A System to Capture and Generation of Traffic Information from Posted Messages on Social Networks

Elisa H. M. Huzita, Tainan G. F. de Souza, Yan H. Kabuki
Departamento de Informática
Universidade Estadual de Maringá (UEM)
Maringá – PR – Brazil
email: emhuzita@din.uem.br, taigfs@gmail.com, yankabuki@gmail.com

***Abstract*— The expansion of the Internet and the growing use of social networks, allowed the capture and use of data related to traffic. This article presents an application, based in an intelligent system that explores the cooperativity of Facebook users so that, by data mining, get a navigable map that points out the most recent transit events reported by users. Thus, it is expected to contribute with useful information that helps any user to take decisions about the best path to be performed for his locomotion, avoiding, for example, traffic jams or accidents that are blocking roads they normally use.**

Keywords- social network; data mining; data extraction; text classification; collaboration

I. INTRODUÇÃO

Na sociedade atual os sistemas de transporte são extremamente importantes para todos que precisam locomover-se para atuar em suas atividades. Sabe-se, que eventos como congestionamentos, acidentes e até alagamentos fazem parte da rotina de centros urbanos, trazendo prejuízos aos usuários.

A quantidade de pessoas com acesso à internet tem crescido muito. No Brasil, segundo dados do IBOPE Nielsen Online [1], o número de acessos cresceu 20% no período entre 6/2009 e 6/2011, permitindo ser acessada pela maioria das classes sociais brasileiras, incentivando o crescimento das redes sociais.

As redes sociais são estruturas compostas por pessoas que se relacionam, compartilhando não só ideias e objetivos semelhantes, mas também problemas do cotidiano. Neste ponto, pode-se utilizar as observações postadas nas redes sociais, para melhorar, de alguma forma, a vida das pessoas. Essa melhoria pode acontecer de inúmeras maneiras. Assim, espera-se que os dados capturados, relacionados ao trânsito e devidamente tratados possam prevenir o usuário de um possível congestionamento ou acidente, que traria atrasos e incômodo.

Logo, a colaboração entre usuários torna-se essencial, de modo que a comunidade mantenha-se informada com as últimas ocorrências de trânsito de sua cidade, e ainda, que a informação venha da própria população, que está nas ruas. Portanto, quanto maior for a adesão dos usuários, mais estes se beneficiarão com informações atuais e confiáveis.

Os três sites de relacionamentos mais acessados no país - *Facebook*, Orkut e *Twitter* - possuem informações relevantes a serem capturadas.

O *Facebook* desponta como a rede que mais cresce e tem o maior número de usuários. Conta, atualmente, com 750 milhões de usuários ativos, sendo que 250 milhões acessam o site também de seus celulares, segundo estatísticas [2]. Um desafio é a política de privacidade do *Facebook* que, ao contrário de alguns concorrentes, como o *Twitter*, restringe o acesso e utilização das informações publicadas. Tanto os dados apresentados, quanto os desafios, foram motivações para a escolha do *Facebook* como objeto de estudo.

O foco deste trabalho é apresentar o aplicativo EPITrans, que tem como objetivo capturar mensagens de usuários da rede social *Facebook* e extrair informações importantes delas, apontando ocorrências diversas sobre o trânsito, como engarrafamentos, acidentes e enchentes.

Este trabalho tem a seguinte estrutura: na seção 2, os trabalhos relacionados; a seção 3 descreve o aplicativo e as tecnologias utilizadas; a seção 4 os testes; a seção 5 alternativas para otimização; e a seção 6 as conclusões.

II. TRABALHOS RELACIONADOS

A importância e relevância de dados provenientes das redes sociais assim como a mineração de dados, têm chamado a atenção de pesquisadores. Esses assuntos englobam várias áreas, como aprendizagem de máquina, processamento de linguagem natural e recuperação de informação. Os trabalhos a seguir foram utilizados para o nosso estudo.

Phelan, McCarthy e Smith [3] apresentam uma técnica que usa o *Twitter*, para recomendação de tópicos de notícias em tempo real, utilizando as mensagens dos usuários para obter as notícias mais comentadas. O *Twitter* também é a fonte de mensagens usada por Pak e Paroubek [4], que as utiliza para mineração de opinião e análise de sentimentos sobre determinados assuntos como, por exemplo, o quão favoráveis as pessoas são sobre determinados produtos. Para tanto, a partir dos dados iniciais, grupos de dados são separados, são removidas as *stopwords* e o classificador realiza o processo de treinamento, para então, classificar novos documentos. Esta é também a ideia utilizada no aplicativo deste trabalho.

Gonçalves e Quaresma [5] apresentam uma abordagem para a classificação de documentos jurídicos em língua portuguesa. No trabalho apresentado, para evitar a classificação manual de documentos por especialistas, os autores utilizam o método de Máquina de Vetores de Suporte para essa classificação, obtendo um bom resultado.

978-0-7695-4890-6/12 $26.00 © 2012 IEEE
DOI 10.1109/SBSC.2012.25

Camargo [6] apresenta os principais métodos de aprendizagem de máquina, fazendo uma comparação prática entre dois deles e utilizando exemplos em português. Entre os principais métodos analisados estão o de Árvore de Decisão, Naive Bayes, Redes Neurais, Máquinas de Vetor de Suporte e Clusterização. Ao comparar a classificação Bayesiana e Máquinas de Vetor de Suporte, Camargo conclui que o desenvolvimento de ferramentas de classificação automática de textos deve ser baseado no Classificador Naive Bayes.

O trabalho de Etzioni, *et al.* [7] apresenta técnicas para a extração de entidades. Duas delas podem ser utilizadas na abordagem do aplicativo deste trabalho. O *Pattern Learning* aprende regras e padrões, para então extrair instâncias e validá-las. Já a técnica *List Extraction* poderia ser utilizada com o auxílio de uma lista de locais, pré-definida, para determinada localidade.

Percebe-se que os trabalhos correlatos abrangem apenas partes isoladas do que é proposto neste trabalho, além de abordarem os problemas de forma diferente.

Primeiramente, a abordagem utilizada propõe que o volume de informações extraídas da rede social seja drasticamente reduzido, com uma pré-seleção, explicada na próxima seção.

Outro ponto importante é que a classificação, por si só, não soluciona o problema proposto. Dada uma mensagem alertando sobre o trânsito, é preciso conhecer não só o tipo da ocorrência, obtido com a classificação, mas principalmente o local em que ela aconteceu. Por isso, métodos de extração de entidade têm de ser aplicados junto à classificação.

Por último, esta abordagem permite que, a partir da cooperatividade dos usuários, novas mensagens sejam processadas em tempo real, devolvendo à comunidade as ocorrências mais recentes relatadas na rede social.

III. EPITRANS

A. *Facebook*

A escolha do *Facebook* como base para o desenvolvimento do aplicativo apresentado neste trabalho se deu por alguns fatores.

Primeiro por ser uma popular rede social e de grande alcance global, além disso, essa plataforma é interessante pelo grande volume de mensagens enviadas por dia, sendo utilizada por pessoas de muitas regiões distintas. A ideia é explorar os dados que estas mensagens carregam e gerar informações úteis, tanto para turistas, quanto para os moradores locais.

A rede social *Facebook* apresenta restrições em sua política de privacidade, de maneira que cada usuário decide a quem é permitida a visualização de suas mensagens, constituindo um desafio para a implementação de aplicativos de mineração de dados.

Outro desafio era definir a qual região uma ocorrência pertencia. Se uma determinada postagem, que relatasse um engarrafamento, por exemplo, se referisse à Avenida Tiradentes, sem informações adicionais, seria impossível saber se tal engarrafamento estaria acontecendo em Guarulhos, São Paulo, Contagem, Londrina ou Maringá, cidades que possuem uma avenida com tal nome.

Por último, o imenso número de postagens feitas nas redes sociais, com temas diversos, tornaria o seu processamento inviável. Por esse motivo, é necessário um meio de filtrar as mensagens que possivelmente sejam interessantes para o contexto deste trabalho.

Utilizando a ferramenta *Grupo* do *Facebook*, encontramos maneiras simples de lidar com essas questões. Um *Grupo* tem um título e permite que usuários interessados se inscrevam e convidem amigos para participar. Postagens efetuadas dentro da ferramenta são visíveis para todos os membros inscritos.

Para esta abordagem, cria-se um grupo para cada cidade interessada, eliminando a ambigüidade em relação aos locais. O usuário interessado teria de solicitar a participação no *Grupo*, concordando que suas postagens na ferramenta passem a ser visíveis para todos os membros. Com essa abordagem, a quantidade global de mensagens cairia drasticamente, e a importância das informações cresceria, dado que a maioria das postagens teria relação com o contexto do *Grupo*.

Tais fatores motivaram a escolha do *Facebook* como base para o desenvolvimento do aplicativo apresentado neste trabalho.

B. *Método de Classificação*

A princípio, buscou-se analisar técnicas de Mineração de Dados (*Data Mining*) para encontrar aquela que fosse mais adequada ao nosso problema: a partir de mensagens não-estruturadas em linguagem natural em português, selecionar aquela que pertença a uma dada categoria e, extrair destas, os dados úteis. É o problema de classificação de documentos.

Dentre os principais métodos de aprendizagem e classificação, estudou-se as Árvores de Decisão como ID3 [8], utilização da classificação de Naive Bayes [9] e as Redes Neurais com percéptrons multi-camadas [10].

Na avaliação das técnicas de classificação foi utilizada, para testes, a ferramenta *RapidMiner* [11]. Uma pequena base de dados para treinamento foi criada, apresentando três classes: Acidente, Trânsito Bom e Trânsito Lento, contando com 51, 67 e 116 mensagens, respectivamente.

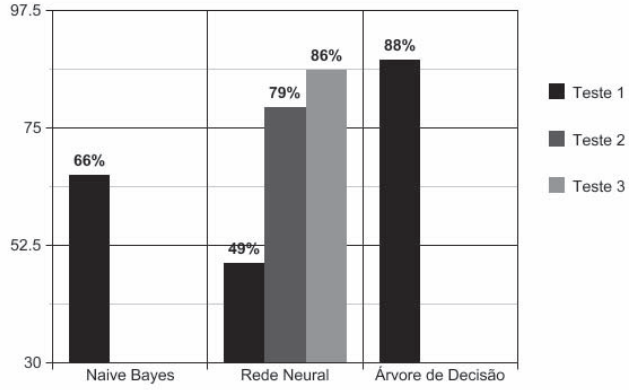

Figure 1. Testes com Métodos de Classificação

A Figura 1 apresenta a precisão de cada método. Para um conjunto de dados extremamente pequeno, quando comparado com situações reais do problema proposto, os tempos de execução foram decisivos. O método de Árvore de Decisão obteve a melhor precisão (88%) após cerca de 2 minutos de processo. Repetiu-se a aplicação do método de Rede Neural, aumentando o número de ciclos em cada teste. Para o Teste 1 com 5 ciclos, o processo levou cerca de 2 minutos, porém apresentou precisão baixa. Nos testes seguintes, com 25 e 50 ciclos, apesar da notável melhora na precisão, os tempos aumentaram para 41 e 60 minutos. O método de Naive Bayes apresentou uma precisão relativamente baixa (66%), porém seu tempo de execução foi de 2 segundos.

Naive Bayes difere dos outros métodos ao desconsiderar dependência entre os termos pertencentes ao documento de entrada. Outra suposição do método é de que a ocorrência de um termo no documento é independente de sua posição no mesmo. O método Bayesiano, ao ignorar a dependência entre termos e posições, ganha em simplicidade, mas também em velocidade, pois, tratando-se de um espaço de termos muito grande (palavras possíveis), deixa de fazer uma série de operações, sem perder muito em desempenho. Mitchell [12] apresenta a utilização do método de Bayes com resultados satisfatórios, alcançando uma precisão de 89%, e afirma que este é um dos métodos mais utilizados para a tarefa de categorização de documentos.

Optou-se então, pela utilização desse método por alguns motivos. O baixo tempo de execução foi determinante, pois a base de dados será constantemente incrementada e, dado que o aplicativo deverá funcionar online, novos treinamentos serão frequentes. Outra causa está no pequeno conjunto de dados de treinamento utilizados. As pouco mais de 200 mensagens utilizadas não formam uma base de dados sólida para que, dado um novo exemplo, o classificador possa determinar com precisão a qual classe o exemplo pertence. A tendência é que, com mais exemplos no *training set*, a precisão de cada classificador, incluindo o Bayesiano, melhore.

C. Tecnologias Utilizadas

O EPITrans foi desenvolvido na linguagem Java com o ambiente de programação *NetBeans* IDE 7.0.1. Para a coleta de dados do *Facebook*, foi usada uma estrutura interna do site, o *Graph* API. Após o processamento, cada documento é enviado para um banco de dados *MySQL*. Por fim, para mostrar os resultados obtidos ao usuário foi utilizada a API do *Google Maps* V2 em uma interface web.

D. Mecanismo de Mineração e Extração

O Mecanismo de Mineração e Extração trata-se da parte principal da aplicação. É o responsável por extrair informação de mensagens "cruas" e desestruturadas que recebe como entrada. Nesta seção será explicado o funcionamento do Mecanismo de Mineração e Extração que, daqui em diante, será tratado apenas como "Mecanismo".

Foram capturadas cerca de 30 mil mensagens relacionadas ao trânsito, incluindo mensagens irrelevantes para o domínio da aplicação. Destas mensagens, 900 foram utilizadas como Dados de Treinamento e outras 2040 foram utilizadas como Dados de Teste.

O Mecanismo consiste em fases. A Seleção por Interesse é a primeira em razão de sua importância. Se uma mensagem relacionada à condição do trânsito não possui o local determinado, não é possível utilizá-la, pois não há condições de informar outros usuários de forma precisa. Assim, nessa fase são consideradas apenas as frases de entrada em que é possível extrair o local.

Na mineração de texto, com a entrada já selecionada, começa a fase de Pré-processamento do texto. As mensagens geralmente não obedecem às regras da língua portuguesa, podendo haver incoerências e caracteres desnecessários, que serão removidos nessa fase. Foi utilizado o processo de *stemming* [13], para reduzir as palavras à forma de seus radicais, de modo que palavras de mesmo significado sejam agrupadas. O processo também remove as *stopwords*, termos que não são interessantes para o contexto de classificação de documentos. São removidos caracteres especiais como '#', '@', '%' ou '!', além de acentos em vogais.

O documento simplificado é transformado na forma de *Bag of words,* um modelo de representação de texto onde cada frase é representada por um conjunto desordenado de termos [14]. Na forma de *Bag of words,* o documento é enviado ao classificador, que aponta a classe mais compatível. O exemplo "O trânsito está congestionado hoje na Avenida Brasil, está tudo parado!" seria compatível com a classe "trânsito lento".

Para definir o tema da mensagem, é utilizado então, o classificador de Naive Bayes. O classificador é um algoritmo de aprendizado de máquina. Nele, cada classe representa uma categoria de mensagens já definida.

Segundo McCallum e Nigam [9], o classificador assume que os dados textuais foram gerados por um modelo paramétrico, e usa um conjunto de dados para calcular suas estimativas entre os termos do modelo. O conjunto de dados são pares, mensagem e categoria, usados para gerar uma base de conhecimento utilizada pelo classificador. Então, de posse dessas estimativas, ele classifica os novos documentos de teste usando a regra de Bayes, calculando a probabilidade de uma classe ter gerado o documento de teste. Após efetuar esse cálculo para todas as classes, o termo é classificado de acordo com a classe mais provável.

A partir dos 'dados de treino' fornecidos, o Mecanismo, ao receber uma mensagem de entrada, define sua classe. Por fim, o Mecanismo gera, como saída, o tipo de ocorrência, o local, o usuário remetente e o horário do processamento.

E. Funcionamento do EPITrans

A Figura 2 oferece uma visão geral do aplicativo: as mensagens (1), somente de usuários do *Facebook* e que aceitem utilizar o EPITrans, são coletadas do *Facebook*, processadas pelo Mecanismo (2) descrito na seção anterior, e as ocorrências mais atuais são exibidas ao usuário.

A princípio, configuramos o Mecanismo com três tipos de ocorrência, sendo elas "trânsito lento", "trânsito bom" e "acidente".

Como a condição do trânsito em um determinado local é algo dinâmico, uma mensagem sobre um acidente, pode não

ter utilidade alguma daqui a duas horas, assim como outra mensagem aconselhando que se opte por uma rua que está fluindo bem, pode ser enganosa se apresentada para os usuários fora de hora. Por esse motivo, o aplicativo permite a definição da validade de cada mensagem.

De posse das informações sobre o local e o tipo de ocorrência, pode-se então definir a duração da ocorrência em função da classe que ela representa. Ou seja, a validade de cada ocorrência pode ser alterada. Como padrão, foi definido que o tempo de utilização de uma mensagem é de trinta minutos.

Para que houvesse esse controle, a solução implementada foi enviar as mensagens para um banco de dados, constando em cada linha o ID da ocorrência, o ID do usuário que enviou a mensagem, a data e hora do registro, o local e o tipo da ocorrência. Porém, no caso de existir mais de uma mensagem para um mesmo local, dentro do tempo de validade, há uma possibilidade de que o tipo de ocorrência delas seja diferente. Para gerir tanto a validade das mensagens quanto o caso de ocorrências para o mesmo local, projetou-se o Gerenciador de Ocorrências.

O Gerenciador de Ocorrências (4) foi implementado em PHP e JavaScript, por serem linguagens compatíveis com a API dos mapas utilizados. A questão da validade é solucionada com uma seleção de mensagens no banco de dados, cujo horário está dentro do período de tempo definido anteriormente. Quanto à possibilidade de haverem diferentes tipos de ocorrência para o mesmo local, definiu-se que o tipo de maior freqüência, dentro da validade, será aquele mostrado para o usuário final.

Foi desenvolvido um aplicativo para dispositivos móveis, usando Android, tendo como objetivo disponibilizar para os usuários outro meio para visualizar e postar ocorrências de trânsito. O aplicativo conta com uma tela inicial de sincronização com o *Facebook*, além de quatro telas.

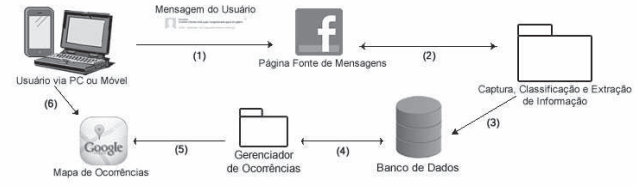

Figure 2. Diagrama de Funcionamento do Aplicativo EPITrans

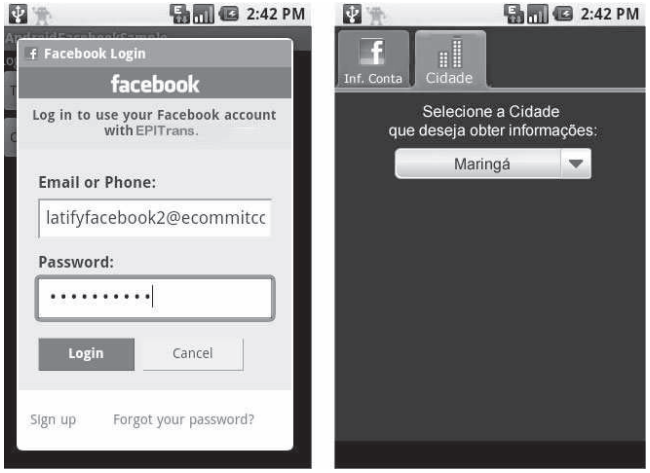

Figure 3. Telas de Interface do Aplicativo EPITrans, Login (esq.) e Configuração de Cidade (dir.)

Na Figura 3 são ilustradas as operações de sincronização com a rede social, à esquerda, tornando possível que qualquer mensagem enviada pelo usuário via dispositivo móvel, possa aparecer também no Grupo da cidade escolhida. Essa escolha acontece nas configurações do aplicativo, ilustrada à direita da Figura 3 acima. Desse modo, cada Grupo terá informações que se restringem a determinada cidade, evitando assim, qualquer tipo de conflito, como por exemplo, a existência de ruas com nomes idênticos em diferentes cidades.

Na Figura 4 são apresentadas duas telas do aplicativo Android. À esquerda da figura, a tela "Mapa" exibe o mapa da cidade pré-selecionada. Para a construção do mapa foi utilizado o *Google Maps*, juntamente com a API do *Google Maps* V2. A API permitiu que após a obtenção das ocorrências ainda válidas (5), fossem utilizados marcadores personalizados para caracterizar cada tipo de ocorrência.

Figure 4. Telas de Interface do Aplicativo EPITrans, Mapa (esq.) e Ocorrências (dir.)

Assim, para cada local em que há pelo menos uma mensagem válida, é inserido um marcador no mapa navegável, que oferece informações sobre o tipo de ocorrência, usuário remetente e o horário da atualização. Isso facilita a visualização da condição das vias da cidade.

Ainda na Figura 4, à direita, a tela "Ocorrências" exibe uma lista das ocorrências indicadas pelas mensagens dos usuários e mineradas pelo Mecanismo do aplicativo. Cada linha possui o ícone que corresponde à condição da via, o número de mensagens que relataram a ocorrência, o tipo de ocorrência e os respectivos local e horário.

Por fim, os mapas e ocorrências poderão ser acessados na página de *Facebook* e no aplicativo para dispositivos móveis (6) dos próprios usuários. Espera-se que de posse dessas informações, o usuário possa tomar melhores decisões em relação a qual percurso seguir, ao deslocar-se nas vias de trânsito da sua cidade.

IV. TESTES

Para a realização dos testes, foram utilizados os Dados de Teste anteriormente selecionados. As 2040 mensagens foram capturadas de duas páginas do *Facebook*, TransitoManaus e TransitoBelem, e uma comunidade, TransitoSalvador.

Na seleção por interesse são selecionadas apenas as entradas que contém o local da ocorrência, pois para o contexto do aplicativo apresentado, se a mensagem apresenta apenas as condições de uma via, sem mencionar o respectivo local, não é possível informar os outros usuários sobre onde o evento aconteceu. Por esse motivo, denominam-se as mensagens que não possuem essas informações como mensagens "lixo", já que não têm serventia para a abordagem deste trabalho. As mensagens "lixo" são descartadas. Por esses motivos, são necessários testes para verificar a confiabilidade dessa seleção.

Dentre os dados de teste, há mensagens que são interessantes para o contexto do trabalho, contendo informações realmente úteis para futuros usuários do aplicativo, mas também existem mensagens irrelevantes, como perguntas, propagandas e notícias que não seguem o modelo de ocorrência-local. Por isso, no primeiro teste, espera-se que o aplicativo possa definir como "lixo" as mensagens não interessantes para a extração, e envie as restantes para o processo de classificação.

Os testes para verificar a porcentagem de acerto na seleção de mensagens importantes foram realizados com todas as 2040 mensagens de teste. Cada mensagem foi inserida como entrada para o Mecanismo. Logo após, verificou-se manualmente se as saídas geradas pela Seleção de Interesse estavam corretas.

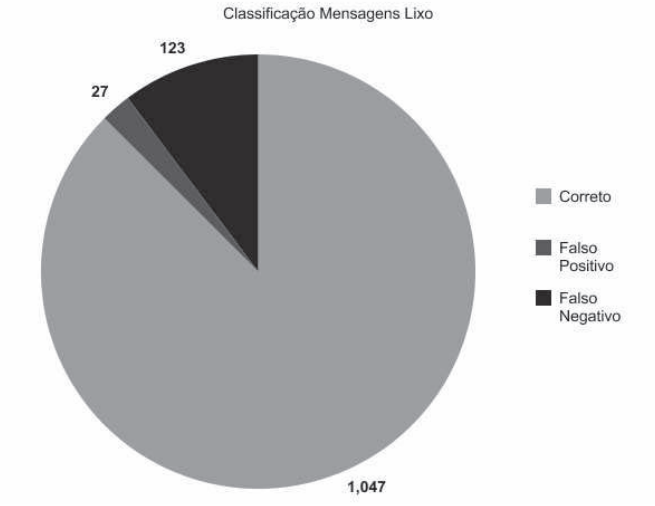

Figure 5. Gráfico dos Testes na Seleção por Interesse

Como resultado, observou-se que a Seleção por Interesse foi eficaz para 87% das mensagens, conforme a figura 5, definindo corretamente 1047 mensagens como lixo, porém, em cerca de 13% delas, mensagens lixo foram erroneamente enviadas para o processo de classificação ou mensagens interessantes foram consideradas lixo.

A próxima avaliação foi no ponto-chave do Mecanismo, a classificação da mensagem entre as classes. Calcular o desempenho dessa classificação se torna necessário, já que o aplicativo deve proporcionar a segurança de que a classe escolhida está, realmente, de acordo com a mensagem do usuário.

Para que os testes de classificação fossem realizados, todas as 2040 mensagens de teste foram classificadas manualmente, entre as três classes padrão (trânsito lento, bom e acidente). Então, cada mensagem foi dada como entrada para o Mecanismo, e o resultado da classificação do aplicativo foi comparado com o resultado da classificação manual.

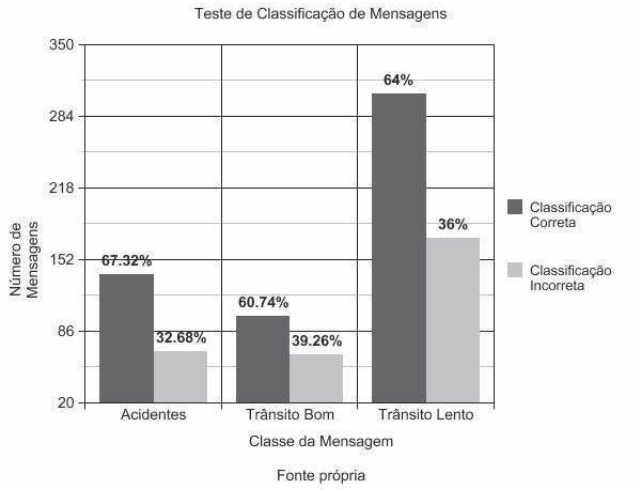

Figure 6. Gráfico dos Testes na Classificação de Mensagens

À medida que os testes foram sendo feitos, também foi realizada uma análise na classificação. Considerando o conjunto geral de mensagens, foi observado que aquelas simples, da forma ocorrência-local, tiveram taxa de acerto superior à média. Foi observado também, que os resultados, de acordo com a figura 6, foram semelhantes entre as classes, apresentando uma média de 64% de acerto.

De acordo com a análise, os principais equívocos na classificação foram o reduzido número de dados de treinamento, além dos erros existentes devido ao processo incorreto de extração do endereço da mensagem; muitas vezes o processo não conseguiu identificar o começo ou o fim do local, por não haver termos que sinalizassem-os (na, avenida, rua, estrada), e a mensagem era considerada lixo. Para minimizar esses erros, estão sendo analisados e testados métodos alternativos para essa extração, seção 5.

V. ALTERNATIVAS PARA OTIMIZAÇÃO

De acordo com Sarawagi [15], a partir de um layout com os campos de informações importantes a serem extraídas de determinado documento, um campo de extração de informação deve ser traçado a partir de cinco dimensões: o tipo da estrutura a ser extraída, o tipo da fonte desestruturada, o tipo de recurso de entrada disponível para extração, o método usado para a extração e a saída para a mesma.

Entidades são frases compostas por substantivos que ocupam poucos *tokens* no texto, como nomes de pessoas e organizações. Neste trabalho, os locais são considerados entidades. A fonte desestruturada é classificada na literatura como uma sentença, que apresenta muitos termos que não são parte de nenhuma entidade. O recurso de entrada trata-se do suporte para que a aplicação tome suas decisões, e pode vir a ser de uma base de dados de entidades conhecidas ou de textos desestruturados e rotulados manualmente (*label*).

O método usado para a extração pode ser *hand-coded*, no qual um especialista na língua define as regras que o extrator deve seguir, ou *learning-based*, um método baseado em aprendizado, em que é construída uma base de exemplos na qual as entidades são rotuladas, manualmente, por alguém que tenha um conhecimento do domínio para então as regras serem geradas por um algoritmo. Na ausência de um especialista, opta-se pelo método *learning-based*. Por fim, a saída deve ser formada apenas pelos *tokens* extraídos que representam o local. O próximo passo é definir regras de extração.

As regras podem ter formatos diferentes, desde expressões regulares como no *Whisk* [16], passando por expressões SQL e também itens e listas padrões, como no *Rapier* [17].

Regras para reconhecer uma entidade completa são constituídas por três partes, um padrão que corresponde ao texto que precede a entidade, outro que deve corresponder aos *tokens* da entidade, e um padrão que corresponde ao texto seguinte à entidade. A vantagem do modelo do *Rapier* é trazer, além de simples expressões regulares entre *tokens*, a opção de restrições também na classe semântica das palavras que o elemento pode corresponder.

(1) Rset = conjunto de regras, inicialmente vazio
(2) Enquanto existe uma entidade x ϵ D não contemplada por alguma regra em Rset
    (a) Forme novas rergas para x
    (b) Adicione as novas regras para Rset
(3) Poste regras do processo para podar regras redundantes

Figure 7. Algoritmo Rule Learning, extraído de Sarawagi (2008)

O algoritmo *Rule Learning* tem a função de gerar regras de extração, a partir de mensagens rotuladas com as entidades. O algoritmo da figura 7, apresentado por Sarawagi [15], busca nos documentos, entidades que ainda não foram contempladas por regras para, então, adicioná-las ao conjunto de regras e compará-las, removendo aquelas redundantes.

Essa aproximação pode ser utilizada no EPITrans, ao obter-se padrões que correspondem ao posicionamento das palavras que representam, na frase, o local. Pode-se também buscar por mais características do termo, de modo que estas evidenciem que a Seleção por Interesse está sendo realizada com maior precisão. Incluem-se dentre estas características informações tais como classe semântica e propriedades dos *tokens*.

Foram então, feitos testes com a abordagem de Sarawagi para identificar a posição do local na mensagem. Foi utilizado o algoritmo *Trigram tagger*, que utiliza um modelo probabilístico para calcular a chance de uma palavra ser um local a partir das palavras anteriores e seguintes [18]. Utilizando uma base de treinamento com 80 mensagens e 113 mensagens para teste (todas oriundas da base de dados maior, citada anteriormente), os testes apontam uma taxa de acerto de 88%. Assim, espera-se obter melhores resultados na extração, comparado com o processo utilizado atualmente no aplicativo.

Os próximos passos para que essas alternativas possam ser aplicadas no EPITrans, são o desenvolvimento de um analisador semântico para que se reconheça a classe semântica de determinado *token* e, assim essa propriedade também possa ser usada nas comparações. Está prevista a construção de uma base sólida de exemplos rotulados, com a utilização de *tags*, de modo que o algoritmo de *rule learning* possa gerar regras mais consistentes para a extração. Estas implementações estarão sendo realizadas e, ao término destas, serão realizados testes exaustivos para que se possa avaliar o ganho de desempenho na extração de informações a respeito de locais nas mensagens.

VI. CONCLUSÃO

Após o estudo necessário para o desenvolvimento do aplicativo, pôde-se perceber que redes sociais são uma fonte riquíssima de dados, que se lapidados de forma correta podem gerar informações úteis para muitas áreas da sociedade.

Porém, essa lapidação tem grandes desafios. Começando pela forma que as mensagens dos usuários de redes sociais apresentam. Frases desestruturadas, gírias e palavras "transformadas" são comuns. Isso torna o processo de obtenção da informação menos eficaz e mais demorado. Outro desafio é ajustar as informações de acordo com o seu

tempo útil, de modo que não seja informado ao usuário algo que não seja condizente com a condição real de uma via naquele momento. Além disso, tem-se como objetivo repassar a informação ao usuário de forma simples e clara, por meio de mapas, para que esse não tenha dificuldades de compreensão, mesmo que com pouco tempo disponível e em trânsito, por exemplo.

A contribuição deste trabalho está no uso de técnicas de mineração tais como a categorização de textos e a extração de entidades, atuando necessariamente em conjunto. Isto tornou possível gerar, em tempo real, informações essenciais, envolvendo, por exemplo, local e tipo de ocorrência.

Espera-se que tais informações mineradas sejam repassadas aos usuários, de modo que estes tenham mais dados para decidir sobre o melhor trajeto a ser percorrido, face a eventuais ocorrências no trânsito.

Na área escolhida como alvo deste trabalho, a de trânsito em cidades, a ideia cooperativa do aplicativo só funcionará se os próprios usuários, interessados em informações sobre as condições das pistas pelas quais trafegam, também enviarem informações de trânsito. Assim, com o auxílio do aplicativo, todos podem se beneficiar com informações atualizadas e confiáveis, via mensagem ou mapa.

AGRADECIMENTOS

Os autores agradecem ao CNPq (processo 560135/2010-6) e à RNP (projeto SIMTUR - edital CTIC) pelo apoio financeiro. Este trabalho é parcialmente apoiado pelo INES (Instituto Nacional de Ciência e Tecnologia para Engenharia de Software).

REFERÊNCIAS

[1] Pesquisa IBOPE Nielsen Online, http://www.ibope.com.br/calandraWeb/servlet/CalandraRedirect?temp=5&proj=PortalIBOPE&pub=T&db=caldb&comp=IBOPE+Nielsen+Online&docid=C2A2CAE41B62E75E83257907000EC04F, Setembro.

[2] Estatísticas Oficiais do Facebook, http://newsroom.fb.com/content/default.aspx?NewsAreaId=22, Julho, 2012.

[3] Smyth, B., Phelan, O. e McCarthy, K. 2009. "Using Twiter to recommend real-time topical news". Em http://irserver.ucd.ie/dspace/bitstream/10197/1893/1/sp145-phelan.pdf, Novembro.

[4] Pak, A. e Paroubek, P. 2010. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining". Em http://deepthoughtinc.com/wp-content/uploads/2011/01/Twitter-as-a-Corpus-for-Sentiment-Analysis-and-Opinion-Mining.pdf, Novembro.

[5] Goncalves, T e Quaresma, P. 2003. "A Preliminary Approach to the Multilabel Classification Problem of Portuguese Juridical Documents". Progress in Artificial Intelligence, vol. 2902/2003.

[6] Camargo, Y.B.L. de. 2007. "Abordagem lingüística na classificação de textos em português". Dissertação em Ciência da Computação, UFRJ. Em http://www.pee.ufrj.br/teses/textocompleto/2007062502.pdf.

[7] Etzioni, O., Caferella, M., Downey, D., Popescu, A., Shaked, T., Soderland, D., Weld, D.S. e Yates, A. 2005. "Unsupervised Named-Entity Extraction from the Web: An Experimental Study". Em Artificial Intelligence, 165(1):91–134.

[8] Quinlan, J.R. 1986. "Induction of decision trees". Machine Learning, volume 1, number 1, 81-106.doi:10.1007/BF00116251.

[9] McCallum, A. e Nigam, K. 1998. "A Comparison of Event Models for Naïve Bayes Text Classification". Em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.65.9324&rep=rep1&type=pdf.

[10] Beale, R. e Jackson, T. 1990. Neural Computing: An Introduction. 3 ed., London, Institute of Physics Publishing.

[11] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, andT. Euler. "Yale: Rapid prototyping for complex data mining tasks". Em L. Ungar, M. Craven, D. Gunopulos, and T. Eliassi-Rad, editors, KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 935–940,New York, NY, USA, August 2006. ACM.

[12] Mitchell, T. M., 1997, Machine Learning. USA, Ed. McGraw-Hill.

[13] Porter, M. 2001. "Snowball: a language for stemming algorithms". Em http://snowball.sourceforge.net.

[14] Harris, Z.S. 1954. Distributional Structure. Word, Vol 10, 146-162.

[15] Sarawagi, S. 2008. "Information extraction". Em Foundations and Trends in Databases, 1(3).doi:10.1561/1900000003.

[16] Soderland, S. 1999. "Learning information extraction rules for semi-structured and free text". Machine Learning, p. 34.

[17] Callif, M.E. 1998. "Relational learning techniques for natural language information extraction". Em http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.33.46&rep=rep1&type=pdf.

[18] Brants, T. (2000b). "TnT – A statistical part-of-speech tagger". In Proceedings of the Sixth Conference on Applied Natural Language Processing ANLP-2000. Seattle, WA. Schmid, H. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. Em http://www2.lirmm.fr/~lopez/Titrage_general/doc/nemlap94.ps.

[19] Biblioteca Open Sourcer Batchfb, http://code.google.com/p/batchfb/, Julho, 2012.

[20] Política de Privacidade do Twitter, http://twitter.com/privacy, Julho, 2012.