Classificação em Tempo Real de Dados em Fluxo Contínuo Perante Treinamento Incerto *

Alexandre Davis[1], **Adriano Veloso**[1]

[1]Departamento de Ciência da Computação - Universidade Federal de Minas Gerais
Belo Horizonte, MG

{agdavis,adrianov}@dcc.ufmg.br

Abstract. *The rapid expansion of social networks has caused an increasing demand for machine learning algorithms (e.g., classifiers) that can operate on real-time data flows, such as message streams. The rapidly changing vocabulary used in such streams makes it difficult to create adequate training sets, and keeping them up-to-date is an even greater challenge. In this paper, we propose a novel way to perform real-time classification on dynamic data streams. Our proposal involves the automatic generation of training sets using the* Expectation-Maximization *approach in a set of messages composed of positive and uncertain examples. In order to demonstrate that our solution is general, we present two applications scenarios that use the proposed real-time classifier to disambiguate named entities references and to perform sentiment analysis in data streams. An assessment of the efficiency of this technique is presented. The* Observatório da Web *project is currently running the proposed approach for real-time disambiguation in its operation pipeline.*

Resumo. *A expansão das redes sociais tem levado a uma demanda crescente por algoritmos de aprendizado de máquina (e.g., classificadores) que possam ser usados em tempo real sobre fluxos de dados, tais como fluxos de mensagens. A volatilidade do vocabulário usado nessas mensagens além de dificultar a criação de conjuntos de treinamento adequados, torna sua atualização um desafio ainda maior. Neste artigo, propomos uma nova estratégia para classificação em tempo real de fluxo de dados dinâmico. Nossa proposta envolve a geração automática de conjuntos de treinamento usando a abordagem* Expectation-Maximization *sobre um conjunto de mensagens composto de exemplos positivos e incertos. Para demonstrar que nossa solução é abrangente, apresentamos duas aplicações que usam o classificador proposto para desambiguar referências a entidades nomeadas e para executar análise de sentimento em fluxos de dados. Uma avaliação da eficiência de nossa proposta é apresentada. O projeto Observatório da Web está atualmente executando a técnica proposta para desambiguação em tempo real em seu ambiente de produção.*

1. Introdução

Uma quantidade crescente de pessoas participa de redes sociais, emitindo comentários e opiniões sobre os mais diversos assuntos da atualidade. Mensagens postadas em redes

*Este trabalho foi parcialmente financiado por CNPq, CAPES e Bolsa UOL Pesquisa.

sociais, como o Twitter e o Facebook, têm sido usadas como indicadores de tendências, opiniões e fontes de informação. No entanto, reconhecer e detectar essas tendências em meio ao intenso fluxo de mensagens não é uma tarefa fácil. Assim, extrair e analisar informações interessantes a partir de fluxos de mensagens tornou-se um tema de pesquisa importante. Entendemos que cada indivíduo que se manifesta em uma rede social se comporta como um sensor, de funcionamento intermitente, que produz mensagens destinadas à interpretação de outros humanos ao invés de medições. Essas mensagens nem sempre possuem estrutura sintática correta e são, tipicamente, muito curtas, inviabilizando estratégias baseadas em processamento de linguagem natural para sua interpretação. Além disso, esses fluxos são volumosos e contêm termos e expressões que mudam constantemente, dificultando o uso de estratégias de mineração de dados e aprendizado de máquina.

Nosso objetivo neste trabalho é buscar soluções para classificação em tempo real, utilizando como entrada mensagens extraídas de redes sociais (em especial, o Twitter), as quais são postadas em fluxo contínuo. Nossa proposta é geral o suficiente para apoiar a solução de problemas tais como análise de sentimento, detecção de tópicos e desambiguação de entidades nomeadas. Diversos desafios surgem quando exploramos este cenário em particular, entre eles: (i) o vocabulário das mensagens é extremamente dinâmico, o que rapidamente torna obsoleto o conjunto de treinamento necessário para a geração do classificador, e (ii) o classificador deve ser regerado continuamente, à medida que o fluxo de dados chega. Sendo assim, a construção manual do conjunto de treinamento é inviável. Além disso, o classificador deve ser baseado em estruturas de dados eficientes e incrementais, para que seja possível mantê-lo constantemente atualizado.

Para a discussão que segue, definimos um fluxo contínuo de mensagens como S. Tal fluxo de mensagens é composto por todas as mensagens postadas contendo pelo menos um termo do conjunto de palavras chave pré-definidas, que denotamos por K. Deseja-se rotular as mensagens de S como sendo positivas ou negativas, de acordo com o significado atribuído pela aplicação. Por exemplo, caso a aplicação de interesse para o classificador seja análise de sentimentos, então as mensagens positivas são aquelas que trazem opiniões positivas acerca de uma entidade considerada. Caso a aplicação de interesse seja desambiguação de entidades, então as mensagens positivas são aquelas que referenciam corretamente a entidade desejada. Dada a natureza das redes sociais e o padrão de postagem de mensagens, podemos supor a existência de conteúdo altamente correlacionado com mensagens positivas. Por exemplo, perfis de certos indivíduos são altamente correlacionados com certas entidades (e.g., o perfil @OficialfFlu é altamente relacionado com a entidade "Fluminense Football Club"). Analogamente, algumas *hashtags* são altamente relacionadas com o conteúdo postado sobre uma entidade (e.g., #DilmaJá denota claramente uma referência positiva à entidade em questão). No entanto, observamos que, em geral, a proporção de mensagens em S que podem ser rotuladas como positivas através desse artifício é ínfima em relação ao universo de todas as mensagens. As mensagens que não puderem ser rotuladas seguindo essa estratégia, serão alvo de nosso classificador. Denotamos o conjunto de mensagens em S rotuladas como positivas como sendo P (Positive), e o restante das mensagens como sendo U (Unlabeled).

Periodicamente, acumulamos mensagens em P e em U. Em seguida, executamos um método iterativo, chamado *Expectation-Maximization Entropy Minimization*, para extrair mensagens positivas que estejam no conjunto U. O método proposto, que será melhor

explicado nas proximas seções, define um corte individual para cada mensagem avaliada. Esse corte define se a mensagem deve ser considerada positiva ou não, através da minimização da entropia nos conjuntos P e U. Uma vez terminado esse procedimento, consideramos todas as mensagens não-rotuladas remanescentes como negativas. Finalmente, usamos todas as mensagens rotuladas como treinamento para nosso classificador, o qual é mantido atualizado incrementalmente. De forma a construir um sistema em produção baseado no método proposto, utilizamos um *pipeline* composto por três estágios (Figura 1). Esse pipeline recebe como entrada os conjuntos K e P e a saída é o conjunto de mensagens S, todas classificadas. Utilizando esse pipeline, avaliamos o potencial de nosso método de classificação em tempo real em aplicações de análise de sentimento e de desambiguação de entidades sobre mensagens coletadas do Twitter.

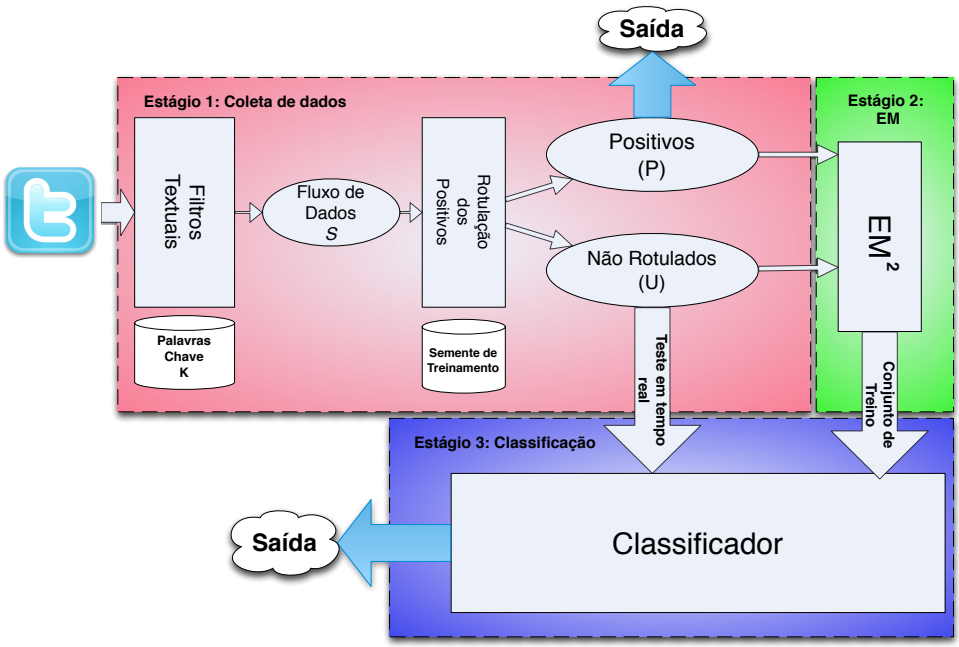

Figura 1. Representação do pipeline para classificação em tempo real.

2. Trabalhos Relacionados

Existe ampla pesquisa sobre classificação perante dados incertos. Incerteza pode aparecer por consequência de erros de medição [Kao et al. 2010], dados incompletos [Batista and Monard 2003], ou pela falta de dados negativos. Nesse último caso, temos um cenário conhecido como Aprendizado-*PU* [Liu et al. 2003, Denis 1998, Comité et al. 1999] (*positive and unlabeled learning*), no qual um classificador é construído a partir de um conjunto de exemplos positivos P juntamente com dados não-rotulados U (onde U pode ser composto tanto por exemplos positivos quanto negativos). Desta forma, este trabalho pode ser considerado como uma instanciação do Aprendizado-*PU*.

A maioria das abordagens para Aprendizado-*PU* utiliza uma estratégia de dois passos [Li et al. 2009, Elkan and Noto 2008]: (1) identificar um conjunto de exemplos negativos; (2) construir um classificador utilizando a abordagem *Expectation-Maximization* [Dempster et al. 1977, Liu et al. 2002] iterativamente. Alternativamente, técnicas de Recuperação de Informação, como Rocchio [Rocchio 1971], também

podem ser utilizadas para extrair exemplos negativos do conjunto de dados não-rotulados [Li and Liu 2003]. O método *Biased-SVM* [Liu et al. 2003] introduziu uma nova abordagem para o primeiro passo, onde pondera-se cada exemplo utilizando uma função de otimização linear.

Em alguns cenários, no entanto, o conjunto de exemplos positivos P pode ser demasiadamente pequeno, tornando também desejável extrair uma quantidade adicional de exemplos positivos dos dados não-rotulados U. Trabalhos recentes [Li et al. 2007, Fung et al. 2006] exploram a extração de exemplos positivos de forma a aumentar a acurácia de classificação. Um problema similar é a classificação de *classe-única* [Schölkopf et al. 2001]. Nesse caso, contudo, os dados não-rotulados são ignorados e informações importantes podem não ser utilizadas. Normalmente, essas propostas fazem a estimativa de densidade para encontrar uma região em U que contenha a maioria dos exemplos positivos disponíveis.

Percebemos que as propostas existentes para realizar Aprendizado-*PU* não escalam quando o volume de dados não-rotulados aumenta. Dessa forma, propomos um solução alternativa altamente incremental, que é capaz de processar dados em larga escala rapidamente. Além disso, adotamos uma estratégia inovadora, que explora informação de distribuição de dados em grão fino para melhorar a escolha de cortes usados para converter ponderações numéricas em uma classe prevista, a qual faz parte do processo de geração automática do conjunto de treinamento.

3. Classificação em Tempo Real de Dados em Fluxo Contínuo

Nesta seção apresentamos a formulação do problema de classificação em tempo real de dados em fluxo contínuo. Em seguida discutiremos nossa solução, que envolve a produção contínua e automática de treinamento, utilizando a abordagem *Expectation-Maximization*. Finalmente, descrevemos o método proposto, o qual chamamos de *Expectation-Maximization Entropy Minimization* (ou simplesmente EM^2), que encontra cortes individuais de forma a minimizar a entropia das partições P e U.

3.1. Formulação do Problema

Considerando um fluxo contínuo de mensagens postadas em redes sociais (e.g., Twitter), e considerando $\{k_1, k_2 \ldots, k_n\} \in K$, palavras chave que delimitam o fluxo S, nossa tarefa é monitorar S e prever se uma mensagem é positiva ou negativa de acordo com a aplicação sendo considerada. Esta tarefa pode ser modelada como um problema de aprendizado supervisionado. Neste caso, recebemos um conjunto de dados como entrada denominado *conjunto de treinamento* e denotado por $\mathcal{D}$. O conjunto de treinamento consiste de exemplos na forma $< m, c >$, onde $m \in S$ é uma mensagem contendo pelo menos um termo do conjunto de palavras chave K, e $c \in \{\oplus, \ominus\}$ é uma variável binária que informa se a mensagem é positiva ou negativa. O conjunto de treinamento é usado para produzir um classificador que relaciona padrões textuais em m com o valor de c. O *conjunto de teste* é denotado por $\mathcal{T}$, e consiste de registros na forma $< m, ? >$, para os quais apenas se sabe a mensagem $m \in S$. O classificador, portanto é usado para indicar quais mensagens em $\mathcal{T}$ são positivas e quais são negativas.

No entanto, métodos supervisionados estão sujeitos ao gargalo de aquisição de exemplos, uma vez que a criação de um conjunto de treinamento requer especialistas humanos para rotular as mensagens em $\mathcal{D}$ manualmente. O custo associado a esse processo

de rotulação é inviável em um ambiente de fluxo contínuo de dados, dada a dinamicidade do vocabulário e a rapidez do fluxo de dados. Contudo, na maioria dos casos, o custo de adquirir uma quantidade limitada de exemplos positivos é potencialmente desprezível. Mais especificamente, em ambientes de mídia social, podemos utilizar perfis ou *hashtags* previamente conhecidas por estarem altamente associadas a mensagens positivas, gerando assim um conjunto inicial de algumas mensagens positivas P e muitas mensagens não-rotuladas U (i.e., $|U| \gg |P|$). Nossa solução assume que a definição desses perfis e *hashtags* é a única supervisão necessária para a geração do conjunto de treinamento, e que o processo então pode ser repetido automaticamente daí em diante.

3.2. Abordagem *Expectation-Maximization*

Assumimos que vale a pena explorar os dados não-rotulados em U, considerando que o custo de obter esses dados é desprezível. O esforço computacional para realizar o melhoramento dos dados em U será recompensado com um ganho na acurácia da classificação. Dessa forma, para tirar vantagem de grandes volumes de dados não-rotulados, propomos um novo método baseado na abordagem *Expectation-Maximization* [Dempster et al. 1977] (ou simplesmente EM). Assumimos dois cenários:

- o conjunto de treinamento $\mathcal{D}$ contém dados incertos, ou seja, é composto por um pequeno conjunto de exemplos verdadeiramente positivos P e um grande conjunto de exemplos não-rotulados U.
- o conjunto de treinamento $\mathcal{D}$ contém dados incertos, ou seja, é composto de um pequeno conjunto de exemplos provavelmente positivos P e um grande conjunto de exemplos não-rotulados U.

Em ambos os cenários, exemplos não-rotulados são tratados inicialmente como sendo exemplos negativos, de modo que o classificador possa ser construído a partir de $\mathcal{D} = \{P \cup U\}$. No entanto, em ambos os cenários, $\mathcal{D}$ provavelmente contém falsos negativos. Além disso, no segundo cenário, $\mathcal{D}$ pode também conter falsos positivos. O método proposto utiliza um classificador que atribui a cada exemplo $x \in \mathcal{D}$ uma probabilidade $\alpha(x, \ominus)$, que denota a probabilidade de x ser um exemplo negativo. O método itera executando um sequência de operações de *transição de rótulo*, de forma que, ao fim do processo, é esperado que os rótulos convirjam para a configuração mais provável. No primeiro cenário, apenas a operação de transição $x^{\ominus \to \oplus}$ é permitida, enquanto no segundo cenário a operação $x^{\oplus \to \ominus}$ tambem é permitida. Em ambos os casos, uma questão crucial que afeta a eficácia de nosso método é decidir quando uma transição de rótulo deve ser realizada. Tipicamente, um corte α_{min} é utilizado, de forma que a operação $x^{\ominus \to \oplus}$ será sempre executada se x for negativo e $\alpha(x, \ominus) \leq \alpha_{min}$. Analogamente, a operação $x^{\oplus \to \ominus}$ é sempre executada se x for positivo e $\alpha(x, \ominus) > \alpha_{min}$.

Infelizmente, o valor ótimo para α_{min} não é conhecido previamente. Além disso, determinar um corte ideal de maneira empírica utilizando procedimentos baseados em validação cruzada não é uma opção viável, já que os dados disponíveis são incertos. Por outro lado, a distribuição dos dados em $\mathcal{D}$ pode fornecer dicas para a escolha do valor de α_{min}. Em nosso método, ao invés de usar um único valor para α_{min}, que poderia ser aplicado para todos os exemplos indiscriminadamente, usamos um corte α^x_{min} para cada exemplo $x \in \mathcal{D}$. Ou seja, cada exemplo $x \in \mathcal{D}$ é usado como um filtro, que divide o conjunto $\mathcal{D}$ em múltiplas projeções $\mathcal{D}_x$, de forma que cada $\mathcal{D}_x$ consiste dos exemplos $y \in$

$\mathcal{D}$ que compartilham termos com x [Veloso and Meira Jr. 2011]. Em seguida, exploramos cada projeção $\mathcal{D}_x$ de forma a determinar valores mais significativos para cada corte α^x_{min}.

As probabilidades $\alpha(x, \ominus)$ são calculadas utilizando um classificador associativo. Esse tipo de classificador produz um modelo de classificação a partir do conjunto de treinamento, composto por regras da forma $\{X \rightarrow c\}$ que mapeam um conjunto de termos X a uma classe c. A escolha por esse tipo de classificador é importante para garantir a praticidade da solução, já que a enumeração das regras acontece de maneira incremental. De fato, provamos que o processo é totalmente incremental, sem nenhuma replicação de esforço computacional. Os teoremas que descrevem nossa prova não estão incluídos aqui por falta de espaço. Em [Davis et al. 2012] está disponível o conjunto completo de teoremas, bem como uma melhor descrição do classificador associativo utlizado.

3.3. Melhor Corte Individual

O método proposto é chamado *Expectation-Maximization Entropy Minimization* (ou simplesmente EM^2). O método procura, para cada $x \in \mathcal{D}$, o corte α^x_{min} que minimiza a entropia no espaço de probabilidades induzidas por $\mathcal{D}_x$. Ou seja, dados os exemplos $\{y_1, y_2, \ldots, y_n\} \in \mathcal{D}_x$, o método EM^2 primeiramente calcula $\alpha(y_i, \ominus)$ para todos $y_i \in \mathcal{D}_x$. Em seguida, os valores para $\alpha(y_i, \ominus)$ são ordenados de maneira crescente. Idealmente, existe um corte α^x_{min} tal que:

- se $\alpha(y_i, \ominus) \leq \alpha^x_{min}$, então $c^{y_i} = \oplus$ (onde c^{y_i} é a classe associada ao exemplo y_i)
- se $\alpha(y_i, \ominus) > \alpha^x_{min}$, então $c^{y_i} = \ominus$

Uma vez calculado α^x_{min}, ele pode ser utilizado para ativar a operação de transição de rótulo $x^{\ominus \rightarrow \oplus}$, que será executada se x for negativo e $\alpha(x, \ominus) \leq \alpha^x_{min}$. Análogamente, a operação $x^{\oplus \rightarrow \ominus}$ também será realizada se x for positivo e $\alpha(x, \ominus) > \alpha^x_{min}$. No entanto, existem casos mais difíceis, nos quais não é possivel obter uma perfeita separação no espaço de probabilidades. Dessa forma, propomos uma solução mais geral para calcular um corte no espaço de probabilidades. A idéia basica é que qualquer valor de α^x_{min} induz duas partições no espaço de valores para $\alpha(y_i, \ominus)$ (por exemplo, uma partição com valores inferiores a α^x_{min} e outro com valores superiores a α^x_{min}. O método EM^2 procura atribuir um valor para α^x_{min} que minimiza a entropia média nessas duas partições (Figura 2). O método EM^2 converge quando nenhuma operação de transição é realizada, e então o conjunto de treinamento está pronto e pode ser fornecido ao classificador.

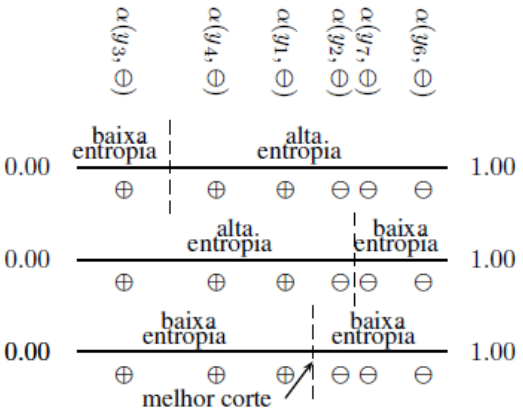

Figura 2. Busca pelo corte que minimiza a entropia.

4. Avaliação Empírica

Para avaliar nosso método de classificação de fluxo de dados, apresentamos dois cenários de aplicação que usam fluxos de mensagens coletadas do Twitter. As mensagens abordam temas que possuem um vocabulário altamente dinâmico, e portanto a geração automática e contínua do conjunto de treinamento é muito importante.

4.1. Desambiguação de Entidades Nomeadas

Definimos nosso primeiro cenário de aplicação da seguinte forma. Dada uma entidade denotada por e, especificamos um conjunto de palavras-chave K que podem ser usadas para referenciar e. Dessa maneira, coletamos um fluxo S contendo mensagens que possivelmente fazem referência a e. Nosso objetivo é classificar cada mensagem $t \in S$ de modo que quando $c^t = \oplus$, então t faz referência a e (caso contrário, $c^t = \ominus$). Por exemplo, considerando a entidade "Clube Atlético Mineiro", podemos definir o conjunto $K =$ { "Atlético-MG", "Atlético", "Alvinegro", "Galo", "Atlético Mineiro" }. Uma mensagem contendo, por exemplo, a palavra-chave "Galo", necessariamente faz parte do fluxo S, mas não necessariamente referencia a entidade "Clube Atlético Mineiro". Ou seja, as palavras-chave em K são ambíguas, e ao classificarmos as respectivas mensagens em $c^t = \{\ominus, \oplus\}$ estamos essencialmente removendo a ambiguidade dessas palavras.

Tabela 1. Resultados da Desambiguação de Entidades Nomeadas

	EM^2			Biased-SVM		
	AUC	F-score	tempo(seg)	AUC	F-score	tempo(seg)
Fluminense Football Club	**0.76**	**0.59**	4.2	0.74	0.57	425.2
Sport Club Internacional	**0.78**	**0.76**	3.2	0.74	0.70	301.7
Sociedade Esportiva Palmeiras	**0.79**	**0.67**	2.4	0.74	0.57	253.1
São Paulo Football Club	**0.87**	**0.84**	1.3	0.84	0.81	122.8
Clube Atlético Mineiro	**0.68**	**0.66**	2.0	0.62	0.59	181.3
Cruzeiro Esporte Clube	**0.80**	**0.74**	2.1	0.68	0.65	208.1

Coletamos seis fluxos de mensagens, onde cada fluxo é composto por mensagens contendo pelo menos uma palavra-chave previamente especificada. As palavras-chave foram escohidas de forma a recuperar mensagens referentes a equipes do Campeonato Brasileiro de Futebol. Sendo assim, cada fluxo é composto por mensagens que potencialmente fazem referência a uma equipe. As mensagens foram coletadas de 21/11 a 21/12/2010, e cada fluxo contém cerca de 500 mil mensagens. Para cada fluxo, um conjunto adicional de mil mensagens foi rotulado manualmente, no intuito de validar o método EM^2. Mais especificamente, o conjunto de treinamento é gerado automaticamente utilizando as mensagens em S. Em seguida o conjunto de treinamento obtido é fornecido ao classificador, o qual prevê as classes para o conjunto adicional de 1000 mensagens. Testes de significancia foram executados ($p < 0.05$) e os melhores resultados, inclusives empates estatísticos são mostrados em negrito. A Tabela 1 traz a efetividade obtida para cada entidade considerada, comparando os resultados obtidos com o método Biased SVM [Liu et al. 2003], que é o representante do estado da arte para esse tipo de dado. Para avaliar os resultados utilizamos métricas tradicionais como o *F-Score* e valores *AUC*. Para cada conjunto positivo+não rotulado (PU) utilizado, escolhemos aleatoriamente $x\%$ de exemplos positivos (P) para se tornarem não rotulados (U). Esse processo nos permite controlar o nível de incerteza do conjunto inicial. Para cada nível temos uma combinação diferente de taxas de falsos positivos e verdadeiros-positivos, assim, podemos traçar curvas ROC e calcular a área sob elas (valores AUC).

4.2. Análise de Sentimento

Definimos nosso segundo cenário de aplicação da seguinte forma. Dada uma entidade denotada por e e um fluxo de mensagens S que verdadeiramente fazem referência a e, nosso objetivo é classificar cada mensagem em $t \in S$ de modo que quando $c^t = \oplus$, então t tem conotação favoravel em relação a e (caso contrário, $c^t = \ominus$). Uma singularidade

deste cenário de aplicação é que, além dos conjuntos P e U, também podemos obter um terceiro conjunto N, o qual contém um número restrito de mensagens com conotação negativa em relação a e. Por exemplo, a *hashtag* #DilmaJá acompanha mensagens de conotação positiva em relação à entidade Dilma Rousseff, ao passo em que a *hashtag* #DilmaNão acompanha mensagens de conotação negativa em relação à mesma entidade.

Desta forma, podemos aplicar duas instanciações de nosso método EM^2: uma instanciação nos conjuntos P e U, e outra nos conjuntos N e U. Ao final, podemos combinar a saída das duas instanciações, de forma a escolher a previsão mais segura para cada mensagem $t \in S$, caso haja discordância entre elas. Avaliamos nosso método utilizando um fluxo de mensagens referentes à candidata Dilma Rousseff, durante as eleições presidenciais de 2010. As mensagens foram coletadas de 11/03 a 30/10/2010, e o fluxo S resultante soma cerca de 67 mil mensagens. O fluxo foi dividido em 12 janelas contendo cerca de 6000 mensagens cada. Dessa forma, simulamos um ambiente real de fluxo de mensagens, onde uma janela de mensagens J_i é utilizada para geração do conjunto de treinamento, enquanto a janela J_{i+1} é utilizada para avaliar as previsões realizadas pelo classificador. Os resultados (Figura 3) indicam que a combinação da saída das duas instanciações tem desempenho ligeiramente superior ao de cada uma em separado.

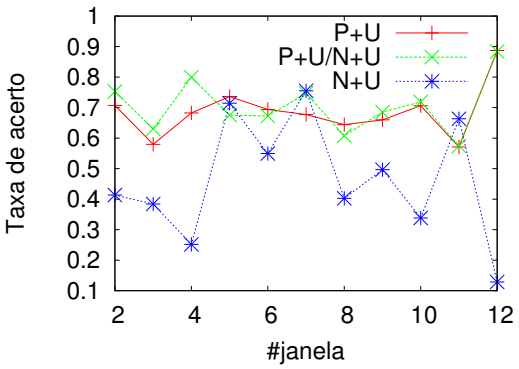

Figura 3. Análise de sentimento em um fluxo de mensagens simulado

5. Sistema em Produção

O método EM^2 foi implementado e transformado em um sistema que encontra-se atualmente em produção no projeto "Observatório da Web"[1]. O sistema realiza a classificação de mensagens postadas em tempo real, removendo a ambiguidade intrinseca das mensagens. A Figura 4 mostra uma série de termos e sentenças que apareceram nas mensagens referentes a uma partida de futebol. Analisando esses termos e sentenças, podemos perceber que as mensagens consideradas foram devidamente desambiguadas, caso contrário seria de se esperar sentenças não relacionadas com a partida em questão.

Figura 4. Entidades ambíguas.

Para avaliar nosso sistema, analisamos a acurácia da desambiguação da entidade "Internacional Sport Club", que é altamente ambígua (i.e., "internacional", "inter", etc.), durante 10 dias de coleta (08/10 a 18/10/2011). Nossa avaliação leva em conta três

[1] http://observatorio.inweb.org.br

configurações diferentes, com o objetivo de investigar o efeito causado pelo intervalo temporal entre as mensagens no conjunto de treinamento e as mensagens a serem desambiguadas. Na primeira configuração, o conjunto de treinamento é gerado usando todas as mensagens coletadas durante uma hora qualquer h_i, e avaliamos as previsões usando as mensagens coletadas durante a hora imediatamente seguinte h_{i+1} (ou seja, as mensagens a serem desambiguadas chegaram logo após as mensagens no conjunto de treinamento). Na segunda configuração, um intervalo de uma hora é embutido, ou seja, o conjunto de treinamento contém mensagens que chegaram durante o período h_i, enquanto as mensagens a serem desambiguadas chegaram no período h_{i+2}. Finalmente, na última configuração, aumentamos o intervalo para 2 horas. A acurácia da classificação cai consistentemente ao aumentarmos a diferença temporal entre as mensagens no conjunto de treinamento e as mensagens a serem desambiguadas (Figura 5), indicando a necessidade de manter o conjunto de treinamento sempre atualizado neste tipo de aplicação altamente dinâmica.

6. Conclusões e Trabalhos Futuros

Neste trabalho propomos um novo método para classificação em tempo real de fluxos de dados, cenário comumente observado nas redes sociais. Dois cenários de aplicação foram utilizados em nossa avaliação: desambiguação de entidades e análise de sentimentos. Nosso método alcança uma acurácia 12% maior do que a acurácia obtida pelo baseline, além de ser aproximadamente 100 vezes mais rápido. Em média, em ambos os cenários de aplicação, a acurácia obtida por nosso método foi de aproximadamente 70%, o que é aceitável, considerando que nenhuma supervisão é necessária após o início do processo. Nosso método foi implementado como um sistema, tendo papel importante em um projeto mais amplo, chamado "Observatório da Web". Nosso sistema vem sendo utilizado em diversos cenários de desambiguação de entidades: Campeonato Brasileiro de Futebol, Liga de Futebol Americano, e Eleições Norte-Americanas. Um dos benefícios oferecidos por nosso sistema de classificação em tempo real é a melhora na precisão da coleta dos dados. Como trabalho futuro, pretendemos construir classificadores multi-rótulo, através da combinação de vários classificadores binários.

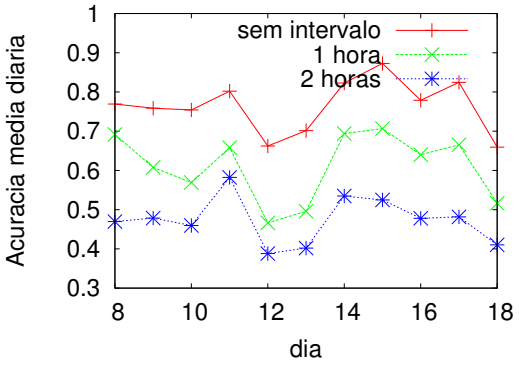

Figura 5. Acurácia média diária para 10 dias de coleta.

7. Considerações Finais

Agradecemos a colaboração dos professores Alberto Laender e Wagner Meira Jr. (DCC - UFMG), e Altigran da Silva (DCC - UFAM), que nos auxiliaram no desenvolvimento deste trabalho. Agradecemos também o suporte financeiro oferecida pelo Programa UOL-Bolsa Pesquisa. Trechos deste trabalho foram publicados [2] no Simpósio Brasileiro de Bancos de Dados [Davis et al. 2011] e na principal conferência em Linguística Computacional, a ACL [Davis et al. 2012].

[2]Os artigos estão disponíveis em `http://www.dcc.ufmg.br/~agdavis/sbbd2011.pdf` e `http://www.dcc.ufmg.br/~agdavis/acl_final.pdf`).

Referências

Batista, G. and Monard, C. (2003). An analysis of four missing data treatment methods for supervised learning. *Applied Artificial Intelligence*, 17(5-6):519–533.

Comité, F. D., Denis, F., Gilleron, R., and Letouzey, F. (1999). Positive and unlabeled examples help learning. In *Proc. of ALT*, pages 219–230.

Davis, A., Santos, W., Veloso, A., Jr., W. M., Laender, A., and da Silva, A. S. (2011). RT-NED: Real-time named entity disambiguation on Twitter streams. In *Proc. of SBBD: Demos Session*, pages 43–48.

Davis, A., Veloso, A., Jr., W. M., Laender, A., and da Silva, A. S. (2012). Named entity disambiguation in streaming data. In *Proc. of ACL (to appear)*.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B*, 39(1):1–38.

Denis, F. (1998). Pac learning from positive statistical queries. In *Proc. of ALT*, pages 112–126.

Elkan, C. and Noto, K. (2008). Learning classifiers from only positive and unlabeled data. In *Proc. of SIGKDD*, pages 213–220.

Fung, G., Yu, J., Lu, H., and Yu, P. (2006). Text classification without negative examples revisit. *IEEE Trans. Knowl. Data Eng.*, 18(1):6–20.

Kao, B., Lee, S. D., Lee, F., Cheung, D., and Ho, W. (2010). Clustering uncertain data using voronoi diagrams and r-tree index. *IEEE Trans. Knowl. Data Eng.*, 22(9):1219–1233.

Li, X. and Liu, B. (2003). Learning to classify texts using positive and unlabeled data. In *Proc. of IJCAI*, pages 587–592.

Li, X., Liu, B., and Ng, S. (2007). Learning to classify documents with only a small positive training set. In *Proc. of ECML*, pages 201–213.

Li, X., Yu, P., Liu, B., and Ng, S. (2009). Positive unlabeled learning for data stream classification. In *Proc. of SDM*, pages 257–268.

Liu, B., Dai, Y., Li, X., Lee, W., and Yu, P. (2003). Building text classifiers using positive and unlabeled examples. In *Proc. of ICDM*, pages 179–188.

Liu, B., Lee, W., Yu, P., and Li, X. (2002). Partially supervised classification of text documents. In *Proc. of ICML*, pages 387–394.

Rocchio, J. (1971). *The SMART Retrieval System - Experiments in Automatic Document Processing*. Prentice-Hall, Inc.

Schölkopf, B., Platt, J., Shawe-Taylor, J., Smola, A., and Williamson, R. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471.

Veloso, A. and Meira Jr., W. (2011). *Demand-Driven Associative Classification*. Springer-Verlag.