

from pymongo import MongoClient
client = MongoClient()
client = MongoClient('mongodb://localhost:27017/')
db = client.trabalho_pratico
termos = db.volume_hora_dia
result = termos.find().sort("value", -1).limit(15)
palavras = []
valores = []
for doc in result:
  palavras.append(doc['_id'].strftime("%m-%d %Hh"))
  valores.append(doc['value'])

cor = ('r', 'b', '#00FF33')
pylab.figure(1)
x = range(len(palavras))
pylab.xticks(x, palavras)
pylab.xlabel('Data/Hora')
pylab.ylabel('Quantidade')
pylab.grid(True)
locs, labels = pylab.xticks()
pylab.setp(labels, rotation=45)
```

```
26 pylab.bar(x,valores,0.5,color=cor, yerr=5, align='center')
27
28 pylab.show()
```

Código Fonte 7. Script de geração de gráfico

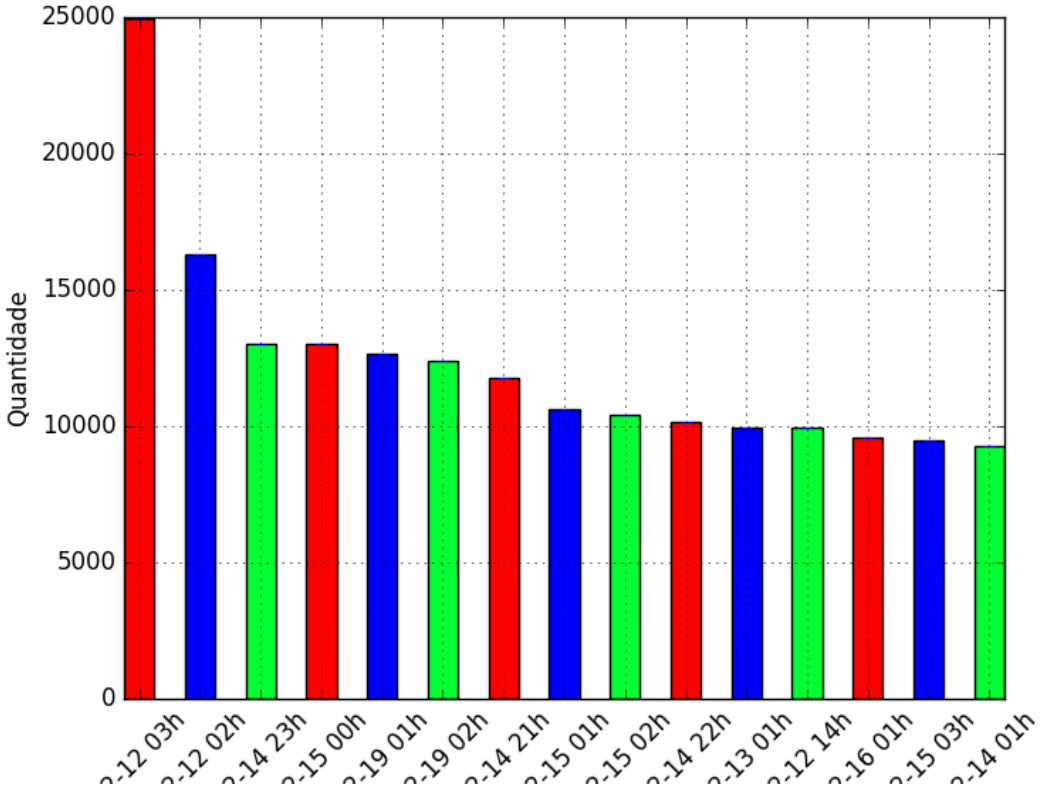

Figura 3. Os 15 hora/dia com maior volume de *tweets*

5.3. Visualização dos Resultados

Foi possível identificar a que algumas séries foram bastante comentadas, por exemplo: The Walking Dead, foi possível identificar isso pois a *hashtag* "twd"apareceu como o terceiro termo mais comentado. Isso se deve ao fato de na semana de coleta de *tweets* foi ao ar o último episódio da temporada. Também apareceram "Flash"e "Black Mirror"entre os 15 termos mais utilizados.

Também é possível identificar o comportamento do Twitter em relação ao horário e dias que possuem mais postagens em relação a séries de TV. O volume se destaca próximo ao final de semana (12/12 durante a madrugada, sendo que dia 12 seria segunda-feira). Dia 11/12 também ocorreu uma premiação referente as séries do ano, gerando uma maior participação dos usuários do Twitter.

Os usuários também costumam realizar postagens no período da noite, tendo poucos picos durante o dia. Isso pode estar relacionado a jornada de trabalho ou estudo das pessoas.

6. CONCLUSÃO

A utilização do Mongo para armazenagem de dados coletados do Twitter foi extremamente fácil e intuitiva. Não houve problemas com latência, os *tweets* foram salvos com facilidade. O tratamento dos dados também não foi difícil, sendo fácil identificar falsos positivos e removê-los. Também houve grande aprendizado, onde utilizou-se de diversas tecnologias para a realização do mesmo.

Para trabalhos futuros, seria interessante a análise de sentimentos em relação as séries, afim de entender a reação do público em relação a episódios novos ou a lançamento de séries. Também seria interessante, realizar a análise das séries mais comentadas, utilizando os termos mapeados para coleta como parâmetro de filtragem.

Referências

[1] MongoDB - Big Data Explained. https://www.mongodb.com/big-data-explained, 2016. [Online; accessed 20-Dezembro-2016].

[2] MongoDB - Blog. https://www.mongodb.com/blog/post/state-of-mongodb-march-2010, 2016. [Online; accessed 20-Dezembro-2016].

[3] NodeJS - About. https://nodejs.org/en/about/, 2016. [Online; accessed 21-Dezembro-2016].

[4] Nodejs.org. https://nodejs.org/, 2016. [Online; accessed 20-Dezembro-2016].

[5] Tableau - Oito principais tendências de Big Data para 2016. http://www.tableau.com/pt-br/asset/top-8-trends-big-data-2016?utm_campaign=Prospecting-BGDATA-ALL-ALL&utm_medium=Paid+Search&utm_source=Google+Search&utm_language=PT&utm_country=BRA&kw=big%20data&adgroup=CTX-Big+Data-Big+Data+All-PT-P&adused=120535072681&matchtype=p&placement=&kcid=b14a50b0-a1cb-4a36-957e-cd8c01ed1679&gclid=CKqYtorUgtECFQQIkQodoXwIGQ#form, 2016. [Online; accessed 20-Dezembro-2016].

[6] Twitter - Sobre. https://about.twitter.com/pt/company, 2016. [Online; accessed 20-Dezembro-2016].

[7] Twitter - Suporte. https://support.twitter.com/articles/262253, 2016. [Online; accessed 20-Dezembro-2016].

[8] David F ÁlvaroCuesta and María D R-Moreno. A framework for massive twitter data extraction and analysis. *Malaysian Journal of Computer Science*, 27:1, 2014.

[9] Cheng Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary Bradski, Andrew Y Ng, and Kunle Olukotun. Map-reduce for machine learning on multicore. *Advances in neural information processing systems*, 19:281, 2007.

[10] Mickael RC Figueredo, Nélio Cacho, and Carlos A Prolo. Redes sociais como fonte de informaç ao para cidades inteligentes.

[11] Steve Lohr. The age of big data. *New York Times*, 11, 2012.

[12] Andrew McAfee, Erik Brynjolfsson, Thomas H Davenport, DJ Patil, and Dominic Barton. Big data. *The management revolution. Harvard Bus Rev*, 90(10):61–67, 2012.