UMA ABORDAGEM DE DETECÇÃO AUTOMÁTICA DE VANDALISMO NA WIKIPEDIA UTILIZANDO APRENDIZADO ASSOCIATIVO ATIVO

MARIA INÊS MUIANGA SUMBANA

UMA ABORDAGEM DE DETECÇÃO AUTOMÁTICA DE VANDALISMO NA WIKIPEDIA UTILIZANDO APRENDIZADO ASSOCIATIVO ATIVO

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Marcos Gonçalves
Co-orientador: Jussara Almeida

Belo Horizonte

Março de 2012

Sumbana, Maria Inês Muianga
S955a Uma abordagem de detecção automática de vandalismo na Wikipedia utilizando aprendizado associativo ativo / Maria Inês Muianga Sumbana.— Belo Horizonte, 2012.
xx, 44 f.: il.; 29 cm.

Dissertação (mestrado) — Universidade Federal de Minas Gerais. Departamento de Ciência da Computação.
Orientador: Marcos André Gonçalves.
Coorientadora: Jussara Marques de Almeida Gonçalves

1. Computação - Teses. 2. Recuperação da informação
– Teses. 3. Wikipedia – Teses. I.Orientador. II.Coorientadora. III. Título.

CDU 519.6*73 (043)

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

Uma abordagem de detecção automática de vandalismo
na Wikipedia utilizando aprendizado associativo ativo

MARIA INÊS MUIANGA SUMBANA

Dissertação defendida e aprovada pela banca examinadora constituída pelos Senhores:

PROF. MARCOS ANDRÉ GONÇALVES - Orientador
Departamento de Ciência da Computação - UFMG

PROFA. JUSSARA MARQUES DE ALMEIDA - Co-orientadora
Departamento de Ciência da Computação - UFMG

PROFA. MIRELLA MOURA MORO
Departamento de Ciência da Computação - UFMG

PROFA. VIVIANE PEREIRA MOREIRA
Departamento de Informática Aplicada - UFRGS

Belo Horizonte, 22 de junho de 2012.

Dedico este trabalhos aos meus filhos Maysa Edwiges Sumbana Gonçalves e Denzel Amani Sumbana Gonçalves

Agradecimentos

Em primeiro lugar agradeço a Deus que é a fonte da minha vida, a luz que me guia.

Aos meus orientadores, professores Marcos Gonçalves e Jussara de Almeida pelo acolhimento, pelos ensinamentos, pela paciência, dedicação na orientação, e pela vossa presença na minha vidas nesses últimos dois anos.

Aos professores do DCC, em especial Virgílio, Gisele, Loureiro, pelos ensinamentos transmitidos nas disciplinas por eles lecionadas.

Ao professor José Marcos e a todo pessoal do Programa de Pos-graduação do DCC, pela oportunidade que me deram de poder cursar o mestrado

Ao prof Adriano Veloso e ao Rodrigo Silva pelo apoio na realização deste trabalho.

Ao Eng. Muchanga e Dr Esselina Macome pelo apoio e voto de confiança.

Ao pessoal da Secretária da Pos-graduação, em especial a Renata e Linda, pelo apoio e pela paciência.

Ao Dr António Cipriano parafino Gonçalves, pela força, apoio e paciência em todos anos de convivência, e por me fazer entender a importância de continuar com os estudos.

Ao Dr Manuel Mangue, pelo apoio e incentivo para entrar no mestrado.

Ao Instituto Nacional de Estatística de Moçambique, em especial o presidente do INE João Loureiro, o director adjunto do Departamento de Sistemas de Informação, Tomás Bernardo, pela oportunidade que me deram de poder continuar com os estudos.

A todos os colegas do DCC que de alguma maneira contribuiram para que esse trabalho fosse realizado. Especial agradecimento a Elisabeth, Gabriel, Hasan, Allan, Thiago cardoso, Thiago Salles.

A minha mãe Marta Muianga e aos meus irmãos Fátima, Germano, Zacarias, Alcinda, Margarida, Manuel, Ana Carlota, Adriano, meus cunhados, sobrinhos em especial Jamila e Leonardo por toda paciência, apoio e compreensão nesses últimos dois anos.

Ao Langton, Dona filismina e Abu, por terem cuidado dos meus filhos durante a minha ausência.

Aos meus amigos Afonso,Atanásia, Manuela, Verónica, Ilda, Marrapucha, Ana Cristina, pela vossa amizade, apoio nos momentos bons e maus.

Aos estudantes bolseiros de Moçambique, Serafim, Elsa, João, Abdulai, Varela, Leonel, Esperança, Lúcio pela convivência nos últimos dois anos.

Um agradecimento muito especial à irmã Emília, dona Amana, Irmã Augusta, Virgínia, Viviane, Joela, Lionice, vocês são muito especiais para mim.

Aos todos os colegas do INE, em especial, Carolina Cubasse, Rute, Nilda, paulo Lipanga, Paulo Matusse, Elias, Mauro, sr Salomão, um especial agradecimento para vocês.

Resumo

A Wikipedia e outros serviços gratuitos cujo conteúdo é gerado colaborativamente têm crescido rapidamente em popularidade. No entanto, a falta de controle da edição tem feito com que esses serviços sejam vulneráveis a vários tipos de ações maliciosas como o vandalismo. Métodos de detecção de vandalismo de estado-de-arte são baseados em técnicas supervisionadas, e portanto, dependem de coleções de treinamento geralmente grandes e representativas. A construção de tais coleções depende, muitas vezes, de um esforço conjunto (*crowdsourcing*), sendo ssim caras de construir. Mais ainda, no caso específico de vandalismo, as coleções disponíveis tendem a ser muito desbalanceadas com muito poucos exemplos de vandalismo, o que afeta o processo da classificação. Visando diminuir o custo da construção das coleções representativas para esse problema, apresentamos uma nova técnica de seleção ativa juntamente com um algoritmo de classificação associativa sob-demanda para a detecção de vandalismo na Wikipédia. Primeiro mostramos que a classificação associativa reforçada por uma técnica simples de balanceamento para a construção do conjunto de treinamento supera classificadores de estado-de-arte como SVM e kNN, e é competitivo com os melhores resultados da competição CLEF em detecção de vandalismo na Wikipedia. Além disso, através da aplicação da abordagem de seleção ativa, fomos capazes de reduzir a necessidade de treinamento em quase 96% com apenas um pequeno impacto sobre os resultados da detecção, tornando assim a nossa solução muito prática para cenários reais.

Abstract

Wikipedia and other free editing services for collaboratively generated content have quickly grown in popularity. However, the lack of editing control has made these services vulnerable to various types of malicious actions such as vandalism. State-of-the-art vandalism detection methods are based on supervised techniques, and thus rely on the availability of large and representative training collections. Building such collections, often with the help of crowdsourcing, is quite costly, as it has to deal with a natural skew towards very few vandalism examples in the available data and dynamic patterns. Aiming at reducing the cost of building such collections, we present a new active sampling technique coupled with an on-demand associative classification algorithm for Wikipedia vandalism detection. We first show that the associative classification enhanced with a simple undersampling technique for building the training set outperforms state-of-the-art classifiers such as SVMs and kNNs, and is competitive with the best results of the CLEF competition on Wikipedia vandalism detection. Furthermore, by applying the active sampling approach, we are able to reduce the need for training in almost 96% with only a small impact on detection results, thus making our solution very practical for real scenarios.

Keywords: vandalism detection, active sampling, associative classification.

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1

Introdução

Ao longo dos anos, a Web tornou-se a aplicação mais popular e a mais usada pelas pessoas, para vários fins. Com o passar do tempo, a Web vem sofrendo transformações, se tornando um ambiente cada vez mais social. É nesse contexto que surge a Web 2.0 [Wikipedia, 2011b] a qual disponibiliza para os usuários uma experiência de uso semelhante à de aplicativos para desktop, proporcionando maior comunicação e aplicações interativas que facilitam aos usuários compartilharem informações. Além disso, a Web 2.0 permite aos usuários escolha livre para interagir e colaborar com os outros usuários.

Um dos serviços mais disponibilizados pela Web 2.0 são sites cujo conteúdo é gerado pelos próprios usuários. Um dos exemplos desses serviços são sites de compartilhamento de vídeos. Dentre esses sites, o YouTube é o que mais se destaca, tendo sido responsável pelo fornecimento de 44,8% do total de vídeos assistidos, a nível mundial, em fevereiro de 2012 [comScore, 2012]. Outro exemplo, são sites cujo conteúdo é formado por artigos escritos colaborativamente pelos usuários. Tais serviços permitem aos usuários finais não apenas a consulta como também a criação e revisão livre deste conteúdo. O mais popular desses serviços, Wikia[1], cresceu de modo vertiginoso, em um espaço de tempo muito curto, contendo mais de quatro milhões de páginas de conteúdo bastante rico. Outro exemplo de como as comunidades podem produzir conteúdo colaborativo [Hasan Dalip et al., 2009; Wang & McKeown, 2010] é a Wikipédia[2]. Esta enciclopédia *on-line* pode ser considerada hoje uma das maiores fontes de informação na Internet, com mais de vinte e cinco milhões de artigos escritos em dezenas de línguas diferentes [Wikipedia, 2012b].

No entanto, esta grande quantidade de informação disponibilizada de forma de-

[1] www.wikia.com
[2] www.Wikipédia.com

mocrática e praticamente sem nenhum controle, abre oportunidades para "comportamentos maliciosos"de pessoas que tentam explorar estes serviços para o benefício próprio (por exemplo, inclusão da publicidade) ou com a intenção de degradar a integridade e confiabilidade do sistema (por exemplo, por meio da inclusão do material poluído como pornografia ou informações explicitamente incorretas). Esse comportamento não-cooperativo, caracterizado pela inclusão de conteúdo poluído incorre em custos adicionais para os administradores de sistema, afeta a qualidade dos serviços de informação (por exemplo, a busca), e pôe em risco a paciência e satisfação dos usuários, já que estes não podem identificar facilmente o conteúdo poluído sem ter contato com este conteúdo.

1.1 Descrição Detalhada do problema

A Wikipédia, é uma enciclopédia on line, livre, onde qualquer um pode editar, modificar ou revisar artigos, desde que os direitos de cópia e modificação sejam preservados [Belani, 2010; Wang & McKeown, 2010]. Por ser livre, também, tem sido alvo de atos maliciosos, praticados por usuários não-cooperativos, tais como o vandalismo, guerras de revisão, dentre outras [Potthast et al., 2010]. Na Wikipédia, o vandalismo é, explicitamente, definido como qualquer adição, remoção ou alteração do conteúdo na tentativa deliberada de comprometer a integridade do sistema[3]. Exemplos típicos de atos de vandalismo são a adição de obscenidades, informações claramente irrelevantes e humor "negro"em um artigo (e.g., *hey look at me, I just vandalize a page LOL*) [Wikipedia, 2011a], remoção ilegítima de páginas, e inserção de conteúdo sem sentido em uma página (por exemplo, *gggfebdgs&%#*88*) [Chin et al., 2010].

Várias medidas para combater o vandalismo foram adotadas na Wikipédia tais como: limitar os privilégios de usuários anônimos; adotar a validação do artigo e a utilização de um "filtro de abuso"que controla as atividades do usuário reagindo automaticamente a comportamentos suspeitos [Belani, 2010]. Porém estas medidas não são ainda suficientes para reduzir o impacto e o crescimento destes atos. Além disso, a detecção de vandalismo é praticamente feita de forma manual por voluntários [Wang & McKeown, 2010], o que requer muito esforço por parte destes. Para detecção de alguns atos de vandalismo, como por exemplo as palavras vulgares é utilizada uma lista de expressões regulares. Esta lista é construída manualmente por alguns editores. A detecção manual tem desvantagens óbvias em termos de custo e escalabilidade, devido ao tamanho atual e taxa de crescimento da Wikipédia. Estas desvantagens motivaram

[3]http://en.Wikipédia.org/wiki/Wikipédia:Vandalism

alguns esforços recentes no sentido de se desenvolver técnicas de detecção automática do vandalismo [Potthast et al., 2008; Smets et al., 2008]. Porém, sendo geralmente supervisionadas, estas técnicas dependem normalmente de grandes coleções especializadas, construídas a partir de um esforço conjunto, (o *crowdsourcing*) [Potthast, 2010] para servirem de dados de treinamento. Além disso, esta tarefa de aprendizagem, é desafiadora dado o fato de que a maioria dos artigos são, na verdade, não vandalizados e, assim, as coleções de treinamento disponíveis tendem a ser muito desequilibradas, com muito poucos exemplos de vandalismo e muitos exemplos de artigos não vandalizados. Este desequilíbrio pode afetar o processo de aprendizagem e a eficácia na detecção do vandalismo. Em problemas de classificação, onde a distribuição de classes é desequilibrada, muitos algoritmos de aprendizado de máquina tendem a classificar erroneamente as instâncias da classe menor. Além disso, alguns algoritmos de aprendizagem de máquina tendem a tratar amostras de uma classe minoritária como ruído.

Mais ainda, dada a dificuldade mencionada de encontrar artigos vandalizados para o treinamento, a construção de novos conjuntos de treinamento é difícil e cara. Por exemplo, para a construção das colecções utilizadas na competição CLEF[4] - *Conference and Labs of the Evaluation Forum* foi necessário um grande esforço conjunto. Esse custo torna-se um fator limitante da aplicação prática dessas técnicas se considerarmos que em situações reais, as ações de vandalismo evoluem e novos padrões surgem para contornar os mecanismos de detecção já existentes ou para explorar os novos atributos da aplicação (por exemplo, as listas de artigos mais populares). Assim, existe uma necessidade constante de se reconstruir os conjunto de treinamento, com atualizações de exemplos, para que os novos padrões sejam introduzidos no modelo de detecção.

1.1.1 Contribuições

A principal contribuição deste trabalho é proposta de uma técnica para a redução do custo da construção do conjunto de treinamento necessário para a tarefa de detecção do vandalismo na Wikipédia. Para esse objetivo, utilizamos uma técnica de seleção ativa para reduzir a necessidade do conjunto de treinamento para a detecção do vandalismo na Wikipédia, **o SADV (Seleção Ativa para Detecção do Vandalismo)** e um algoritmo de classificação associativa sob demanda para detecção de vandalismo. Para nosso conhecimento é a primeira vez que está técnica é utilizada para a detecção do vandalismo na Wikipédia. Também propusemos uma técnica de *undersampling* muito simples para balancear os dados de treinamento. Para a avaliação da nossa estratégia utilizamos o conjunto de dados do PAN-WVC-10 [Javanmardi et al., 2011; Potthast

[4]http://pan.webis.de

et al., 2010], que consiste em mais de 30 mil revisões, de 28 mil artigos, dos quais apenas cerca de 2,4 mil revisões são de vandalismo.

1.2 Organização da Dissertação

Esta dissertação está organizado da seguinte forma: No Capítulo 2 são apresentados os trabalhos relacionados. A metodologia proposta e os classificadores são apresentados no Capítulo 3. No Capítulo 4 são detalhados os experimentos realizados e a metodologia utilizada para sua realização. Os resultados experimentais são apresentados no Capítulo 5. Por fim no Capítulo 6 descrevemos as conclusões e trabalhos futuros.

1.3 Publicações

O seguinte artigo, publicado na International Conference on Theory and Practice of Digital Libraries (TPDL 2012), é uma das principais contribuições desta dissertação:

- Sumbana, M., Gonçalves, M. A., Almeida, J. M., Silva, R., Veloso, A. Automatic Vandalism Detection in Wikipedia With Active Associative Classification

Capítulo 2

Trabalhos Relacionados

2.1 Deteção de Comportamentos Maliciosos

Diferentes tipos de comportamentos maliciosos foram identificados em várias aplicações da Web 2.0, motivando vários esforços para a construção de mecanismos para a sua detecção automática. A maioria desses mecanismos depende de algoritmos de classificação para distinguir entre os usuários mal-intencionados e os legítimos. Por exemplo, com base em um estudo realizado por pesquisadores de segurança para observar as atividades maliciosas dos hackers, em [Lee et al., 2010] os autores propuseram uma estratégia para detectar spammers sociais novos e emergentes em sistemas sociais online. Para tal, eles introduziram no sistema *honeypots* sociais para monitorar e registrar comportamentos dos usuários com alta probabilidade de serem spammers. Com base nas informações armazenadas e no perfil do usuário criaram um conjunto de treinamento e aplicando um algoritmos de classificação para a construção do modelo puderam distinguir entre os usuários legítimos e spammers sociais. Este estudo foi feito em duas redes sociais Myspace e Twitter. Em [Benevenuto et al., 2009], os autores concentraram-se na detecção automática de usuários que introduzem conteúdo poluído no YouTube, a quem eles definiram como spammers ou promotores de conteúdo, dependendo da intenção do usuário. Eles definiram vários atributos do usuário com base no seu perfil, no conteúdo compartilhado e nas interações com os outros usuários. Explorando esses recursos e utilizando um método de classificação supervisionada, puderam classificar com eficácia os usuários em spammer, legítimo ou promotor com eficácia.

2.2 Vandalismo na Wikipédia

Em trabalhos anteriores foram descritas algumas características particulares através das quais uma revisão é reconhecida como vandalismo [Potthast et al., 2008]. Por exemplo o uso freqüente de pronomes pessoais, quando um vândalo expressa sua opinião pessoal, é um dos pontos característicos numa revisão de vandalismo. Outro ponto comum, é que nas revisões vandalizadas são geralmente introduzidos textos que contradizem o senso comum ou expressões incorretamente formadas em relação à língua em que o artigo foi escrito. Estas expressões incluem palavras absurdas, vulgares ou uma seqüência de caracteres sem sentido. Um texto ou expressão duplicado dentro de um artigo, também pode ser identificado como um ato de vandalismo. Outra característica que é muito comum nas revisões vandalizadas é a presença de letras maiúsculas em lugares impróprios ou repetição de caracteres. Com base nessas ações, alguns trabalhos de pesquisa definiram diferentes tipos de vandalismo na Wikipédia, [Chin et al., 2010; Priedhorsky et al., 2007; Potthast et al., 2008; Viégas et al., 2004]. A seguir passamos a descrever alguns tipos de vandalismo definidos nesses trabalhos, que comumente ocorrem na Wikipédia:

- *Blanking*: Consiste na remoção de partes significativas do conteúdo da página, ou apagar todo conteúdo e inserir um texto sem sentido. Por exemplo apagar o conteúdo da página e substituir com palavras repetitivas.

Figura 2.1. Exemplo de vandalismo do tipo *Blanking*

- *Edit summary vandalism*: neste tipo de vandalismo, o vândalo edita resumos com palavras ofensivas na tentativa de deixar algo que não será removido facil-

mente. Geralmente, para se reverter este tipo de vandalismo recorre-se a ação do administrador.

- *Hidden vandalism*: é um tipo de vandalismo que só é visível durante a edição do artigo, mas não é visível no artigo final. Inclue links, mensagens maliciosas e ofensivas ou *spam* em comentários ocultos para ser visto pelo editor.
- *Image vandalism*: consiste em inserir imagens explícitas de uma forma inadequada ou substituir uma imagem existente por outra irrelevante com objetivo de danificar a página.
- *Link vandalism*: é a inserção ou substituição de links internos ou externos que levam para outras páginas com conteúdo irrelevante para o artigo.
- *Page creation, illegitimate*: O usuário cria páginas com intenção maliciosa. Tais páginas incluem, por exemplo, publicidade ou artigos escritos para depreciar o assunto.
- *Page lengthening*: neste tipo de vandalismo, o usuário insere grandes quantidades de infomação (medido em bytes) com intenção de tornar o tempo de carregamento da página muito longo, ou fazer com que a página seja impossível de abrir.
- *Page-move vandalism*: consiste em mudar o nome da página por um outro irrelevante ou inadequado. Este tipo de ato de vandalismo é somente feito por usuários confirmados.
- *Silly vandalism*: é a inserção de conteúdo obsceno ou criação de páginas sem sentido que incluem comentários que prejudicam a qualidade do artigo.
- *Sneaky vandalism ou Misinformation*: é um tipo de ato de vandalismo difícil de detectar, porque as mudanças que o usuário faz são imperceptíveis. Consiste na alteração da informação existente por informações falsas, tais como trocar o nome, datas, trocar alguns dígitos num número ou editar duas revisões de vandalismo e reverter uma, usando contas ou endereços IP diferentes.

2.3 Detecção de Vandalismo na Wikipédia

Em um dos primeiros estudos em que se abordou o problema de detecção automática de vandalismo, os autores definiram o problema como uma tarefa de classificação binária [Potthast et al., 2008]. Analisando o conteúdo e as categorias das revisões

Figura 2.2. Exemplo de vandalismo do tipo *Sneak*

identificaram alguns tipos de vandalismo (como foi reconhecido por seres humanos) e definiram os atributos necessários para identificá-los. Com base em um classificador baseado em regressão logística eles foram capazes de classificar novos exemplos de vandalismo com certa acurácia. Mais recentemente, os autores de [Chin et al., 2010] com base em uma taxonomia das revisões da Wikipédia construída a partir de ações primárias, descritas na seção 2.2, definiram e identificaram sete tipos de vandalismo. Eles aplicaram *Statistical Language Models* sobre a diferença entre duas revisões consecutivas para construir entradas para o classificador, que por sua vez foram utilizados para detectar os casos de vandalismo. Embora eles se referiram ao método como baseado em aprendizado ativa, esta abordagem não pode ser de fato considerada como tal no sentido de que em cada iteração o método proposto não escolhe a melhor nova instância para rotular. Ao invés, um classificador é treinado com dados do [Potthast et al., 2008], e em cada iteração, eles simplesmente ordenam as instâncias do conjunto de dados pela probabilidade de uma revisão ser vandalismo, e as adicionam as 50 primeiras instâncias junto com informações de revisão. Devido à complexidade do método, os autores trabalharam com revisões de apenas dois artigos da Wikipédia, Microsoft e Lincoln.

A maioria dos métodos de detecção de vandalismo na Wikipédia que constam na lista de vencedores da competição CLEF 2010 usaram variações da árvore de decisão nos seus detectores tais como *random forests, alternating decision trees, naive Bayes decision trees, and C4.5 decision trees* [Potthast et al., 2010]. Por exemplo, o detector vencedor usou *random forests* de 1000 árvores com 5 atributos aleatórios

cada. Todos eles usaram o conjunto de dados PAN-WVC-10 (descrito em detalhes na Seção 4.1) dividido praticamente ao meio, sendo uma metade para treinamento e outra para teste. Os promotores da competição CLEF-2010 [Potthast et al., 2010] também combinaram as previsões dos oito melhores resultados usando *random forests* como um meta-classificador, o que levou a ganhos consideráveis de eficiência de detecção. Aqui não estamos preocupados com a combinação de métodos, embora o nosso método proposto poderia ser parte desta abordagem.

Em um trabalho recente, [Javanmardi et al., 2011], descreveu um modelo para a detecção de vandalismo em UGC (*user generated content*). Utilizando como base a coleção PAN-WVC-10, os autores extraíram vários novos atributos e os organizaram em quatro grupos, a saber: atributos do usuário, atributos textuais, atributos de metadados e atributos de modelo de linguagem. Em seguida, eles aplicaram a técnica de regularização *Lasso* para reduzir o número de atributos dentro dos grupos e considerando todos ao mesmo tempo, mantendo apenas os mais discriminativos, e utilizaram *random forests* para aprender o modelo de classificação.

Assim como nos estudos anteriores, também neste presente trabalho utilizamos as técnicas de aprendizado de máquina para desenvolver uma técnica de detecção automática de vandalismo na Wikipédia. Porém, comparando com os trabalhos anteriores, nós reduzimos a necessidade de treino utilizando uma técnica de seleção ativa, e além disso para melhorar os resultados da classificação aplicamos sobre os dados de treinamento uma solução para o balancear estes dados. Para tal, propomos uma abordagem de detecção de vandalismo baseada em classificação associativa e em aprendizado ativo que descrevemos no capítulo 3.

2.4 Aprendizado Supervisionado

O aprendizado de máquina é um ramo da inteligência artificial voltada ao desenvolvimento de algoritmos que permitem aos computadores evoluírem comportamentos baseados em dados empíricos [Wikipedia, 2012a]. O foco principal do aprendizado de máquina é aprender a reconhecer padrões complexos e tomar decisões inteligentes com base nos dados. O processo de aprendizado consiste em extrair características de interesse (atributos) a partir dos exemplos (dados) e com base nessas características ser capaz de classificar novos exemplos. O objetivo da classificação é prever a classe de um objeto, representado por um vetor de atributos. Para a realização da tarefa da classificação podem ser utilizados como dados de entrada, informações fornecidas por humanos, tais como o número de classes, a classe do objeto (aprendizado super-

visionado) ou apenas a própria coleção dos objetos sem rotulação (aprendizado não supervisionado).

Aprendizado supervisionado utiliza como entrada um conjunto de dados constituído por objetos previamente rotulados (os dados de treinamento). Cada objeto consiste em um conjunto de atributos (geralmente númericos) e um atributo que define a classe que o objeto pertence. Neste tipo de aprendizado, o objetivo da classificação é descobrir o relacionamento entre os atributos e a classe utilizando exemplos cuja classe é conhecida para que posteriormente esses atributos sejam utilizados para prever a classe de um objeto [Pappa, 2002] cuja classe não é conhecida.

Antes de se realizar a tarefa de classificação, é necessário dividir os dados de treinamento disponíveis em conjunto de treinamento e conjunto de teste. O conjunto de treinamento é utilizado pelo classificador para criar o modelo. Para medir o desempenho do modelo criado é utilizado o conjunto de teste, isto é, o classificador utiliza o modelo para prever as classes dos exemplo do conjunto de teste. Durante a classificação, as classes dos exemplos de teste não são consideradas pelo classificador. Após a classificação, as classes dos exemplos de teste previstas pelo classificador são então comparadas com as classes reais previstas pelos humanos. Se a classe prevista for igual a classe real, então diz-se que o classificador classificou corretamente. Existe uma grande variedade de algoritmos de aprendizado supervisionados disponíveis, cada um com seus pontos fortes e fracos, porém não há um único algoritmo de aprendizado que funciona melhor para todos os problemas de aprendizado supervisionado. Nesta dissertação apresentamos três deles, *Suport Vector Machine* - SVM e *K-Nearest Neighbor* - kNN e a Classificação Associativa que foi utilizada neste trabalho.

2.5 Classificadores

2.5.1 Suport Vector Machine - SVM

O SVM é um método de classificação definido sobre um espaço vetorial n-dimensional [Joachims, 1998]. O objetivo é encontrar um hiperplano que separa, com uma margem máxima, os dados de treinamento em duas porções de um espaço n-dimensional. A margem é obtida obtida pela distância entre o hiperplano e os vetores que estão mais próximos ao hiperplano(os vetores suportes).

A Figura 2.3 mostra um espaço de atributos linearmente separavéis para um conjunto de treinamento bi-dimensioal. A linha escura representa o hiperplano de separação entre as classes e é representada pela função $f(\overrightarrow{x}) = (\overrightarrow{w} \bullet \overrightarrow{x}) + b$. Um hiperplano é considerado de margem máxima se a distância entre os vetores mais

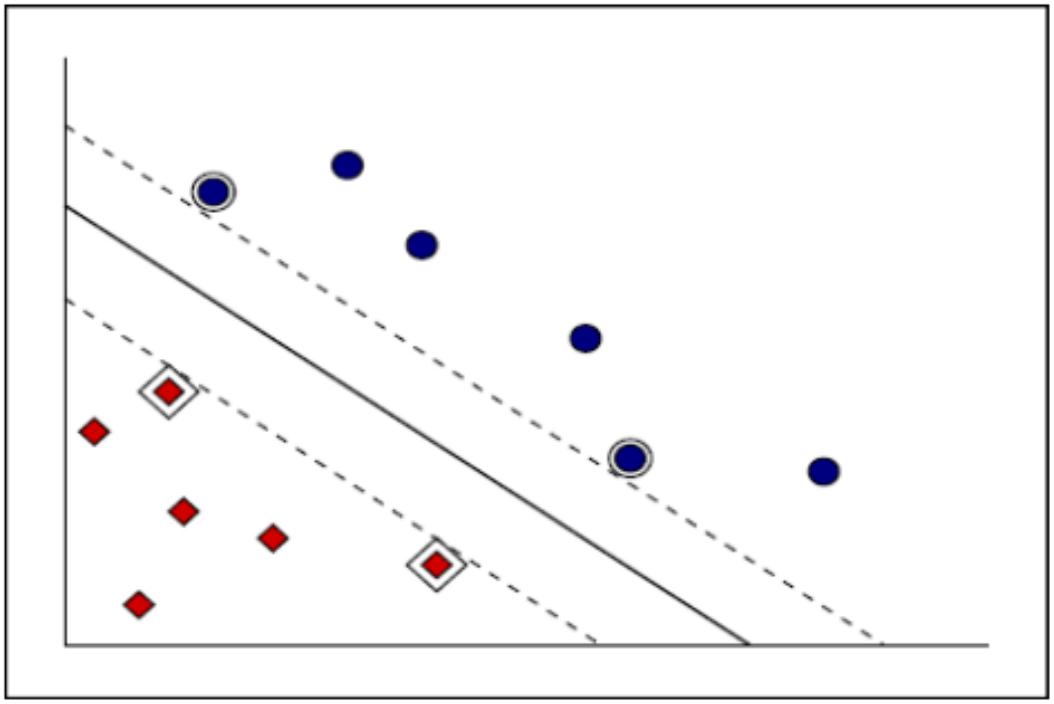

Figura 2.3. Espaço de atributos de duas classes linearmente separáveis

próximos do hiperplano é máxima [Gonalves, 2012]. Se o conjunto de treinamento é linearmente separável, então o hiperplano de separação é o hiperplano de margem máxima (ou hiperplano ótimo) e é descrito pela expressão $(\overrightarrow{w} \bullet \overrightarrow{x}) + b = 0$, onde $\overrightarrow{x}$, é um ponto arbitrário que representa o objeto a ser classificado, e $\overrightarrow{w}$ e b são o vetor de pesos e bias respectivamente.

Seja $\mathcal{D}$ um conjunto de exemplos de treinamento $x_i \in \mathcal{R}^{\mathcal{N}}$, onde $i = 1, 2, \ldots, n$. Cada exemplo x_i pertence a uma das duas classes -1,+1. Supomos que existe um hiperplano que separa os exemplos negativos dos positivos e os exemplos sobre o hiperplano satifazem a condição $(\overrightarrow{w} \bullet \overrightarrow{x}) + b = 0$. Uma classificação linear consiste em determinar a função f: $X \subseteq \mathcal{R}^{\mathcal{N}} \rightarrow \mathcal{R}^{\mathcal{N}}$ que atribui a classe $+1$ se $f(\overrightarrow{x}) \geq 0$ e -1 caso contrário [Gonalves, 2012].e podemos representá-la da seguinte forma:

$$f(\overrightarrow{x}) = (\overrightarrow{w} \bullet \overrightarrow{x}) + b = \sum_{i=1}^{n} \overrightarrow{w_i}\overrightarrow{x_i} + b \tag{2.1}$$

Os valores de $\overrightarrow{w}$ e b são obtidos pelo processo de aprendizado a partir dos dados de entrada. Esta função determina a posição do vetor x em relação ao hiperplano.

Seja d^+ a menor distância entre o hiperplano de separação e os pontos da fronteira da classe $+1$, e d^-, a menor distância entre o hiperplano de separação e os pontos mais próximos da fronteira da classe -1. A margem do hiperplano é dada como $(d^+ + d^-)$

e representa o quanto o hiperplano pode ser movido, assumindo que todos os dados de treinamento satisfazem as restrições a seguir,

$$\overrightarrow{w} \bullet \overrightarrow{x_i} + b \ \geq \ +1, \tag{2.2}$$

para $y_i = +1$

$$\overrightarrow{w} \bullet \overrightarrow{x_i} + b \ \leq \ -1, \tag{2.3}$$

para $y_i = -1$

Então o SVM procura encontrar o hiperplano que separa os dados e treinamento e que tem o vetor de menor peso. Este hiperplano pode ser encontrado resolvendo o problema de otimização [Joachims, 1998]:

minimizar $||\overrightarrow{w}||$ de modo que

$$y_1(\overrightarrow{w} \bullet \overrightarrow{x_i} + b) - 1 \ \geq \ 0 \tag{2.4}$$

para $\forall i = 1, 2, \ldots, n$

Utilizando os multiplicadores de Lagrange podemos traduzir o problema de minimização para o problema de minimização quadrática [Joachims, 1998]:

minimizar:

$$\sum_{i=1}^{n} \alpha_i + \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i x_j \tag{2.5}$$

tal que

$$\sum_{i=1}^{n} \alpha_i y_i = 0 \tag{2.6}$$

para $\forall i : \alpha_i \ \geq \ 0$

O resultado do processo de otimização é um conjunto de coeficientes α_i^* para qual a expressão 2.5 é mínima Estes coeficientes podem ser utilizados para construir o hiperplano de separação [Joachims, 1998] a partir as equações 2.4 e 2.5. SVMs foram originalmente projetados para classificação binária, mas podem ser estendidos para várias classes usando várias estratégias como por exemplo utilizar diferentes classificadores para aprender cada classe e para tomar a decisão final comparam-se os resultados de cada classificador utilizando um esquema de votação. Para os nossos experimentos, utilizamos o libSVM [chung Chang & Lin, 2001], uma implementação de SVM que fornece uma série de facilidades,tais como a normalização de atributos numéricos e a busca pelos melhores parâmetros de classificação utilizado o conjunto de treinamento.

2.5.2 K-Nearest Neighbor - KNN

KNN é um método de classificação de objetos baseado nos exemplos de treinamento mais próximos ao objeto a ser classificado. É um classificador do tipo *postergado*, pois a priori não constrói um modelo de classificação. Os exemplo de treinamento consistem em vetores n-dimensionais e a classe correspondente no espaço de atributos [Cunningham & Delany, 2007].

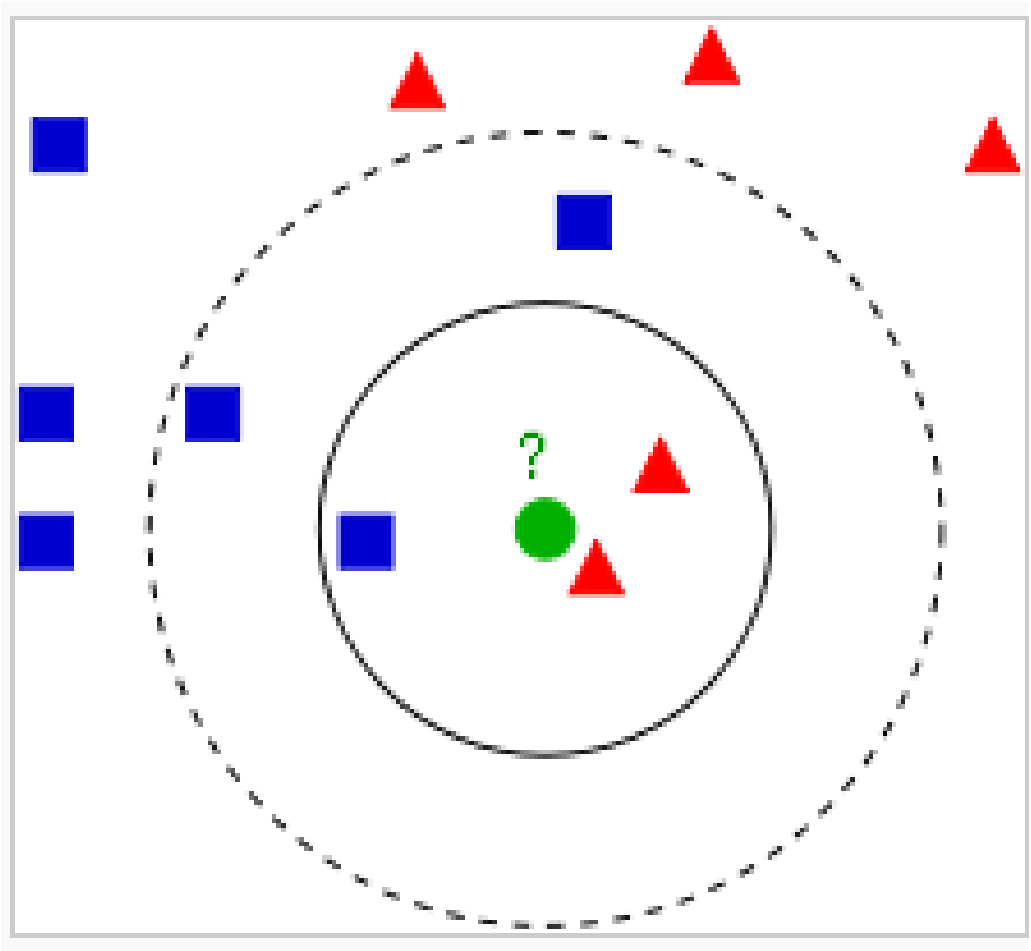

Figura 2.4. Exemplo da classificação KNN

A Figura 2.4, mostra o exemplo da classificação com KNN. O círculo verde representa o objeto a ser classificado com base no vizinho representado pelo quadrado azul ou pelo vizinho representado pelo triangulo vermelho. Se k=3, ao exemplo será atribuído à classe dos triângulos vermelhos, pois existem dois triângulos e apenas um quadrado na vizinhança do objeto.

A classificação no KNN consiste em duas fases, a determinação do vizinhos mais próximos do objeto a ser classificado e a determinação da classe do objeto com base nas

características dos vizinhos, isto é, o objeto de teste é classificado com base no voto da maioria dos vizinhos mais próximos e é atribuído a classe dos vizinhos mais semelhantes à ele. Para tal são utilizadas métricas de distância para o cálculo da similaridade entre os vizinhos, entre elas a métrica da distância Euclidiana e a similaridade do cosseno. A função do voto é utilizada para atribuir a pontuação a cada objeto de teste e é dada pela fórmula:

$$S_{d,c_i} = \sum_{d_t \in \mathcal{N}_k(d)} Similarity(d, d_t) f(c_i, d_t) \tag{2.7}$$

Onde $\mathcal{N}_k(d)$ são os k vizinhos mais próximos do objeto a ser classificado no conjunto de treinamento e $f(c_i, d_t)$ é a função que retorna o valor 1 se o objeto de treinamento d_t pertence a classe c_i e 0 caso contrário. O classificador atribui ao objeto d, a classe c_i com maior pontuação. Para os nossos experimentos utilizamos a distância euclidiana para o cálcula da medida de similardade e k=5.

2.5.3 Classificação Associativa

Segundo [Agrawal et al., 1993], o problema de regras de associação pode ser definido da seguinte forma: Seja $\mathcal{I}$=$\{i_1, i_2, \ldots, i_n\}$ um conjunto de n atributos designados por *itens*, e seja $\mathcal{D}$=$\{t_1, t_2, \ldots, t_k\}$, um conjunto de transações, identificadas por um *id* num banco de dados. Seja $\mathcal{X}$ um subconjunto de $\mathcal{I}$, diz-se que a transação t satisfaz $\mathcal{X}$ se para todos os *itens* i_k em $\mathcal{X}$, $t_k = 1$, isto é cada *item* i_k em $\mathcal{X}$ foi comprado através de uma transação t_k.

Seja $\mathcal{X}$=$\{x_1, x_2, \ldots, x_n\}$ e $\mathcal{Y}$=$\{y_1, y_2, \ldots, y_m\}$ conjuntos de *itens* tal que $x_i \neq y_i$ para todo i e j e $\mathcal{X}, \mathcal{Y} \subseteq \mathcal{I}$. Então uma regra de associação é uma regra do tipo da forma $\mathcal{X} \rightarrow \mathcal{Y}$, onde $\mathcal{X} \cup \mathcal{Y}$ é o conjunto de *itens* comprados pelo cliente. Esta associação indica que se um cliente compra $\mathcal{X}$ é provável que ele compre também $\mathcal{Y}$.

Duas medidas importantes são utilizadas para quantificar a qualidade da associação, o suporte e a confiança. O suporte de $\mathcal{X} \rightarrow \mathcal{Y}$, denota-se por $\sigma(\mathcal{X} \rightarrow \mathcal{Y})$ e a percentagem das transações que contêm todos os *itens* de $\mathcal{X} \cup \mathcal{Y}$. Se o suporte for baixo significa que não existe evidência forte de que os *itens* no conjunto $\mathcal{X} \cup \mathcal{Y}$ ocorrem juntos. A confiança de $\mathcal{X} \rightarrow \mathcal{Y}$, denota-se por $\theta(\mathcal{X} \rightarrow \mathcal{Y})$ é a probabilidade condicional de que *itens* no $\mathcal{Y}$ sejam comprados dado que os *itens* no $\mathcal{X}$ são comprados pelo cliente.

Capítulo 3

Método de Detecção do Vandalismo com Seleção Ativa

3.1 Detecção do Vandalismo

Assim como nos trabalho anteriores, [Potthast et al., 2008; Javanmardi et al., 2011], no nosso trabalho também abordamos a tarefa de detecção de vandalismo na Wikipedia como um problema de classificação binária. Dada uma revisão e, introduzida por um usuário em um dado artigo, nosso objetivo é detectar automaticamente, se a revisão e é um ato de vandalismo ou não, no último caso referido como regular.

Mais formalmente, podemos definir a tarefa de detecção de vandalismo na Wikipedia da seguinte maneira: Seja $\mathcal{D}$, a coleção de treinamento e $\mathcal{T}$ a coleção de teste, definidas como entradas. $\mathcal{D}$ consiste num conjunto de registros com formato $< e, l >$, onde e é uma revisão e l o rótulo que identifica a classe da revisão e, isto é regular or vandalismo, (l é igual a 0 ou 1 respectivamente). Cada revisão e é representada como uma lista de m valores dos atributos $\{f_1, f_2, \ldots, f_m\}$. Os atributos considerados neste trabalho são descritos no Capítulo 4. A coleção de treinamento $\mathcal{D}$ é usada para aprender o modelo de detecção de vandalismo, $\mathcal{M}$, que relaciona os atributos de uma revisão às classes correspondentes. A coleção de teste $\mathcal{T}$ consiste em registros com revisões não rotuladas (isto é, $< e, ? >$). O modelo de detecção de vandalismo, $\mathcal{M}$, é usado para prever a classe de cada revisão no conjunto $\mathcal{T}$.

Nós propomos um mecanismo de detecção de vandalismo que emprega um método de classificação associativa sob demanda, para aprender $\mathcal{M}$, acoplado com uma estratégia de seleção ativa (descrito na Seção 3.2.1) para reduzir o custo de construção da coleção de treinamento $\mathcal{D}$. O classificador e a técnica de seleção foram propostos

em trabalhos anteriores e aplicados em outros contextos [Veloso et al., 2006, 2008], particularmente na classificação de documentos. Para o nosso conhecimento, esta é a primeira vez que este método é aplicado na detecção de vandalismo na Wikipedia.

Neste capítulo, primeiro descrevemos o método de classificação utilizado neste trabalho,(Secção 3.1.1) e em seguida o método e a estratégia de seleção ativa projetada para reduzir o custo de construção da coleção de treinamento (3.2).

3.1.1 Classificador Associativo sob Demanda

Neste trabalho usamos o LAC - Lazy Associative Classifier [Veloso et al., 2006], uma variação postergada de um classificador associativo, para atribuir cada revisão na coleção de teste $\mathcal{T}$ à classe de vandalismo ou revisão regular. O LAC explora o fato de que geralmente há fortes associações entre os valores dos atributos e as classes. Tais associações são freqüentemente encobertas nos dados de treinamento $\mathcal{D}$ e quando são descobertas revelam aspectos muito importantes que podem ser usadas para prever classe de novos elementos (elementos do conjunto $\mathcal{T}$). Estas associações são expressas através de regras da forma $\mathcal{X} \rightarrow k$, indicando a associação entre o conjunto dos valores de atributos $\mathcal{X}$ e a classe k. O LAC aprende o modelo de classificação, $\mathcal{M}$, a partir das regras de associação extraídas do conjunto de treinamento, $\mathcal{D}$. Seja $\mathcal{R}$ um conjunto arbitrário de regras, $\mathcal{R}_k$ um subconjunto de $\mathcal{R}$ composta por regras $\mathcal{X} \rightarrow k$. No LAC o processo de aprendizado é dividido em duas fases principais: extração de regras sob-demanda e a previsão da classe, descritas a seguir.

3.1.1.1 Extração de Regras sob-Demanda

Tipicamente, a tarefa de extração de regras a partir de $\mathcal{D}$ é direcionada por um limiar de suporte mínimo σ_{min}, isto é, as regras que ocorrem pelos menos σ_{min} vezes no conjunto $\mathcal{D}$ são selecionadas e usadas para construir o modelo de classificação, $\mathcal{M}$. A eficiência e a eficácia desta abordagem dependem fortemente da escolha do σ_{min}. A escolha de um valor muito baixo do σ_{min} implica na geração de um número elevado de regras extraídas do conjunto $\mathcal{D}$ o que incorre em um custo computacional muito alto.

Além disso, note que a regra $\mathcal{X} \rightarrow k$ é uma regra útil para prever a classe de uma revisão $e \in \mathcal{T}$ se e contém todos os valores de atributos em $\mathcal{X}$. Assim um valor muito pequeno de σ_{min}, pode também levar a um grande número de regras que serão pouco usadas na classificação das revisões em $\mathcal{T}$. Por outro lado se escolhermos um número muito elevado de σ_{min}, regras importantes podem se perder, e isso pode prejudicar significativamente a eficácia da classificação. Para reduzir o custo de extração das regras de associação no conjunto $\mathcal{D}$ de modo a evitar a perda de regras importantes,

o LAC processa a extração de regras sob demanda [Veloso et al., 2008]. Note que o processo de extração de regras é realizado no momento da classificação. O LAC projeta o espaço de busca das regras de acordo com as informações contidas nas revisões do conjunto de teste, $\mathcal{T}$, para permitir que a extração de regras seja eficiente. Ele projeta/filtra o conjunto de treinamento de acordo com os valores dos atributos da revisão $e \in \mathcal{T}$, e extrai regras para o conjunto de treinamento projetado, denominado $\mathcal{D}^e$. Isso garante que somente regras que carregam informações sobre a revisão e são extraídas do conjunto de treinamento, limitando drasticamente o número possível de regras.

3.1.1.2 Previsão da Classe

Algumas regras são mais fortes que as outras e para ordená-las é usada uma estatística chamada confiança de uma associação [Agrawal et al., 1993] , denotada por $\theta(\mathcal{X} \rightarrow k)$, representando a força da associação entre $\mathcal{X}$ e k. A confiança é estimada pela probabilidade condicional de k ser a classe da revisão e dado que $\mathcal{X} \subseteq e$. O LAC prevê a classe da revisão $e \in \mathcal{T}$ através da combinação das confianças de todas as regras $\mathcal{X} \rightarrow k$ tal que $\mathcal{X}$ contém valores dos atributos que coincidem com os da revisão e. Mais especificamente, seja $\mathcal{R}_k^e$ conjunto de regras que predizem a classe da revisão e como k, extraídas de $\mathcal{D}$. $\mathcal{R}_k^e$ é interpretado como uma enquete em cada regra $\mathcal{X} \rightarrow k \in \mathcal{R}_k^e$, é um voto dado pelos atributos em $\mathcal{X}$ para a classe k. O peso do voto $\mathcal{X} \rightarrow k$ depende da força da associação entre $\mathcal{X}$ e k que é dada pela confiança $\theta(\mathcal{X} \rightarrow k)$. O processo de estimativa da probabilidade de k ser a classe de revisão e começa pela soma dos votos ponderados para k e em seguida calcula-se a média obtida pela razão entre a soma total dos votos, e pelo número total de votos. É expressa pela função de pontuação $s(k, e)$ mostrada na equação abaixo, onde $r_j \subseteq \mathcal{R}_k^e$, e $|\mathcal{R}_k^e|$ é o número total de regras. A função de pontuação $s(k, e)$ dá-nos a confiança média das regras em $\mathcal{R}_k^e$.

$$s(k, e) = \frac{\sum_{j=1}^{|\mathcal{R}_k^e|} \theta(r_j)}{|\mathcal{R}_k^e|}, \tag{3.1}$$

Então a probabilidade estimada de k ser a classe de uma revisão e, denotada por $p(k|e)$ obtida pela normalização da função $s(k, e)$ é a seguinte:

$$p(k|e) = \frac{s(k, e)}{\sum_{j=1}^{n} s(k, e)}. \tag{3.2}$$

A classe de e associada ao maior valor da probabilidade $p(k, e)$, é prevista como a classe da revisão e.

Os dois parametros mais importantes do LAC são o número máximo de atributos em $\mathcal{X}$ e o mínimo de confiança, θ, permitida. No nosso trabalho definimos os mesmos valores que os da referência [Veloso et al., 2006], 3 e 0.001 respetivamente.

3.2 Seleção Ativa Para Detecção do Vandalismo

Nesta secção, apresentamos a estratégia de seleção ativa aplicada para reduzir a quantidade necessária de exemplos de treinamento (isto é, o número de revisões rotuladas em $\mathcal{D}$). Esta estratégia foi originalmente proposta para o contexto de ranqueamento de documentos [Silva et al., 2011]. Aqui aplicamos esta técnica para reduzir o esforço manual envolvido na construção do conjunto de treinamento utilizado para criar o modelo de detecção de vandalismo, $\mathcal{M}$, na Wikipedia. Nós referimos a nossa abordagem, como SADV - Seleção Ativa para Detecção de Vandalismo.

3.2.1 Seleção Ativa baseada em regras

A idéia por trás do SADV é que, escolhendo cuidadosamente um conjunto menor de exemplos de treinamento pode, ainda, ser possível aprender o modelo de detecção de vandalismo, $\mathcal{M}$, com uma eficácia similar a usar o modelo aprendido com uma coleção de treinamento muito maior. Como discutiremos mais abaixo, o SADV funciona iterativamente, isto é em cada iteração, seleciona um exemplo de um conjunto inicial de revisões não rotuladas, $\mathcal{U}$. Os exemplos selecionados são rotulados e depois inseridos no conjunto $\mathcal{D}$. O objetivo é mostrar que mesmo com poucos exemplos selecionados é ainda possível ter o mesmo desempenho a usar um conjunto de treinamento maior. A seguir descrevemos a estratégia de seleção (secção 3.2.1.1) e a condição de parada (Secção 3.2.1.2) adotadas no processo iterativo de seleção.

3.2.1.1 Estratégia de Seleção

Considere $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$, um grande conjunto de revisões não rotuladas. O objetivo é desenvolver um procedimento para selecionar, do conjunto $\mathcal{U}$, um pequeno subconjunto de revisões, tal que as revisões selecionadas contenham quase as mesmas informações de todas as revisões do conjunto $\mathcal{U}$. Estas revisões, são consideradas as mais informativas, e irão formar o conjunto de dados de treinamento, $\mathcal{D}$, tal que $|\mathcal{D}| \ll |\mathcal{U}|$. O SADV explora a redundância que existe entre as diferentes revisões no espaço de atributos do conjunto $\mathcal{U}$, isto é, muitas revisões em $\mathcal{U}$ podem ter atributos com mesmos valores. Então o SADV usa este fato para processar uma estratégia de seleção ativa baseada, a princípio, numa heurística, a *heurística de extração de regras*.

Intuitivamente, se uma revisão $u_i \in \mathcal{U}$ for inserida no conjunto de treinamento, $\mathcal{D}$, então o número de regras úteis para as revisões em $\mathcal{U}$ que compartilham os mesmos valores de atributos de u_i pode aumentar. Ao contrário, em relação as revisões em $\mathcal{U}$ que não contêm nenhum atributo com mesmo valor de um dos atributos da revisão selecionada, permanecem inalteradas. Portanto o número de regras extraídas para cada revisão em $\mathcal{U}$ pode ser usada como uma aproximação da quantidade de informação redundante entre as revisões em $\mathcal{D}$ e entre as revisões em $\mathcal{U}$. A função de seleção usada pelo SADV explora esta idéia para selecionar, primeiramente, revisões que contribuem com informação não redundante, e estas são as revisões mais informativas, são as que provavelmente irão exigir menor número de regras em $\mathcal{D}$. Mais especificamente, a função de seleção, $\delta(\mathcal{U})$, retorna uma revisão em $\mathcal{U}$ de acordo com a equação 3.3:

$$\delta(\mathcal{U}) = \{u_i \text{ tal que } \forall u_j : \ |\mathcal{R}^{u_i}| \ \leq \ |\mathcal{R}^{u_j}|\} \tag{3.3}$$

onde $|\mathcal{R}^{u_i}|$ é o número de regras da revisão u_i, isto é, regras $\mathcal{X} \rightarrow k$ tal que u_i contém todos os atributos em $\mathcal{X}$. A revisão retornada pela função de seleção é inserida no conjunto $\mathcal{D}$, mas também é mantida no conjunto $\mathcal{U}$.

Este procedimento continua iterativemente, selecionando em cada iteração uma revisão u_j das revisões que restam no conjunto $\mathcal{U}$. Note que em cada iteração o número de regras extraídas do $\mathcal{D}$ para cada revisão de $\mathcal{U}$ é suscetível a variar devido as revisões inseridas em $\mathcal{D}$ nas iterações anteriores. A intuição é sempre escolher uma revisão em $\mathcal{U}$ que exija um número menor de regras e que tenha poucos atributos com mesmos valores dos atributos de revisões já inseridas em $\mathcal{D}$. Isto é, o fato de que apenas poucas regras são extraídas de uma revisão u_i, serve como evidência de que $\mathcal{D}$ não contém revisões semelhantes a u_i e assim a informação contida na revisão u_i não é redundante. A heurística de "seleção de regras"trabalha em um nível refinado de valores de atributos, tentando maximizar a diversidade no conjunto de treinamento. As regras extraídas capturam a co-ocorrência dos valores dos atributos, ajudando a aumentar a diversidade, uma vez que a revisão que exige menor número de regras é aquela que contém menor número possível de atributos com valores iguais das instâncias existentes no conjunto dos dados de treinamento, $\mathcal{D}$.

Note que no início, o conjunto $\mathcal{D}$ está vazio pelo que não é possível o SADV extrair regras de $\mathcal{D}$. As primeiras revisões a serem rotuladas e inseridas em $\mathcal{D}$ são selecionadas no conjunto de revisões não rotuladas, o conjunto $\mathcal{U}$. A fim de maximizar a cobertura inicial de $\mathcal{D}$, a revisão selecionada é aquela que maximiza o tamanho dos dados projetados em $\mathcal{D}$. Tal revisão é a que tem muitos atributos com mesmos valores que dos outros da coleção e pode ser considerada a mais representativa da

coleção. Depois de selecionar a primeira revisão e rotular, o algoritmo continua usando a heurística acima descrita.

3.2.1.2 Condição de Parada

Depois de selecionar a primeira revisão, em cada iteração posterior, o SADV executa a função de seleção e uma nova revisão é selecionada do conjunto $\mathcal{U}$ e inserida no conjunto $\mathcal{D}$. A revisão selecionada em cada iteração é provável ser tão diferente possível das revisões já inseridas em $\mathcal{D}$. O SADV termina quando todas a revisões ainda disponíveis no conjunto $\mathcal{U}$ forem menos informativas do que qualquer revisão inserida no conjunto $\mathcal{D}$. Isto ocorre, exatamente, quando o SADV seleciona uma revisão que já existe em $\mathcal{D}$. Quando esta condição é alcançada, e o processo de seleção continuasse, levaria a escolha da mesma revisão várias vezes. Neste ponto, o conjunto de treinamento $\mathcal{D}$ contém as revisões mais informativas, e então podemos usar o algoritmo de classificação, no nosso caso, o LAC para detectar as instâncias de vandalismo no conjunto $\mathcal{T}$.

Capítulo 4

Avaliação Experimental

Nesta seção, descrevemos a avaliação experimental da técnica proposta. Começamos com a descrição do nosso conjunto de dados, seguido pela descrição dos *baselines*, e finalmente as métricas de avaliação utilizadas.

4.1 Conjunto de Dados

Para a tarefa de detecção do vandalismo na Wikipedia utilizamos o conjunto de dados do PAN-WVC-10 - *Uncovering Plagiarism, Authorship, and Social Software Misuse - Task 2: Wikipedia Vandalism Detection* - que compreende 32.452 edições em Inglês de 28.468 artigos diferentes, das quais 2.391 edições são de vandalismo [Potthast, 2010]. Cada edição representa a diferença entre duas revisões consecutivas de um artigo e a respectiva classe indica se a edição é vandalismo ou não. Como discutido, este conjunto de dados é muito desbalanceado com 92,7% dos casos pertencentes à classe de não vandalismo e apenas 7,3% sendo casos de vandalismo (a classe positiva). Esta situação tem algum impacto sobre as técnicas de detecção vandalismo por nós proposta, como veremos na Seção 5.1.

A construção do conjunto de dados do PAN-WVC-10 foi realizado por meio um esforço conjunto (*crowdsourcing*) especialistas da Wikipédia e para a sua anotação utilizou-se a estratégia similar à adotada para anotar o conjunto de dados Webis-WVC-07[1] [Potthast, 2010]. Nesta estratégia, cada edição foi anotada por três ou dezasseis revisores. Para casos em que as edições foram anotadas por três revisores levou-se em consideração duas situações que poderiam acontecer. A primeira é que todos os revisores podem concordar completamente ou 2 concordarem e um discordar

[1] http://www.uni-weimar.de/cms/medien/webis/research/corpora/webis-wvc-07.html

e vice versa. Para casos em que a edição foi analizada por 16 revisores consideraram o acordo de mais de $\frac{2}{3}$ dos revisores. Com esta estratégia 93% das edições do conjunto de dados da Webis-WVC-07 foram corretamente anotadas. Para a anotação do conjunto de dados do PAN-WVC-10, cada edição foi analizada por três revisores. Em casos em que a concordância em relação a uma edição foi abaixo de $\frac{2}{3}$, a edição é re-anotada por outros 3 revisores assim sucessivamente até atingir acordo de $\frac{2}{3}$ dos revisores ou a uma percentagem de acordo que não fosse suficiente para que a edição fosse anotada manualmente. A fim de se verificar o sucesso dos anotados na classificação das edições, cada 5ª edição a ser classificada foi de fato uma edição de vandalismo porque foi escolhida aleatoriamente do conjunto de dados da Webis-WVC-07. A partir da terceira iteração a verificação das edições foi feita através da escolha das edições de vandalismo já identificadas. Desta forma estas edições receberam mais votos do que necessário. Até a 8ª iteração restavam apenas 70 edições por analisar. Estas foram anotadotas por apenas dois especialistas que tomaram decisão de acordo com o melhor do seu conhecimento. A qualidade das anotações foi avaliada através Mechanical Turk da Amazon[2].

4.2 Representação das edições

Para a tarefa de detecção de vandalismo é essencial determinar quais atributos serão utilizados. Neste trabalho, foram utilizados atributos propostos em [Javanmardi et al., 2011; Potthast et al., 2010]. Nas próximas seções são apresentados os 67 atributos, utilizados no presente trabalho, divididos em quatro categorias.

4.2.1 Atributos de Usuário

Estes atributos foram extraídos através da mineração do histórico das edições até um dado momento T (18-11-2009). São no total 12 atributos nessa categoria. Note que o usuário pode ser registrado ou anônimo (representado por IP).

- *DSR, DDSR, Reputação*: estes atributos estão relacionados com a reputação do usuário. A reputação é a pontuação atribuída a um usuário com base nas edições anteriores. Geralmente aos usuários anonimos e novos atribui-se a reputação 0 DSR (Data Stability Ratio): mostra a percentagem de conteúdo contribuído por um usuário que não tenha sido, ainda, excluído por outros usuários.

[2]http://en.wikipedia.org/wiki/Amazon_Mechanical_Turk

- *Inserted/Deleted word* : número total de palavras inseridas ou removidas pelo usuário.
- *Lost Words*: número total de palavras removidas do usuário.
- *Inserted/Deleted Revision*: número total de revisões inseridas ou removidas pelo usuário.
- *Inserted/Deleted pages*: número total de páginas criadas ou removidas pelo usuários.
- *User type*: direitos especiais dos usuários (administrador, burocrata, ou robot);
- *User page*: se o usuário tiver pagina pessoal, este atributo assume valores 1, 0 do contrário;

4.2.2 Atributos de Texto

Foram extraídos, desta categoria, 30 atributos calculados a partir do conteúdo inserido ou apagado e para distinguí-los usaram os prefixos "Ins"e "Del"que correspondem a inserir ou apagar o conteúdo. Outro exemplo deste tipo de atributos, consiste no vulgarismo, isto é espera-se que a inserção de palavras vulgares seja sinal de vandalismo, e o ato de apagar tais palavras seja sinal de uma edição legítma com objetivo de remover o vandalismo.

- *Inserted Size*: número de palavras inseridas na revisão.
- *Deleted Size*: número de palavras removidas na revisão.
- *Revision Size*: proporção entre a diferença dos tamanhos da nova revisão e da anterior, $\frac{1-|new|}{1-|old|}$
- *Blanking*: quando todo artigo foi apagado.
- *Internal Links*: número de links internos adicionados nos artigos da Wikipedia.
- *External Links*: número de links externos adicionados.
- *Word Repetitions*: tamanho da maior palavra repetida inserida no texto. Este atributo é muito utilizado para detectar palavras sem sentido, cujo sua existência pode ser sinal de vandalismo.

- *Char Repetitions*: comprimento da maior sequência consecutiva do mesmo caracter inserido no texto. Sequências longas do mesmo caráter são freqüentes no vandalismo.

- *Compressibility*: taxa de compreensão de texto inserido numa revisão. Foi calculado utilizando o algorítmo *Lempel-Ziv-Welch*[3] (LZW) .Este atributo é útil para detectar caracteres ou palavras repetidas e palavras sem sentido.

- *Capitalization**: proporção de carateres maiúsculos sobre caracteres minúsculos. Geralmente, os vândalos não seguem as regras de capitalização escrevem tudo em minúsculas ou em maiúsculas, $\frac{1-|upper|}{1-|lower|}$

- *Capitalization All**: Proporção de caracteres maiúsculos sobre todos os caracteres, $\frac{1-|upper|}{1-|lower|+|upper|}$

- *Digits**: taxa de dígitos sobre todos os caracteres. Este atributo ajuda a detectar em revisões muito pequenas aquelas em que houve apenas alteração de dígitos (nos números ou datas), o que pode ajudar a identificar casos de vandalismo sutís, $\frac{1-|digit|}{1-|all|}$

- *Special Chars**: Proporção de caracteres não-alfanuméricos sobre todos os caracteres. Um excesso de caracteres não-alfanuméricos em textos curtos pode indicar o uso excessivo de pontos de exclamação ou emoções, $\frac{1-|nonalphanumeric|}{1-|all|}$

- *Diversity**: Mede o número de caracteres diferentes em comparação com o comprimento do texto inserido, dado o comprimento expressão, $\frac{1}{differentchars}$

- *Inserted Words**: média da frequência de um termo nas palavras inseridas.

- *Vulgarism**: frequência de palavras vulgares ou ofensivas.

- *Bias**: frequência (impacto) de palavras com viés de alta. Por exemplo *coolest*

- *Sex**: frequência de palavras relacionadas com sexo.

- *Spam**: Frequência (impacto) de palavras usadas frequentemente em spam.

- *Pronouns**: frequência (impacto) de pronomes pessoais. É a percentagem com que pronomes pessoais aumentam de uma revisão para a revisão.

- *Markup**: Proporção de caracteres wikitexto novos (alterados) sobre todos os caracteres wikitexto.

[3]http://en.wikipedia.org/wiki/Lempel%E2%80%93Ziv%E2%80%93Welch

- *Special Words**: agregação de vulgarismo, viés, sexo, spam, pronomes e taxas de markup.

4.2.3 Atributos de Metadados

A extração destes atributos foi feita a partir dos comentários associados às edições, sendo no total 22 atributos. Alguns destes atributos são semelhantes aos atributos textuais (marcados com *), só que estes foram extraídos com base no comentário. Para este tipo de atributos, também foram extraidos unigramas, bigramas e trigrama a partir do tipo de comentários. Por exemplo, neste tipo de atributos, o intervalo de tempo muito curto entre duas edições pode ser um sinal para detecção do vandalismo.

- *Time Diff*: Intervalo de tempo entre a submissão da antiga e da nova revisão.
- *Category*: se o comentário automatic contém a *"category"*.
- *Early Years*: se o comentário automatic contém *"early years"*.
- *Copyedit*: se o comentário automatic contém *"copyedit"*.
- *Personal Life*: se o comentário automatic contém a *"personal life"*.
- *Revert*: se o comentário automatic contém *"revert"*.
- *Length*: tamanho do comentário.
- *Reverted*: se o MD5 (Message-Digest algorithm 5) de revisões novas é o mesmo que uma das mais antigas.

4.2.4 Atributos do Modelo de Linguagem

Consistem em 3 atributos, extraído a partir do cálculo do *Kullback-leibler distance* (KLD) entre dois modelos de linguagem unigrama. Calculou-se o KLD entre a revisão anterior e a atual; KLD entre o conteúdo inserido e a edição anterior; O KLD entre o conteúdo apagado e a edição anterior. Este tipo de atributo, foi introduzido, porque alguns casos de vandalismo surgem através de palavras inesperadas, e essas mudanças podem ser vistas através da distância, e apagar as palavras inesperados pode ser um indicador de uma revisão legítima.

- *KL Distance*: distância *Kullback_ Leibler* entre a antiga e nova revisão.
- *KL Distance Ins*: distância *Kullback_ Leibler* entre o conteúdo inserida e a revisão anterior.

- *KL Distance Del*: distância *Kullback_ Leibler* entre o conteúdo removido e a revisão anterior.

4.3 Baselines

Realizamos uma série de experimentos com o nosso classificador (LAC) e a com abordagem proposta, utilizando o conjunto de treinamento completo e o reduzido. Para avaliar a eficácia do nosso classificador e da abordagem proposta consideramos três *baselines*. Primeiro comparamos os resultados do nosso classificador (LAC) com os resultados dos classificadores do estado de arte, nomeadamente, SVM e KNN, como nossos *baselines*. Nesta comparação utilizamos, no primeiro caso, o conjunto completo de treino sem balanceamento, mostrado na Seção 5.1, e em seguida aplicamos proporções de balanceamento da classe 0 (regular) sobre a classe 1 (vandalismo) no conjunto de treinamento, como explicado na Secção 5.1.1. Esta comparação permite-nos avaliar a relação entre número de exemplos rotulados (com e sem distribuição assimétrica das classes) e a eficácia da classificação. O nosso segundo *baseline* foi o resultado obtido da classificação com um conjunto de treino, construído a partir da seleção aleatória. Neste caso selecionamos de forma aleatória a mesma quantidade de instâncias selecionadas pelo SADV na primeira, segunda e terceira rodadas (descrito na Secção 5.2) e usamos o LAC para aprender a função de detecção do vandalismo. A ideia é mostrar que a seleção ativa é muito melhor, isto é, seleciona melhor a mesma quantidade de instâncias em cada rodada, que a seleção aleatória. E por fim comparamos os nossos resultados com os melhores resultados publicados na competição PAN- 2010 [Potthast, 2010], da qual usamos o mesmo conjunto de dados.

4.4 Métricas de Avaliação

4.4.1 Macro-F1

Para avaliar a eficácia de nossa estratégia de classificação utilizamos as métricas amplamente utlizadas na recuperação da informação nomeadamente, precisão, revocação, microF_1 e Macro-F_1 Baeza-Yates & Ribeiro-Neto [2011]. A revocação (r) de uma classe l_i é a razão entre o número de edições correctamente classificadas e o número de edições na classe l_i. Precisão (p) de uma classe l_i é a razão entre o número de edições corretamente classificadas pelo número total de edições previstas como da classe l_i. MicroF_1 é a média entre a precisão e a revocação, $\frac{2p_i r_i}{p_i + r_i}$. A Macro-F1 é uma variação da F1 e o seu valor é calculado a partir do cálculo de F1 para cada classe, e em seguida

a média de todas as classes. A Macro-F_1 considera com igualdade a importância da efetividade das classes independentemente do tamanho da classe. Sendo uma métrica importante quando estamos perante um conjunto de dados cuja distribuição de classes é assimétrica, como no nosso caso, a fim de verificar o desempenho do classificador na classe menor.

4.4.2 Receiver Operator Characteristic-Area Under the Curve(ROC-AUC)

Para avaliar o desempenho dos nossos classificadores utilizamos também a curva ROC. Segundo [Davis & Goadrich, 2006], a curva ROC, mostra como o número de exemplos positivos corretamente classificados varia de acordo com o número de exemplos negativos incorretamente classificados . Esta é recentemente utilizada, por muitos investigadores [Hegedus et al., 2010; Mola-Velasco, 2010], como a métrica mais relevante para avaliar o desempenho dos seus algoritmo em tarefas de classificação binária. Por exemplo, os resultados do PAN-10 publicados em [Potthast, 2010] foram ordenados de acordo com os valores de ROC-AUC. Em problemas de classificação binária, o classificador rotula os exemplos como positivos (P) ou negativos (N). No nosso caso denotamos as edições de vandalismo como positivos e as regulares como negativos. Esta classificação é representada numa matriz de confusão conforme mostra a tabela 4.1, onde verdadeiros positivos (TP) são os exemplos positivos corretamente classificados, falsos positivos (FP), referem-se aos exemplos negativos classificados como positivos. Os verdadeiros negativos (TN), são os exemplos negativos corretamente classificados e finalmente, os falsos negativos (FN) correspondem aos exemplos positivos que foram incorretamente classificados, como negativos. Então a matriz de confusão é usada para construir cada ponto no espaço ROC. O espaço ROC represena a gráfico cujo eixo-x corresponde aos valores da taxa de falsos positivos (FPR) e o eixo-y a taxa de verdadeiros positivos (TPR). O FPR mede a fração dos exemplos negativos que foram incorretamente classificados como positivos, e o TPR (ou revocação) corresponde a fração dos exemplos positivos corretamente classificados como positivos.

Tabela 4.1. Matriz de Confusão

	Real	
Predições	P	N
P	TP	FP
N	FN	TN

$$TPR = \frac{TP}{TP + FN} \tag{4.1}$$

$$FPR = \frac{FP}{FP + TN} \tag{4.2}$$

A curva ROC é simples de ser gerada quando avaliamos algorítmos de classificação binária, cujas saídas são os valores das probabilidades da classe. Para se obter um ponto da curva, o limiar α é aplicado para mapear os valores da probabilidade da classe. Existem pontos notáveis no espaço ROC [Fawcett, 2006]: O ponto (0,0), é o ponto que indica que o classificador não detectou nenhum positivo. Neste caso, todas as edições são classificadas como regulares. O segundo ponto é de (1,1), onde tudo é classificado como positivo, isto é, todas as edições são classificadas como vandalismo. Em (1,0) todas as edições são classificadas correctamente e em (0,1) todas as edições são classificados incorretamente. A diagonal de (0,0) para (1,1) mostra os desempenhos esperados dos classificadores que selecionam as classes de forma aleatória. A curva ROC é gerada atraves dos pontos obtidos pela variação do limiar α entre {0,1} no espaço ROC. O simples valor do desempenho do classificador é obtido pelo cálculo da área sob a curva (AUC) que é independente do α. Para o cálculo do AUC utilizamos o AUCCalculator[4]. Através do valor do desempenho, o classificador pode ser comparado com os outros, quanto maior for a AUC, melhor é o classificador.

4.5 Projeto Experimental

Os experimentos realizados tiveram como objetivo analisar os resultados do nosso classificador na detecção do vandalismo na Wikipedia e comparar a nossa abordagem com outros métodos propostos que utilizaram o mesmo conjunto de dados. Todos os experimentos da classificação foram realizados utilizando a validação cruzada com 5-*folds*. Em cada teste, a amostra original é particionada em 5 sub-amostras, das quais quatro são utilizados como dados de treinamento, e uma, a restante, é usada para testar o classificador. O processo é então repetido 5 vezes, com cada uma das 5 sub-amostras produzindo assim 5 resultados. Os resultados apresentados (em tabelas) são as médias das 5 execuções de cada classificador (KNN, SVM, LAC, SADV), com respectivos intervalos de confiança com 95% de nível de confiança. Os experimentos foram repetidos várias vezes, variando os parâmetros e em todos os casos relatamos os melhores resultados.

[4]http://mark.goadrich.com/programs/AUC/

Para os resultados, apresentados na seção 5.1, em relação ao LAC, SVM e KNN usamos o conjunto de treino completo, como dados de treinamento para aprender a função de classificação, já para o SADV utilizamos como dados de treinamento, os exemplos selecionados. Para SADV, o conjunto de treino utilizado foi formado através do processo de seleção, descrito na Seção 3.2. Neste processo, o conjunto de treino, $\mathcal{D}$, é inicialmente vazio e aumenta a medida que um exemplo é selecionado do conjunto maior de exemplos não rotulados, rotulado e inserido no conjunto. O algoritmo, eventualmente, converge quando se seleciona um exemplo que já foi selecionado. Quando isto acontece, os exemplos selecionados (e rotulados) são usados para constituir o conjunto de treinamento reduzido para aprender a função de clssificação binária. O processo de seleção, para formar os conjuntos de treinamento reduzido utilizados para aprender a função de classificação no SADV foi realizado da seguinte forma: selecionamos os exemplos, através do processo descrito na Seção 3.2.1 formando o primeiro conjunto de treinamento, e nos referimos como primeira rodada. Em seguida removemos do conjunto de dados não rotulados, os exemplos selecionados, e executamos novamente o algorítmo selecionando novas instâncias, na segunda rodada. Adicionamos as novas instâncias sobre as instâncias selecionadas na primeira rodada, formando assim o segundo conjunto de treinamento. O procedimento foi repetido até a terceira rodada, onde formamos o terceiro conjunto de treinamento que é composto pelas instâncias selecionadas na primeira, segunda e terceira rodadas. E os experimentos realizados com conjuntos de treinamentos constituidos por instâncias selecionadas de forma aleatória foram repetidos 30 vezes com diferentes amostras do mesmo tamanho.

Capítulo 5

Resultados Experimentais

Esta seção apresenta os resultados mais relevantes comparando diferentes abordagens de classificação consideradas neste trabalho. Em todos os experimentos, os conjuntos de teste são mantidos iguais para todas as abordagens avaliadas. Todos os resultados aqui apresentados são médias de 5 execuções, como explicado na seção anterior. E os nossos resultados são relatados em termos de precisão, revocação e Macro-F_1 com os respectivos intervalos de confiança para cada classe e cada classificador, exceto os da seção 5.3, são apresentados em termos de ROC-AUC. Para os nossos experimentos utilizamos utilizamos o libSVM [chung Chang & Lin, 2001], uma implementação de SVM, o KNN com 5 vizinhos mais próximos e o LAC, com os seguintes parâmetros σ_{min}=1, $\mathcal{X} = 3$, $\theta(\mathcal{X} \rightarrow k) = 0,001$.

5.1 Resultados da classificação Sem considerar assimetria de Dados

Nesta seção apresentamos os resultados obtidos por todos os métodos supervisionados utilizando os conjuntos de treinamento completo, sem ponderar as proporções de balanceamento entre as classes. Como podemos ver, na Tabela 5.1, particularmente, nenhum algoritmo executa bem, mas o SVM apresenta o melhor valor de Macro-F1. O LAC, não foi capaz de classificar corretamente nenhuma instância de vandalismo, no conjunto de teste. Enquanto que, o KNN e SVM só conseguem recuperar em média apenas 15% e 19% de instâncias de vandalismo, respectivamente. Nossa suposição, para explicar estes resultados, é o fato de que o nosso conjunto de treinamento apresenta uma distribuição de classes muito desbalanceada. Face a esta situação, na próxima seção propomos soluções para este problema.

Tabela 5.1. Resultados da Comparação com Baselines sem considerar a assimetria dos dados com respetivos intervalos de confiança

algoritmo de	Regular		Vandalismo		Macro
Classificação	Precisão	revocação	Precisão	revocação	$F-1$
SVM	93.86±0.0015	99.3±0.0023	68.61±0.0167	18.48±0.0108	62.75.±0.0108
KNN	93.5±0.4421	98.3±0.1905	40.1±3.9141	14.3±1.8145	58.4±1.2518
LAC	92.6±0.0740±	1(0)	0(0)	0(0)	47.8±0.0008

5.1.1 Resultados da classificação aplicando as proporções de balanceamento

Para validar a nossa suposição, de que a assimetria entre os dados no conjunto de treinamento pode afetar, seriamente, a eficácia de alguns algoritmos de classificação, realizamos nesta seção experimentos aplicando uma técnica de balanceamento muito simples. Para equilibrar os conjuntos de treinamento, eliminamos instâncias na classe com maior quantidade de instâncias, a classe 0 (ou a classe negativa que representa revisões regulares), até encontrarmos proporções de exemplos negativos sobre os positivos, que produzem melhores resultados em termos de Macro-F_1. Executamos vários experimentos, para determinar a melhor proporção de balanceamento, tomando uma porção do conjunto de treinamento, como conjunto de validação e variando as proporções de exemplos negativos sobre os exemplos positivos no conjunto de treinamento. Para equilibrar o nosso conjunto treinamento, definimos essa a proporção p que varia de 0.5 a 5.0, C_0 como o número de exemplo negativos e C_1 o número de exemplos positivos. Usando a fórmula $p \times C_1$, calculamos o número de instâncias negativas que vão permanecer no conjunto de treinamento. Em seguida eliminamos, de forma aleatória, as instâncias negativas até restar $p \times C_1$ instâncias. A Figura 5.1 mostra a variação da Macro-F_1 de acordo com a proporção de negativos:positivos para cada classificador. Como podemos ver em relação ao KNN e SVM a variação é muito pequena enquanto que o LAC converge muito rápido. Dado ao fato de a variação da Macro-F_1, para o SVM e KNN, ser muito pequeno, apresentamos no gráfico os resultados a partir da proporção, p=1. Os três classificadores atingiram o valor máximo da Macro-F_1, em proporções diferentes. Para o LAC, o valor máximo da Macro-F_1, foi encontrado entre as proporções 1.5 e 1.6, para o KNN entre 3.5 e 3.6. Finalmente o SVM atingiu o seu máximo entre as proporções 4,3 e 4,4.

A Tabela 5.2 mostra os resultados de cada um dos algoritmos supervisionados (LAC, KNN, SVM) nas proporções 1.5, 3.6 e 4.4 respectivamente. Podemos notar que o LAC supera os dois outros algoritmos com uma Macro-F_1 de 80,9%. Além disso, os resultados do LAC são elevados em termos de revocação, isto é, o LAC foi capaz de

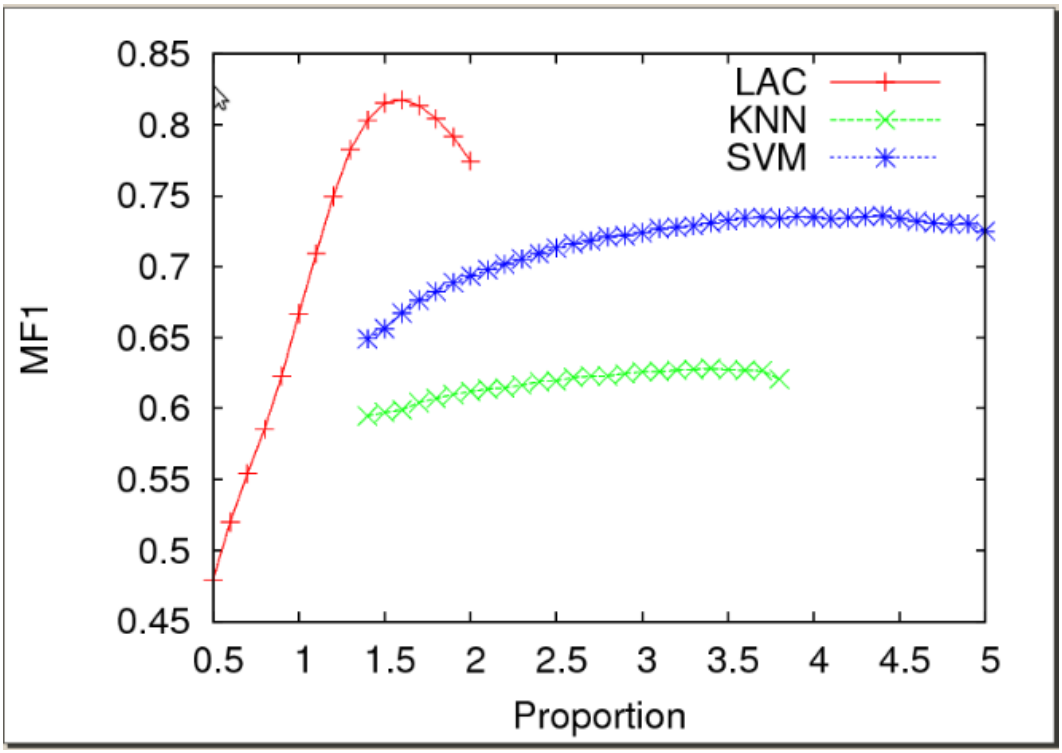

Figura 5.1. Resultados da Macro-F_1 para o LAC, KNN, E SVM como função da proporção exemplos negativos sobre os exemplos positvos no conjunto de treinamento

recuperar cerca de 75% de revisões da classe do vandalismo[1], enquanto que cerca de 60% foram previstas como sendo de fato revisões vandalizadas. Levando em consideração o fato de que o LAC classificou erroneamente uma quantidade muito pequena de revisões regulares e que o número de exemplos de vandalismo é muito menor do que o normal, a nossa estratégia pode ser considerada adequada para ser utilizada, pelos editores da Wikipedia que classificam as revisões manualmente, como um sistema de filtragem para avaliar se as revisões previstas como vandalismo são de fato atos maliciosos.

Tabela 5.2. Resultados dos métodos supervisionados com proporções negativo:positivos de 1.5:1, 3.6:1 e 4.4:1 respetivamente (com respectivos intervalos de confiança com um nível de confiança de 95%)

Algorítmo de Classificação	Regulares		Vandalismo		Macro $F-1$
	Precisão	Revocação	Precisão	Revocação	
SVM	96.66±0.0018	94.54±0.00158	46.09±0.0069	58.59±0.0231	73.58±0.0.0068
KNN	95.75±1.105	86.77±4.998	26.26±3.1353	55.83±13.5691	62.82±1.6924
LAC	97.9±0.0023	95.5±0.0064	57.3±0.0161	75.1±0.0265	80.9±0.0065

[1]note que os baselines tem valor baixo da precisão

5.2 Resultados da classificação com Treino Reduzido

Como vimos na seção anterior, os experimentos realizados foram baseados no conjunto de treinamento completo. Como explicamos na seção 4.1, a construção destes conjuntos de dados incorre em custos elevados para a Wikepedia. Além disso há necessidade constante de actualização destas coleções já que alguns padrões são identificados após a submissão de uma nova revisão, e só depois os editores poderão rotular a revisão como legítima ou vandalismo. Como o objetivo principal deste trabalho é reduzir os custos da construção do conjunto de treinamento, nesta seção apresentamos uma breve discussão sobre os resultados obtidos pelo LAC usando o conjunto de treino selecionado de forma ativa na primeira, segunda e terceira rodadas. Assim como na seção anterior, aqui também aplicamos a estratégia de balanceamento no conjunto de treino selecionado. A Tabela 5.3 mostra os resultados da classificação, da primeira, segunda e terceira rodadas, utilizando SADV bem como a percentagem cumulativa de dados de treinamento selecionados em cada rodada. Neste caso, para determinar a melhor proporção (p) de negativo:positivo, dividimos os conjuntos selecionados pelo SADV, em duas metades, utilizando-se de uma para treinamento e a outra para a validação. Como podemos ver na tabela, em qualquer uma das rodadas, os resultados foram muito próximo aos da seção anterior, com a diferença apenas na melhor proporção de balanceamento que foi de 1.6:1 ao invés de 1.5:1. A percentagem final do conjunto de treinamento utilizado para a aprender a função de classificação é visto na última coluna da tabela.

Tabela 5.3. Resultados comparativos entre selecção ativa e selecção aleatória com a proporção negativo:positivo de 1.6:1 (com intervalo de confiança com 95% de confiança)

Algoritmo de	Regular		Vandalismo		Macro	%_Treino	
Classificação	Precision	Recall	Precision	Recall	$F-1$	selec	balanc
Round1	96.7±0.0083	95.5±0.0174	53±0.7039	58.77±0.11053	75.22±0.103	1.3	0.4
Round2	96.8±0.0107	95.9±0.0195	56±0.0970	59.8±0.1428	76.1±0.0163	2.68	0.8
Round3	96.7±0.0091	96.4±0.0193	59.4±0.0796	59±0.1208	77.0±0.0060	4.14	1.13
Random 1	94.2±0.002	98.4±0.003	57±0.037	23.8±0.034	62.9±0.0164	1.3	0.4
Random 2	94.4±0,002	97.2±0.014±	62±0.023	26.8±0.360	63.9±0.0169	2.68	0.8
Random 3	94.4±0.0805	98.5±0.114	64±2,167	26.3±2.364	65.5±1.236	4.14	1.13

Podemos ver que SADV foi capaz de selecionar muito poucos exemplos, 1.3% de todos os dados de treinamento disponíveis, na primeira rodada; 2.68% na segunda e 4.14% na terceiro. Isto corresponde, em média, a aproximadamente 40, 80 e 110 instâncias de vandalismo, no conjunto de treinamento selecionado em cada rodada. De fato, a quantidade de instâncias de vandalismo que foi realmente utilizada para a

treinamento é ainda menor (devido ao desequilíbrio), embora, na prática, o utilizador tenha que rotular todas as instâncias selecionadas durante o processo de seleção ativa. Observe que o resultado na terceira rodada é muito próximo das duas outras, e se o custo de rotulagem for alto então não seria necessário ir até a terceira rodada, poderíamos parar na segunda ou então ficar apenas na primeira. Importa ainda salientar que a eficácia do método, com uma quantidade muito reduzida de treinamento, é muito próxima a do LAC que utiliza o conjunto completo de treino (veja a última linha da tabela 5.2), com 3,9% a menos em termos de Macro-F_1.

Para validar a nossa abordagem, de seleção ativa, comparamos os nossos resultados com os resultados da classificação utilizando conjuntos de treinamento formados através da seleção aleatória, também em três rodadas. Como podemos ver na tabela 5.3, os resultados da nossa abordagem em termos Macro-F_1 superam a seleção aleatória até 14,5% na terceira rodada o que demostra claramente a superioridade da nossa aboradgem de seleção ativa em selecionar melhor as instâncias para formar o treino reduzido.

Em termos da curva ROC, o valor do AUC obtido com o treino reduzido em quase 96%, com base na validação cruzada de 5-*folds* foi de 0,9226.

5.3 Comparação dos resultados das nossas abordagem com os melhores resultados da Competição CLEF 2010

Nesta seção comparamos as nossas abordagens com as melhores abordagens da competição CLEF 2010 da qual utilizamos conjunto de dados. Uma vez que na competição utilizaram, nos seus experimentos, um procedimento diferente do nosso (aqui utilizamos a validação cruzada de 5-*folds*), optamos por utilizar o procedimento deles para que os nossos resultados fossem diretamente comparáveis. Especificamente os organizadores da CLEF 2010 na tarefa de detecção do vandalismo dividiram a coleção PAN-WVC-10 em duas partes, aproximadamente 50-50(%) em treinamento e teste [Potthast, 2010]. O conjunto de teste era composto por 17.443 revisões das quais apenas 1481 são de vandalismo e não houve repetições dos seus experimentos. A principal métrica de avaliação por eles utilizada foi o ROC-AUC, embora os resultados em termos de PR-AUC também tenham sido relatados. A Tabela 5.4 mostra os resultados do desempenho de LAC e SAVD em termos de ROC-AUC posicionados entre os melhores métodos concorrentes, já na posição correta de acordo com [Potthast, 2010]. Para gerar as

curvas ROC e calcular as áreas sob as curvas, foi utilizado um limiar (α) que vai de 0.95 a 0 com intervalos de 0.05. A melhor proporção negativo:positiva que encontramos para essa divisão usando validação cruzada no conjunto de treinamento foi 1,4:1 para ambos(LAC e SADV).

Tabela 5.4. Comparação dos resultados do LAC e SADV com os resultados dos melhores competidores da CLEF 2010 em termos de ROC-AUC

ROC-AUC	ROC Rank	Detector	%_Treino
0.9223	1	[Mola-Velasco, 2010]	100%
0.9110	**2**	**LAC**	100%
0.90351	3	[Adler et al., 2010]	100%
0.89856	4	[Javanmardi et al., 2011]	100%
0.89377	5	Chichkov [2010]	100%
0.89350	**6**	**SADV**	**4.65%**
0.87990	7	Seaward [2010]	100%
0.87669	8	[Hegedus et al., 2010]	100%
0.85875	9	Harpalani et al. [2010]	100%
0.84340	10	[White & Maessen, 2010]	100%
0.65404	11	[Iftene, 2010]	100%

Para SADV utilizamos o conjunto de treinamento correspondente a segunda rodada, que foi a que produziu melhores resultados nesta experimentação. Este treinamento equivale a **4.65%** do conjunto original dos dados. Como podemos ver na tabela, os métodos por nós propostos, encontram-se entre as melhores posições, de acordo com a métrica ROC-AUC, em relação a alguns competidores. O LAC é o segundo melhor método, enquanto que o SADV encontra-se em sexto lugar, mas com valor muito próximo ao do quarto e quinto lugares, que também se configura entre as melhores posições.

5.4 Considerações Finais

Em [Davis & Goadrich, 2006], os autores sugerem que a curva ROC pode apresentar uma visão excessivamente otimista do desempenho de um algoritmo em casos de tarefas de classificação com as distribuições de classe altamente enviesada, como é o nosso caso. Portanto, nós também comparamos os nossos valores de F1 com os resultados do vencedor do concurso CLEF-2010 relatados em [Mola-Velasco, 2010]. Eles apresentam a precisão, a revocação e MacroF_1 de 0.861, 0.568 e 0.684 respectivamente. Comparando estes resultados com os nossos melhores, podemos ver que o valor da revocação para o LAC é muito superior ao deles. Como afirmamos antes, acreditamos que um sistema de

filtragem para editores humanos tem como objetivo maximizar a detecção de revisões que são suspeitas como vandalismo (já que o número de revisões de vandalismo é na prática muito menor que o das revisões regulares), mantendo níveis consideráveis da precisão para evitar falsos negativos e reduzir o esforço extra para verificação manual. Em outras palavras, um método prático de detecção de vandalismo deve maximizar a revocação na classe positiva enquanto mantém bons níveis de precisão. Quando comparado com os melhores resultados do SADV (Ver Tabela 5.3) a nossa revocação ainda está um pouco melhor, com uma queda muito pequena de precisão, o que significa que os editores podem ter algum trabalho extra olhando para as revisões de vandalismo mal classificadas. No entanto, isto vem com reduções enormes de custos para construir o conjunto de treinamento inicial, onde apenas algumas (muito poucas) revisões devem ser rotulados, como visto na seção 3.2.1.1, para se obter uma eficácia muito boa. Podemos sempre que necessário acrescentar novas instâncias selecionadas e rotuladas no conjunto de treinamento para ser utilizados pelo LAC para criar modelo de classificação. Além disso, porque o nosso método é preguiçoso, qualquer treinamento rotulado recentemente será imediatamente disponível para ser utilizado para detecção do vandalismo. Finalmente, outro aspecto muito importante é que o SADV pode ser executado a qualquer momento para selecionar novas instâncias, permitindo a detecção de novos padrões de comportamentos mal-intencionados que podem ocorrer no sistema à medida que este evolui.

Capítulo 6

Conclusão e Trabalhos Futuros

Nesta dissertação foi proposta uma abordagem de detecção de atos maliciosos em sites cujo conteúdo é gerado colaborativamente, na Web 2.0. Adotamos a Wikipédia como caso de estudo, e utilizando o conjunto de dados da competição PAN-10 desenvolvemos uma abordagem de deteção de vandalismo com base em Classificador LAC e na seleção ativa, utilizada para formar o conjunto reduzido de treinamento. A nossa principal motivação foi reduzir os custos de rotulação dos dados. Na primeira fase dos nossos experimentos, sem considerar o desbalanceamento dos dados, notamos que o nosso classificador alcançou resultados piores comparando com outros métodos supervisionados, SVM e KNN. A aplicação sobre os nossos dados uma solução simples de balanceamento do treino, tornou a nossa abordagem muito superior aos outros classificadores em termos de Macro-F_1. Além disso, utilizando a seleção ativa para a detecção do vandalismo (SADV), fomos capazes de reduzir o conjunto de treinamento em quase 96%, e utilizando o classificador associativo e o balanceamento dos dados, produzimos resultados muitos próximos aos resultados obtidos quando utilizamos o conjunto de treinamento completo, o que torna a nossa abordagem muito prática e eficaz. Comparando os nossos métodos com os resultados dos melhores métodos da competição CLEF de 2010, obtivemos resultados superiores à maioria dos métodos em termos de AUC-ROC.

Como trabalhos futuros pretendemos explorar a nossa abordagem com a estratégia multi-visão. Também planejamos integrar à nossa abordagem métodos de seleção de atributos utilizando algoritmos genéticos.

Referências Bibliográficas

Adler, B. T.; de Alfaro, L. & Pye, I. (2010). Detecting wikipedia vandalism using wikitrust.

Agrawal, R.; Imielinski, T. & Swami, A. N. (1993). Mining association rules between sets of items in large databases. In *SIGMOD Conference*, pp. 207–216. ACM.

Baeza-Yates, R. & Ribeiro-Neto, B. (2011). *Modern Information Retrieval - the concepts and technology behind search, Second edition.* Pearson Education Ltd., Harlow, England.

Belani, A. (2010). Vandalism detection in wikipedia: a bag-of-words classifier approach. *Computing Research Repository.*

Benevenuto, F.; Rodrigues, T.; Almeida, V.; Almeida, J. & Gonalves, M. (2009). Detecting spammers and content promoters in online video social networks. In *Proc. of Int'l ACM SIGIR*, Boston, MA, USA.

Chichkov, D. (2010). Submission to the 1st international competition on wikipedia vandalism detection. In *CLEF (Notebook Papers/LABs/Workshops).*

Chin, S.-C.; Street, W. N.; Srinivasan, P. & Eichmann, D. (2010). Detecting wikipedia vandalism with active learning and statistical language models. In *Proceedings of the 4th workshop on Information credibility*, WICOW '10, pp. 3--10, New York, NY, USA. ACM.

chung Chang, C. & Lin, C.-J. (2001). Libsvm: a library for support vector machines.

comScore (2012). comscore releases february 2012 u.s. online video rankings. `http://www.comscore.com/Press_Events/Press_Releases/2012/3/comScore_Releases_February_2012_U.S._Online_Video_Rankings`.

Cunningham, P. & Delany, S. J. (2007). k-nearest neighbour classifiers.

Davis, J. & Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*. ACM.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recogn. Lett.*, pp. 861--874.

Gonalves, A. R. (2012). Mquina de vetores suporte. `www.dca.fee.unicamp.br/~andreric/arquivos/pdfs/svm.pdf`.

Harpalani, M.; Phumprao, T.; Bassi, M.; Hart, M. & Johnson, R. (2010). Wiki vandalysis - wikipedia vandalism analysis - lab report for pan at clef 2010. In *CLEF (Notebook Papers/LABs/Workshops)*.

Hasan Dalip, D.; André Gonçalves, M.; Cristo, M. & Calado, P. (2009). Automatic quality assessment of content created collaboratively by web communities: a case study of wikipedia. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, JCDL '09, pp. 295--304, New York, NY, USA. ACM.

Hegedus, I.; Ormandi, R.; Farkas, R. & Jelasity, M. (2010). Novel balanced feature representation for wikipedia vandalism detection task - lab report for pan at clef 2010. In *CLEF (Notebook Papers/LABs/Workshops)*.

Iftene, A. (2010). Submission to the 1st international competition on wikipedia vandalism detection. In *CLEF (Notebook Papers/LABs/Workshops)*, Romania. From the Universtiy of Iasi, Romania.

Javanmardi, S.; McDonald, D. W. & Lopes, C. V. (2011). Vandalism detection in wikipedia: a high-performing, feature-rich model and its reduction through lasso. In *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, WikiSym '11, pp. 82--90, New York, NY, USA. ACM.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features.

Lee, K.; Caverlee, J. & Webb, S. (2010). Uncovering social spammers: social honeypots + machine learning. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pp. 435--442, New York, NY, USA.

Mola-Velasco, S. M. (2010). Wikipedia vandalism detection through machine learning: Feature review and new proposals - lab report for pan at clef 2010. In *CLEF (Notebook Papers/LABs/Workshops)*.

Pappa, G. L. (2002). Seleo de atributos utilizando algortmos genticos. `www.lania.mx/~ccoello/EMOO/thesis_pappa.pdf.gz`.

Potthast, M. (2010). Crowdsourcing a wikipedia vandalism corpus. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pp. 789--790, New York, NY, USA. ACM.

Potthast, M.; Stein, B. & Gerling, R. (2008). Automatic vandalism detection in wikipedia. In *Proceedings of the IR research, 30th European conference on Advances in information retrieval*, ECIR'08, pp. 663--668, Berlin, Heidelberg. Springer-Verlag.

Potthast, M.; Stein, B. & Holfeld, T. (2010). Overview of the 1st International Competition on Wikipedia Vandalism Detection. In Braschler, M.; Harman, D. & Pianta, E., editores, *CLEF (Notebook Papers/LABs/Workshops)*.

Priedhorsky, R.; Chen, J.; Lam, S. T. K.; Panciera, K.; Terveen, L. & Riedl, J. (2007). Creating, destroying, and restoring value in wikipedia. In *Proceedings of the 2007 international ACM conference on Supporting group work*, GROUP '07, pp. 259--268, New York, NY, USA. ACM.

Seaward, L. (2010). Submission to the 1st international competition on wikipedia vandalism detection. In *CLEF (Notebook Papers/LABs/Workshops)*.

Silva, R.; Gonçalves, M. A. & Veloso, A. (2011). Rule-based active sampling for learning to rank. In *Proceedings of the 2011 European conference on Machine learning and knowledge discovery in databases - Volume Part III*, ECML PKDD'11, pp. 240--255, Berlin, Heidelberg. Springer-Verlag.

Smets, K.; Goethals, B. & Verdonk, B. (2008). Automatic Vandalism Detection in Wikipedia: Towards a Machine Learning Approach. In *Proceedings of the Association for the Advancement of Artificial Intelligence (AAAI) Workshop on Wikipedia and Artificial Intelligence: An Evolving Synergy (WikiAI08)*, pp. 43--48. AAAI Press.

Veloso, A.; Meira Jr., W. & Zaki, M. J. (2006). Lazy associative classification. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pp. 645--654, Washington, DC, USA. IEEE Computer Society.

Veloso, A. A.; Almeida, H. M.; Gonçalves, M. A. & Meira Jr., W. (2008). Learning to rank at query-time using association rules. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '08, pp. 267--274, New York, NY, USA. ACM.

Viégas, F. B.; Wattenberg, M. & Dave, K. (2004). Studying cooperation and conflict between authors with history flow visualizations. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '04, New York, NY, USA. ACM.

Wang, W. Y. & McKeown, K. (2010). "Got You!": Automatic Vandalism Detection in Wikipedia with Web-based Shallow Syntactic-Semantic Modeling. In Huang, C.-R. & Jurafsky, D., editores, *COLING*. Tsinghua University Press.

White, J. & Maessen, R. (2010). Zot! to wikipedia vandalism - lab report for pan at clef 2010. In *CLEF (Notebook Papers/LABs/Workshops)*.

Wikipedia (2011a). The motivation of a vandal. `http://en.wikipedia.org/wiki/Wikipedia:The_motivation_of_a_vandal`.

Wikipedia (2011b). Web 2.0. `http://pt.wikipedia.org/wiki/Web_2.0`.

Wikipedia (2012a). Machine learning. `http://en.wikipedia.org/wiki/Machine_learning`.

Wikipedia (2012b). Statistics. `http://en.wikipedia.org/wiki/Special:Statistics`.