Aprendizado de Máquina Aplicado na Detecção Automática de Comentários Indesejados

Túlio C. Alberto[1], **Tiago A. Almeida**[1]

[1]Departamento de Computação (DComp)
Universidade Federal de São Carlos (UFSCar), Sorocaba, SP – Brasil

tuliocasagrande@gmail.com, talmeida@ufscar.br

Abstract. *A significant amount of the available information on web sites come from the interaction with users, such as news sites and blogs, where readers can post comments and sometimes develop habits of frequenting them. Some blogs specialize in certain subjects, receive the users credibility and become references in the field. However, the ease of inserting content through text comments makes room for unwanted messages, which affect the user experience, reduce the quality of the information provided by the websites and indirectly causing personal and economic losses. Given this scenario, this paper presents an analysis of machine learning techniques applied to automatically detect unwanted comments posted on blogs. Experiments carried out with a real and public database indicate that support vector machines, trained with both attributes extracted from the text messages and metadata from the posts, is promising in the task of filtering unwanted comments.*

Resumo. *É cada vez mais comum que uma parcela significativa da informação disponível em determinados web sites venha da interação com os usuários, como em sites de notícias e blogs, nos quais os leitores podem escrever comentários e por vezes desenvolvem hábitos de frequentar o domínio. Alguns blogs se especializam em determinados assuntos, ganham credibilidade dos usuários e tornam-se referências na área. Contudo, a facilidade encontrada na inserção de conteúdo por meio de comentários abre espaço para mensagens indesejadas, que prejudicam a experiência dos usuários, reduzem a qualidade das informações presentes nos sites e indiretamente provocam prejuízos pessoais e econômicos. Diante desse cenário, esse trabalho apresenta uma análise de técnicas de aprendizado de máquina aplicadas na detecção automática de comentários indesejados postados em blogs. Experimentos realizados com uma base de dados real e pública indicam que a aplicação de máquinas de vetores de suporte, treinadas com atributos extraídos tanto das mensagens quanto dos metadados, é promissora na tarefa de filtragem de comentários indesejados.*

1. Introdução

A interatividade com os usuários é uma marca muito forte da web 2.0. Na atualidade, praticamente qualquer site com conteúdo gerado pelo usuário tem suporte a comentários, tais como páginas de notícias, fotos, blogs etc. Essa interatividade serve tanto como um indicador de importância do conteúdo da página, quanto para auxiliar o autor a corrigir informações que eventualmente estejam incorretas ou incompletas em seu texto [Mishne e Glance 2006]. Por outro lado, essa facilidade de inserção de conteúdo

abre espaço para mensagens indesejadas que atrapalham os leitores e reduzem a qualidade das informações presentes nos sites.

Comentários indesejados causam confusão nos mecanismos de busca e acabam reduzindo o *pagerank* [Page et al. 1998] desses sites, podendo prejudicar os autores pela redução no tráfego de novos leitores. Estatísticas de tráfego são um ponto-chave para quem vende anúncios em seu site e, dessa forma, um número menor de acessos implica na diminuição do interesse de possíveis anunciantes.

A discussão em torno de um *post* é também, para muitos autores da blogosfera, parte indissociável do conteúdo e também um incentivo para que continuem publicando. Segundo [Mishne e Glance 2006], os comentários representam cerca de 30% do conteúdo total dos blogs e, portanto, consideram de extrema importância o efetivo combate ao spam nesses meios, para assegurar aos leitores informações de qualidade e que realmente acrescentem algum valor ao *post*.

Um dos poucos serviços de filtragem de *comment blog spam* atualmente disponível é o Akismet[1]. De acordo com informações presentes no seu web site, mais de 82 bilhões de comentários spam foram detectados desde que entrou em operação, sendo que em um único dia mais de 1,5 milhão de comentários são bloqueados. O volume de comentários legítimos em média é inferior a 15% do total de mensagens enviadas. Além disso, ao contrário do spam enviado por email, grande parte dos comentários spam não são enviados por *spam bots*, mas por meio de pessoas que se passam por usuários legítimos e tentam transmitir mensagens com links e propagandas para os leitores de blogs, sem que seus autores percebam[2].

Diante desse cenário, este trabalho apresenta uma análise de desempenho de diversas técnicas bem conhecidas de aprendizado de máquina que podem ser aplicadas para auxiliar no combate desse problema. O objetivo é encontrar métodos promissores que podem ser explorados e empregados para auxiliar na detecção automática de comentários indesejados postados em blogs e sites de notícias. Comparativamente aos poucos trabalhos existentes na literatura, este artigo oferece as seguintes contribuições:

1. resultados de experimentos realizados com diversos métodos que ainda não foram avaliados na aplicação do problema em questão. Nesse sentido, destacam-se os métodos de *boosting*, métodos baseados em árvores e máquinas de vetores de suporte com diferentes opções de *kernel*;
2. experimentos realizados com diferentes vetores de características (atributos) extraídos dos comentários disponíveis em uma base de dados real e pública; e
3. comparação dos resultados obtidos nesse trabalho com os resultados obtidos por métodos propostos por outros autores da literatura;

Este artigo está estruturado da seguinte forma: na Seção 2 são brevemente descritos os trabalhos correlatos disponíveis na literatura. Na Seção 3 são apresentados os conceitos básicos sobre os métodos classificadores avaliados neste trabalho. Na Seção 4 são descritas as configurações adotadas nos experimentos e a base de dados empregada. Os resultados experimentais são apresentados na Seção 5. Por fim, conclusões e direções para trabalhos futuros são descritos na Seção 6.

[1] *Akismet* – Disponível em `https://akismet.com/`.

[2] *How we stop comment and trackback spam*. Disponível em: `https://akismet.com/how/`.

2. Trabalhos correlatos

O problema abordado é relativamente recente e, portanto, existem poucos trabalhos que oferecem avanços significativos no sentido de resolvê-lo. Em um dos trabalhos pioneiros da literatura, [Mishne et al. 2005] propuseram uma técnica que emprega o método KL-*divergence* para computar a divergência de modelo de linguagem entre o conteúdo do *post* e do comentário. Uma vez que o *post* serve de contexto para o comentário, é possível utilizá-lo como aliado na identificação de comentários indesejados. Essa abordagem tenta criar modelos estatísticos de linguagem por meio de distribuições de probabilidade da ocorrência de cadeias de caracteres dentro de um texto para verificar a presença de links que possam levar a páginas indesejadas. Os autores também disponibilizaram a primeira base de dados pública de blog spam que, desde então, vem sendo empregada em estudos posteriores [Mishne e Glance 2006, Romero et al. 2010, Bhattarai e Dasgupta 2011].

[Mishne e Glance 2006] estudaram as relações entre comentários e *posts*, e estimaram que o volume dos comentários em blogs corresponde a cerca de 30% do conteúdo total dos *posts*. Estudaram também relações entre popularidade e padrões de comentários através da comparação do número de visualizações e número de comentários. Ao medir a contribuição dos comentários para o blog, observou-se que estes são de vital importância, uma vez que mecanismos de pesquisa também indexam os comentários e, portanto, quanto mais relevante o conteúdo destes, maior será o *pagerank* do site nesses mecanismos, o que aumenta o número de acessos ao site.

[Cormack et al. 2007] avaliaram a eficácia de métodos tradicionais, usados na detecção de spam enviado por email, na filtragem de spam em mensagens curtas e com menos informações, como comentários de blogs, SMS e postagens em fóruns. Os autores concluíram que o conteúdo das mensagens por si só não fornece informações suficientes para os classificadores e, portanto, é necessário empregar técnicas adicionais para expandir o espaço de atributos.

[Romero et al. 2010] realizaram um estudo comparativo de quatro métodos de aprendizado de máquina na classificação de *blog comment spam*, usando atributos extraídos dos *posts*. Os classificadores avaliados foram: Naïve Bayes, *k*-vizinhos mais próximos, redes neurais artificiais e SMO, sendo que o último obteve o melhor resultado.

[Shin et al. 2011] propuseram um modelo de filtragem de spam pela análise de informações como origem geográfica, horário, número de URL's, além do conteúdo dos comentários. Originalmente idealizado para fóruns de discussão, o problema abordado possui características semelhantes aos blogs, como quantidade de informações insuficiente no conteúdo da mensagem. Os resultados indicam que o método é promissor, porém o filtro foi desenvolvido para um único site e pode não ser efetivo quando empregado de forma mais genérica.

[Bhattarai e Dasgupta 2011] apresentaram um método auto-supervisionado de classificação baseado em árvores de decisão. Segundo a avaliação dos autores, o método ainda precisa ser melhorado, mas é flexível o suficiente para ser utilizado de forma mista, na qual parte da supervisão é feita por seres humanos e outra parte pelo próprio algoritmo.

[Kantchelian et al. 2012] partiram da hipótese de que é considerado spam qualquer conteúdo não informativo. Em seguida, criaram uma métrica chamada *content complexity* para medir o nível de informação presente em um comentário. Como forma de incrementar a métrica, os autores agruparam os comentários que possuíam o mesmo usuário, o mesmo endereço IP, a citação dos mesmos *links* e outros dados. Utilizando

uma base de dados privada, os autores avaliaram um classificador baseado em regressão logística, aplicando a métrica proposta.

[Wang et al. 2012] estudaram um problema que eles chamaram de comentários de distração, ou *diversionary comments*. Segundo os autores, muitos blogs políticos sofrem de tais comentários, que visam distrair ou desviar a atenção dos leitores sobre o assunto inicial do *post*. Os autores consideraram o assunto como ainda inédito na literatura e aplicaram filtros como o KL*-divergence* para medir a similaridade entre o conteúdo do *post* e do comentário.

[Chu et al. 2013] propuseram uma técnica para detectar *blog bots* – programas automáticos que preenchem formulários e postam comentários. Diferentemente dos métodos de detecção convencionais, que requerem participação direta do usuário, como o CAPTCHA, os autores apresentaram uma abordagem de detecção que utiliza biometria comportamental, como o uso do mouse e do teclado, para distinguir entre uma pessoa real e um *blog bot*.

3. Classificadores

Apesar da existência de trabalhos cujo intuito é filtrar *comment spam*, ainda não há um consenso se a aplicação de técnicas de aprendizado de máquina é realmente eficaz. Tendo isso em vista, foram avaliados neste trabalho diversos métodos de classificação bem conhecidos e amplamente empregados na literatura.

Nessa seção, são brevemente descritos os métodos classificadores avaliados nesse trabalho: Naïve Bayes, máquinas de vetores de suporte (SVM – *support vector machines*), métodos baseados em árvores (C4.5 e floresta aleatória), IBK, *boosting* adaptativo (AdaBoost – *adaptive boosting*), regressão logística e método baseado em regras (PART). A escolha de tais métodos reside no fato de terem sido avaliados e listados como as melhores técnicas de mineração de dados e classificação atualmente disponíveis [Wu et al. 2008].

A técnica de floresta aleatória também foi escolhida, mesmo não estando na lista originalmente proposta por [Wu et al. 2008], pois ela faz uma combinação de árvores de decisão. Logo, como o método C4.5 é um algoritmo de árvore de decisão e está na lista dos melhores métodos, acredita-se que a combinação de árvores de decisão também possa conduzir a bons resultados.

3.1. Naïve Bayes

O classificador *Naïve* Bayes (NB) é o filtro mais simples derivado da teoria da decisão Bayesiana. Do teorema de Bayes e da teoria da probabilidade total, sabe-se que a probabilidade de uma mensagem $\vec{x} = \langle x_1, \ldots, x_n \rangle$ pertencer a categoria $c_i \in \{c_s, c_l\}$ é $P(c_i|\vec{x}) = \frac{P(c_i).P(\vec{x}|c_i)}{P(\vec{x})}$.

Uma vez que o denominador não depende da classe, o filtro NB classifica cada mensagem na categoria que maximiza $P(c_i).P(\vec{x}|c_i)$. Isso é equivalente a classificar uma mensagem como *spam* (c_s) se $\frac{P(c_s).P(\vec{x}|c_s)}{P(c_s).P(\vec{x}|c_s)+P(c_l).P(\vec{x}|c_l)} > T$, com $T = 0,5$. Variando-se T, obtêm-se mais verdadeiros negativos e menos verdadeiros positivos ou vice-versa. A probabilidade $P(c_i)$ pode ser estimada pela frequência de documentos que pertencem a classe c_i no conjunto de treinamento Tr, enquanto que $P(\vec{x}|c_i)$ é praticamente impossível de ser estimada diretamente, pois isso requer que Tr contenha mensagens idênticas as que estão sendo classificadas. Entretanto, os classificadores NB fazem a suposição de que

os termos de uma mensagem são condicionalmente independentes e que a ordem em que eles aparecem é irrelevante.

Apesar dessa suposição ser um tanto quanto simplista, diversos estudos indicam que o classificador NB é surpreendentemente eficaz na filtragem de *spams* [Almeida et al. 2011].

3.2. Máquinas de vetores de suporte

Máquinas de vetores de suporte (SVM – *support vector machines*) é um método de aprendizagem de máquina que pode ser usado para problemas de classificação e regressão e outras tarefas de aprendizagem. Elas foram conceitualmente implementadas seguindo a ideia de que vetores de entrada são não-linearmente mapeados para um espaço de atributos de alta dimensão. Nesse espaço, é construída uma superfície de decisão que permite distinguir as classes dos exemplos de entrada.

Segundo [Haykin 1998], a ideia principal do SVM, no contexto de dados linearmente separáveis, é construir uma superfície de decisão por meio de um hiperplano ótimo com máxima margem de separação entre os dados de classes diferentes. Já no contexto de dados não-separáveis, o objetivo do SVM é construir um hiperplano ótimo que minimize a probabilidade de erro de classificação em relação ao conjunto de treinamento.

Para separar dados linearmente ou não-linearmente separáveis, o principal recurso usado pelo método SVM é uma função de *kernel* $k(x_i, x_j)$. Através dela, o SVM constrói uma superfície de decisão que é não-linear no espaço de entrada, mas é linear no espaço de atributos [Haykin 1998]. As principais funções de *kernel* que podem ser utilizadas no SVM são [Haykin 1998]: linear, função de base radial (RBF) e polinomial.

- linear: $k(x_i, x_j) = x_i^T x_j$;
- função de base radial (RBF): $k(x_i, x_j) = \exp(-\frac{1}{2\gamma^2}\|x_i - x_j\|^2), \gamma > 0$;
- polinomial: $k(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \gamma > 0$;

Nas funções de *kernel* apresentadas acima, os parâmetros γ, r e d devem ser especificados a priori pelo usuário.

Para a escolha dos parâmetros do SVM, a técnica recomendada em [Haykin 1998] é a *grid search* (busca em grade). Então, considerando o SVM com *kernel* RBF, em que deve-se definir o parâmetro de regularização C e o parâmetro γ, eles propõem testar as seguintes sequências exponenciais: $C = 2^{-5}, 2^{-4}, 2^{-3}..., 2^{15}$ e $\gamma = 2^{-15}, 2^{-14}..., 2^{3}$.

3.3. C4.5

O C4.5 é um dos mais clássicos algoritmos de árvores de decisão e trabalha tanto com atributos categóricos quanto contínuos. Ele emprega um método de dividir e conquistar para aumentar a capacidade de predição das árvores de decisão. Dessa forma, um problema é dividido em vários sub-problemas, criando-se sub-árvores no caminho entre a raiz e as folhas da árvore de decisão.

O C4.5 gera um classificador na forma de árvores de decisão que em sua estrutura possuem folhas, que indicam alguma classe do problema e nós de decisão, que especificam algum teste que deve ser realizado sobre um valor de atributo [Quinlan 1993].

3.4. Floresta aleatória

Uma floresta aleatória (*random forest*) é uma combinação de árvores de decisão, em que cada árvore depende dos valores de vetores aleatórios amostrados de forma independente e distribuídos igualmente para todas as árvores na floresta.

Na construção de uma floresta aleatória para a k-ésima árvore de decisão, um vetor aleatório Θ é gerado, independente dos vetores aleatórios $\Theta_1, ..., \Theta_{k-1}$ gerados no passado, porém usando a mesma distribuição. A árvore é cultivada usando o vetor aleatório gerado e um conjunto de treinamento. Desta forma, o resultado é a criação de um classificador $h(x, \Theta_k)$, onde x é um vetor de entrada. Para cada vetor aleatório gerado, uma nova árvore de decisão é construída. Depois que um determinado número de árvores são geradas, cada uma emite um voto para uma classe do problema, considerando o vetor de entrada x. Por fim, a classe mais votada é escolhida na predição do classificador [Breiman 2001].

3.5. IBK

O IBK é um método de aprendizagem baseado em instâncias (IBL – *instance-based learning*). Esse tipo de algoritmo é derivado do método de classificação k-vizinhos mais próximos (k-*nearest neighbor*). Porém, este último é um algoritmo não-incremental e tem como objetivo principal manter uma consistência perfeita com o conjunto inicial de treinamento. Já o algoritmo do tipo IBL é incremental e tem como objetivo maximizar a acurácia sobre novas instâncias do problema [Aha et al. 1991].

O algoritmo IBL usa uma função que mapeia padrões de um problema para categorias, sendo que a saída inicial do classificador é uma categoria. Logo, dado um padrão do conjunto de dados, a predição produzida pelo algoritmo é dada pelo valor previsto para a categoria a qual este padrão pertence. Ele emprega uma função de similaridade que calcula a proximidade entre o padrão de treinamento x e os demais padrões já incluídos em cada categoria. A função de classificação, que recebe o resultado da função de similaridade e os registros de desempenho da classificação dos padrões que possuem a mesma categoria do padrão x, gera uma predição para x. Por fim, o atualizador de categorias mantém os registros sobre o desempenho da classificação e decide quais padrões devem ser incluídos em cada categoria [Aha et al. 1991].

3.6. *Boosting* adaptativo

O método de *boosting* adaptativo (AdaBoost) [Freund e Schapire 1996] é um algoritmo de *boosting* amplamente utilizado para problemas de classificação. Em geral, assim como qualquer método de *boosting*, ele faz uma combinação de classificadores. Porém, segundo [Freund e Schapire 1996], ele possui algumas propriedades que o tornam mais prático e fácil de ser implementado do que os algoritmos de *boosting* que o antecederam, pois ele não necessita de nenhum conhecimento prévio das predições obtidas por classificadores ruins. Ao invés disso, ele se adapta às predições incorretas e gera uma hipótese majoritária ponderada, em que o peso das predições fornecidas pelos classificadores "ruins" torna-se uma função de sua predição.

3.7. Regressão Logística multinomial

Regressão logística multinomial, é um modelo usado para prever a probabilidade de diferentes resultados possíveis de uma variável dependente categoricamente distribuída, dado um conjunto de variáveis independentes. O uso do termo "multinomial" em seu nome surgiu a partir da fusão comum entre as distribuições categóricas e multinomial [Greene 2003].

3.8. PART

O método PART combina os métodos C4.5 e RIPPER em uma tentativa de eliminar os problemas de ambos os algoritmos isolados. A maior vantagem do PART em relação aos outros dois é que o PART não precisa executar otimizações globais para gerar conjuntos precisos de regras. O método PART usa a estratégia de divisão e conquista na qual ele cria um conjunto de regras eliminando todas as instâncias que este conjunto abrange e repete este algoritmo recursivamente até que todas as instâncias sejam eliminadas. Este método se difere das abordagens padrões na maneira com que cada regra é criada. Essencialmente, para criar uma única regra, uma árvore de decisão é construída para o conjunto de dados e, posteriormente, o nó-folha com maior cobertura é selecionado para definir a regra [Frank e Witten 1998].

4. Configurações e base de dados

Para tornar os resultados completamente reprodutíveis, são apresentadas nessa seção as configurações adotadas para cada classificador, além de informações gerais sobre a base de dados e a metodologia experimental empregada.

4.1. Configurações

O SVM foi implementado utilizando a biblioteca LIBSVM [Chang e Lin 2011]. Foram feitas simulações com as funções de *kernel* linear, RBF e polinomial. A técnica de *grid search* foi empregada para a definição dos parâmetros. Porém, para o SVM com *kernel* polinomial, que possui maior número de parâmetros, optou-se por realizar *grid search* apenas sobre os parâmetros C e γ, devido ao excessivo custo computacional. Neste caso, adotou-se os valores padrões da LIBSVM para os demais parâmetros.

A *grid search* foi realizada com um conjunto de treino (80% dos dados) e teste (20% dos dados), escolhidos aleatoriamente para cada conjunto de atributos avaliado. Após a execução, os melhores parâmetros foram escolhidos e empregados na realização dos experimentos. A Tabela 1 apresenta os parâmetros usados nas simulações com o método SVM.

Tabela 1. Parâmetros obtidos por *grid search* que foram empregados no método SVM.

Kernel	C	γ
Linear	1.0	–
RBF	1.0	0.1
Polinomial	1.0	0.1

Os demais métodos foram implementados usando a ferramenta WEKA [Hall et al. 2009] com parâmetros padrões definidos na ferramenta.

4.2. Base de dados

A base de dados utilizada foi criada por [Mishne et al. 2005][3] e trata-se de uma coleção real, pública e não codificada constituída por 1024 comentários extraídos de 50 blogs, classificados manualmente como spam ou legítimos.

[3] *Information and Language Processing Systems (ILPS)*: *Blog Spam Corpus*. Disponível em `http://ilps.science.uva.nl/resources/commentspam`.

A base apresenta diferentes tipos de *comment spam*, sendo que alguns consistem de uma simples lista de palavras-chaves seguidas de links para outros sites, enquanto que outros empregam frases sofisticadas. As mensagens estão distribuídas nas classes da seguinte forma: 68%(692) spam e 32%(332) ham (mensagens legítimas).

Cada mensagem (*post*) é composta por informações da postagem (metadados – em `typewriter`) e o texto em si (em *itálico*), conforme exemplo a seguir.

```
comment id="26" post="Posted by:  &lt;a target=&quot;
_blank&quot; title=&quot; http://www.bestukcasinos.co.uk &quot;
href=&quot; mt-comments.cgi?__mode=red; id=23964&quot; &gt; UK
Casinos&lt; /a&gt; at January 25, 2004 05:53 PM"
```

Thanks for the great blog!Best Regards,Marry———————————The best online roulette right here.Play blackjack online today.Online slots gambling.Quality keno gaming action.Play online video poker today!

Dessa forma, foram realizados três experimentos com cada método de classificação:

1. usando somente atributos extraídos dos textos das mensagens, totalizando 11.901 atributos;
2. usando somente atributos extraídos dos metadados, totalizando 2.442 atributos; e
3. usando todos os atributos, totalizando 13.776 atributos (excluindo repetições).

Foi adotada a representação binária para representar cada mensagem e, portanto, cada atributo indica a sua presença (ausência) na amostra. Para extração dos termos (*tokens*) foi empregado um tokenizador para obter qualquer sequência de caracteres separados por: espaço em branco, tabulação, quebra de linha, ponto, vírgula, ponto e vírgula ou traço. Dessa forma, pretende-se preservar outros símbolos que podem auxiliar na classificação das mensagens. Além disso, nenhum tipo de seleção de atributos ou pré-processamento foi realizado, como eliminação de *stopwords* ou *stemming*, pois resultados de pesquisa apontam que tais técnicas tendem a prejudicar o desempenho dos métodos de aprendizado de máquina para classificação de spam [Almeida et al. 2011].

4.3. Metodologia experimental

Todos os experimentos foram realizados utilizando validação cruzada com 10 partições. Para avaliar os classificadores, foram empregadas medidas de desempenho bastante populares na literatura de aprendizagem de máquina e filtragem de spam, tais como:

- Acurácia (Acc) – porcentagem de amostras corretamente classificadas;
- *Spam Caught* (SC) – porcentagem de spam corretamente classificados;
- *Blocked Ham* (BH) – porcentagem de mensagens legítimas erroneamente classificadas como spam;
- F-medida – média harmônica entre precisão e sensitividade;
- *Matthews correlation coefficient* (MCC) – mede a qualidade da classificação binária em função da quantidade de verdadeiros (falsos) positivos (negativos), retornando valores reais entre -1 e $+1$, sendo que: um coeficiente igual a $+1$ indica uma predição perfeita; 0, uma predição aleatória média; e -1, uma predição inversa (Equação 1) [Almeida et al. 2011].

$$MCC = \frac{(v_p \times v_n) - (f_p \times f_n)}{\sqrt{(v_p + f_p) \times (v_p + f_n) \times (v_n + f_p) \times (v_n + f_n)}} \quad (1)$$

5. Resultados

Nessa seção, são apresentados os resultados da detecção automática de *blog comment spam* obtidos pelos métodos de aprendizado de máquina. As Tabelas 2, 3 e 4 apresentam os resultados obtidos por cada método de classificação usando atributos extraídos apenas dos textos dos comentários, dos metadados das postagens e da combinação de ambos, respectivamente. Os resultados estão ordenados por MCC e os valores em negrito destacam os melhores desempenhos para cada medida de avaliação.

Tabela 2. Resultados obtidos na detecção de *blog comment spam* usando atributos extraídos dos textos das mensagens.

Classificador	Acc (%)	SC (%)	BH (%)	F-medida	MCC
Boosted NB	**87,50 ± 3,10**	92,63 ± 3,00	23,20 ± 6,78	**0,909 ± 0,022**	**0,711 ± 0,074**
SVM-Lin	86,52 ± 2,11	93,78 ± 2,16	28,66 ± 6,79	0,904 ± 0,014	0,685 ± 0,052
Boosted C4.5	85,16 ± 2,06	91,33 ± 2,65	27,71 ± 4,43	0,893 ± 0,015	0,656 ± 0,049
Naive Bayes	84,08 ± 2,78	85,84 ± 3,72	**19,57 ± 5,07**	0,879 ± 0,022	0,650 ± 0,059
Logistic	85,05 ± 2,75	93,06 ± 1,69	31,64 ± 6,29	0,894 ± 0,019	0,648 ± 0,068
PART	84,58 ± 3,28	90,91 ± 3,25	28,65 ± 7,96	0,889 ± 0,023	0,642 ± 0,080
SVM-Pol	82,33 ± 4,62	89,01 ± 8,72	31,70 ± 12,48	0,870 ± 0,041	0,601 ± 0,092
RandomForest	82,81 ± 2,99	92,78 ± 2,23	37,99 ± 8,22	0,880 ± 0,020	0,593 ± 0,077
SVM-RBF	81,16 ± 4,95	84,28 ± 5,54	25,34 ± 5,04	0,857 ± 0,040	0,582 ± 0,101
C4.5	80,86 ± 2,77	88,73 ± 2,31	35,52 ± 8,59	0,863 ± 0,018	0,552 ± 0,071
IBk-1	70,99 ± 2,26	99,13 ± 0,95	87,69 ± 8,04	0,822 ± 0,011	0,245 ± 0,096
IBk-3	68,07 ± 0,78	99,86 ± 0,43	98,18 ± 2,01	0,809 ± 0,004	0,069 ± 0,077
IBk-5	67,87 ± 0,59	**99,86 ± 0,43**	98,79 ± 1,48	0,808 ± 0,003	0,049 ± 0,064

Tabela 3. Resultados obtidos na detecção de *blog comment spam* usando atributos extraídos dos metadados das postagens.

Classificador	Acc (%)	SC (%)	BH (%)	F-medida	MCC
SVM-Lin	**93,07 ± 2,06**	95,10 ± 3,28	11,11 ± 5,93	**0,949 ± 0,016**	**0,845 ± 0,046**
SVM-Pol	93,07 ± 2,31	95,53 ± 2,59	12,01 ± 5,92	0,949 ± 0,017	0,843 ± 0,052
SVM-RBF	92,78 ± 2,09	**96,83 ± 2,11**	15,63 ± 6,89	0,948 ± 0,015	0,835 ± 0,049
IBk-1	92,29 ± 2,76	92,93 ± 3,66	**8,99 ± 5,78**	0,942 ± 0,021	0,830 ± 0,060
Logistic	91,21 ± 2,22	92,35 ± 2,78	11,13 ± 4,00	0,934 ± 0,017	0,804 ± 0,049
RandomForest	91,22 ± 2,25	95,68 ± 2,47	18,06 ± 5,32	0,936 ± 0,016	0,798 ± 0,052
Boosted C4.5	90,82 ± 2,28	93,36 ± 2,50	14,41 ± 7,24	0,932 ± 0,016	0,792 ± 0,054
IBk-3	90,82 ± 3,01	92,93 ± 3,05	13,57 ± 6,11	0,932 ± 0,022	0,792 ± 0,070
C4.5	90,04 ± 2,12	92,64 ± 2,91	15,31 ± 8,14	0,926 ± 0,015	0,776 ± 0,052
PART	89,94 ± 2,63	92,92 ± 2,84	16,27 ± 7,40	0,926 ± 0,019	0,771 ± 0,062
Boosted NB	89,65 ± 2,98	92,64 ± 3,41	16,56 ± 5,91	0,924 ± 0,022	0,765 ± 0,069
IBk-5	89,46 ± 3,35	93,08 ± 3,43	18,07 ± 6,33	0,923 ± 0,025	0,759 ± 0,077
Naive Bayes	87,89 ± 3,34	91,77 ± 4,06	20,16 ± 6,29	0,911 ± 0,025	0,725 ± 0,074

A maioria dos métodos avaliados foi capaz de detectar uma quantidade expressiva de *comment spam*, independentemente do conjunto de atributos empregado. Observa-se que, em todos os testes, a maior parte das técnicas foi capaz de filtrar mais de 90% das mensagens indesejadas. Contudo, é notório que os atributos extraídos dos metadados das postagens têm forte influência na separabilidade das classes. É importante observar que, de maneira geral, uma grande quantidade de mensagens legítimas foi bloqueada pelos métodos quando empregados apenas dados extraídos dos textos das mensagens. Isso se deve, muito provavelmente, ao fato de que muitos comentários indesejados são

Tabela 4. Resultados obtidos na detecção de *blog comment spam* usando atributos extraídos dos textos das mensagens e dos metadados.

Classificador	Acc (%)	SC (%)	BH (%)	F-medida	MCC
SVM-Lin	**93,75 ± 2,10**	96,68 ± 1,45	12,35 ± 5,79	**0,954 ± 0,015**	**0,857 ± 0,049**
Boosted C4.5	92,68 ± 1,44	94,51 ± 1,68	**11,14 ± 4,45**	0,946 ± 0,010	0,834 ± 0,034
Boosted NB	92,09 ± 2,90	94,95 ± 2,43	13,86 ± 10,10	0,942 ± 0,020	0,820 ± 0,070
SVM-Pol	90,82 ± 2,88	96,82 ± 1,69	21,68 ± 8,92	0,935 ± 0,019	0,788 ± 0,069
Logistic	90,33 ± 1,99	93,06 ± 2,80	15,36 ± 4,54	0,929 ± 0,015	0,781 ± 0,045
PART	88,38 ± 2,08	90,46 ± 2,61	15,98 ± 5,89	0,913 ± 0,015	0,739 ± 0,048
C4.5	87,80 ± 2,17	91,63 ± 3,40	20,16 ± 4,97	0,910 ± 0,017	0,722 ± 0,050
RandomForest	87,70 ± 2,66	93,93 ± 2,32	25,29 ± 7,84	0,912 ± 0,018	0,714 ± 0,064
Naive Bayes	86,42 ± 2,11	87,72 ± 4,13	16,26 ± 3,02	0,897 ± 0,018	0,702 ± 0,038
IBk-1	83,69 ± 2,77	97,98 ± 1,60	46,12 ± 7,84	0,891 ± 0,017	0,621 ± 0,072
IBk-3	78,71 ± 2,47	99,13 ± 0,96	63,89 ± 7,45	0,863 ± 0,014	0,501 ± 0,070
IBk-5	75,98 ± 2,29	99,71 ± 0,58	73,51 ± 6,65	0,849 ± 0,012	0,429 ± 0,074
SVM-RBF	72,07 ± 1,47	**99,71 ± 0,58**	85,56 ± 4,32	0,828 ± 0,008	0,305 ± 0,060

também escritos e postados por humanos, de tal forma que são facilmente confundidos com mensagens legítimas, uma vez que normalmente são textos curtos e oferecem pouca informação ao classificador. Essa observação confirma o que já havia sido constatado pelo serviço Akismet. Além disso, as mensagens de ambas as classes frequentemente possuem gírias, abreviações e símbolos que também dificultam a separabilidade das classes. Com relação aos métodos avaliados, os resultados indicam que o SVM linear, na média, apresenta o melhor desempenho. Ele foi capaz de detectar em média 96,68% dos comentários indesejados, quando utilizados atributos extraídos dos textos das mensagens e dos metadados. Os SVMs com *kernel* não-linear obtiveram resultados satisfatórios quando empregados apenas atributos extraídos dos metadados, uma vez que tal conjunto de dados possui dimensionalidade bem inferior aos demais. Essa constatação vem ao encontro de outros resultados da literatura, que também sugerem que o método SVM linear é mais indicado para problemas de categorização e classificação de texto, devido à alta dimensionalidade do hiperespaço de atributos [Haykin 1998]. Por outro lado, o método IBk em média apresentou o pior desempenho e bloqueou uma quantidade inaceitável de mensagens legítimas, quando atributos dos textos das mensagens foram empregados.

Para facilitar a avaliação dos métodos, é apresentado na Tabela 5 uma comparação entre os melhores resultados desse trabalho e os disponíveis na literatura de *blog comment spamming* que empregam a mesma coleção de dados e metodologia de avaliação.

Tabela 5. Comparação entre os melhores resultados obtidos nesse trabalho e os resultados disponíveis na literatura

Classificadores	Acurácia (%)	F-medida
Resultados disponíveis na literatura		
KL-*divergence* [Mishne et al. 2005]	83,00	—
SMO [Romero et al. 2010]	84,61	—
C4.5 [Bhattarai e Dasgupta 2011]	86,00	0,890
C4.5 auto-supervisionado [Bhattarai e Dasgupta 2011]	71,58	0,825
Melhores resultados obtidos nesse trabalho		
SVM-Linear (metadados)	93,07	0,949
SVM-Linear (mensagens + metadados)	**93,75**	**0,954**

É importante notar que o SVM linear obteve desempenho superior aos métodos propostos e avaliados em outros trabalhos da literatura. Além disso, com base nos resultados obtidos, é conclusivo que tal técnica de aprendizado de máquina pode ser empregada com sucesso para auxiliar o processo de detecção automática de *blog comment spam*.

6. Conclusões e Trabalhos Futuros

Este trabalho apresentou uma análise de desempenho dos métodos de classificação mais empregados na literatura na tarefa de detecção automática de *blog comment spam*.

Os resultados dos experimentos indicaram que a maioria das técnicas avaliadas foi capaz de detectar uma taxa expressiva de comentários indesejados, independentemente do conjunto de atributos utilizado. Entretanto, uma grande quantidade de amostras legítimas foi bloqueada pelos métodos quando empregados apenas dados extraídos dos textos das mensagens. A principal razão para isso reside no fato de que muitos comentários indesejados apresentam alta similaridade com as mensagens legítimas, uma vez que são normalmente postados por pessoas. Além disso, conforme apontado por [Cormack et al. 2007], o conteúdo das mensagens por si só não fornece informações suficientes para os classificadores.

Com relação aos conjuntos de atributos analisados, ficou evidente que o emprego de atributos extraídos dos metadados das mensagens oferece melhor separabilidade entre as classes e, consequentemente, reduz o volume de mensagens legítimas bloqueadas. Nesse sentido, os melhores resultados foram obtidos quando as técnicas de classificação foram empregadas com a combinação de informações extraídas dos textos das mensagens e dos metadados dos *posts*.

Dentre os métodos avaliados, as máquinas de vetores de suporte com *kernel* linear em média obtiveram os melhores desempenhos, inclusive superando os resultados presentes na literatura e, portanto, demonstraram ser adequadas para auxiliar na detecção de *blog comment spam*.

Trabalhos futuros compreendem o estudo de formas de adaptar os métodos mais promissores para otimizar seu desempenho, além da proposição de técnicas para expansão do hiperespaço de atributos, uma vez que as mensagens originais normalmente são curtas e repletas de gírias e abreviações.

Agradecimentos

Os autores são gratos a Fapesp e CNPq pelo apoio financeiro ao desenvolvimento desse projeto.

Referências

Aha, D. W., Kibler, D., e Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1):37–66.

Almeida, T., Almeida, J., e Yamakami, A. (2011). Spam Filtering: How the Dimensionality Reduction Affects the Accuracy of Naive Bayes Classifiers. *Journal of Internet Services and Applications*, 1(3):183–200.

Bhattarai, A. e Dasgupta, D. (2011). A self-supervised approach to comment spam detection based on content analysis. *International Journal of Information Security and Privacy*, 5(1):14–32.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Chang, C.-C. e Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Trans. on Intelligent Systems and Technology*, 2:27:1–27:27.

Chu, Z., Gianvecchio, S., Koehl, A., Wang, H., e Jajodia, S. (2013). Blog or block: Detecting blog bots through behavioral biometrics. *Computer Networks*, 57(3):634–646.

Cormack, G., Hidalgo, J. G., e Sanz, E. P. (2007). Spam filtering for short messages. In *Proceedings of the 16th ACM CIKM*, pages 313–320, Lisbon, Portugal.

Frank, E. e Witten, I. (1998). Generating accurate rule sets without global optimization. In *Proceedings of the 15th ICML*, pages 144–151.

Freund, Y. e Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Proc. of the 13th ICML*, pages 148–156, Bari, Italy. Morgan Kaufmann.

Greene, W. H. (2003). *Econometric Analysis*. Prentice Hall, 5 edition.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., e Witten, I. H. (2009). The WEKA data mining software: an update. *SIGKDD Explorations Newsletter*, 11(1):10–18.

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. Prentice Hall, New York, NY, USA, 2th edition.

Kantchelian, A., Ma, J., Huang, L., Afroz, S., Joseph, A. D., e Tygar, J. D. (2012). Robust detection of comment spam using entropy rate. In *Proceedings of the 5th ACM AISec*, pages 59–69, Raleigh, United States.

Mishne, G., Carmel, D., e Lempel, R. (2005). Blocking blog spam with language model disagreement. In *Proceedings of the 1st AIRWeb*, pages 1–6, Chiba, Japan.

Mishne, G. e Glance, N. (2006). Leave a reply: An analysis of weblog comments. In *Proceedings of the 3rd WWE*, pages 1–8, Edinburgh, UK.

Page, L., Brin, S., Motwani, R., e Winograd, T. (1998). The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th WWW*, pages 161–172, Brisbane, Australia.

Quinlan, J. R. (1993). *C4.5: programs for machine learning*. Morgan Kaufmann, San Mateo, CA, USA, 1th edition.

Romero, C., Valdez, M. G., e Alanis, A. (2010). A comparative study of machine learning techniques in blog comments spam filtering. In *Proceedings of the 6th IEEE WCCI*, Barcelona, Spain.

Shin, Y., Gupta, M., e Myers, S. (2011). Prevalence and mitigation of forum spamming. In *Proceedings of the 30th IEEE INFOCOM*, pages 1–9, Shangai, China.

Wang, J., Yu, C. T., Yu, P. S., Liu, B., e Meng, W. (2012). Diversionary comments under political blog posts. In *Proceedings of the 21st ACM CIKM*, pages 1789–1793, Maui, United States.

Wu, X., Kumar, V., Quinlan, J. R., Ghosh, J., Yang, Q., Motoda, H., McLachlan, Ng, A., Liu, B., Yu, P. S., Zhou, Z.-H., Steinbach, M., Hand, D. J., e Steinberg, D. (2008). Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14(1):1–37.