Mineração de Textos para Apoiar a Predição de Severidade de Relatórios de Incidentes: um Estudo de Viabilidade

Jacson Rodrigues Barbosa[1,4], **Ivone Penque Matsuno**[1,3], **Eduardo R. Guimarães**[4], **Solange Oliveira Rezende**[1], **Auri M. R. Vincenzi**[2], **Márcio E. Delamaro**[1]

[1]Instituto de Ciências Matemáticas e de Computação (ICMC)
Universidade de São Paulo (USP) - São Carlos/SP, Brasil, 13560-970

[2]Departamento de Computação (DC) - Universidade Federal de São Carlos (UFSCar)
São Carlos/SP, Brasil, 13565-905

[3]Universidade Federal de Mato Grosso do Sul (UFMS)
Três Lagoas/MS, Brasil, 79620-080

[4]Instituto de Informática (INF) - Universidade Federal de Goiás (UFG)
Goiânia/GO, Brasil, 74690-900

{jacsonrb, ivone.matsuno}@usp.br, eduardo.humberto@ufg.br

{solange, delamaro}@icmc.usp.br, auri@dc.ufscar.br

Abstract. Context: *Due to a large number of incident reports that are persistent in Bug Tracking Systems repositories and the need to prioritize them according to the type of severity, it is necessary to investigate tools that support the prediction of incident reports severity.* ***Objective:*** *Apply Text Mining (TM) techniques and learning methods to help the prediction of incident reports severity from their descriptions.* ***Method:*** *A viability study was conducted to evaluate the application of preprocessing techniques and classification methods.* ***Results:*** *The semi-supervised learning method TCBHN presented good performance concerning the other approaches.* ***Conclusion:*** *The use of two-way heterogeneous networks and semi-supervised classification methods for predicting the severity of incident reports are promising.*

Resumo. Contexto: *Devido à grande quantidade de relatórios de incidentes que são persistidos em Sistema de Rastreamento de Incidentes (SRI) e a necessidade em priorizá-los conforme o tipo de severidade, faz-se necessário investigar ferramentas que apoiem a predição de severidade de relatórios de incidentes.* ***Objetivo:*** *Aplicar técnicas de Mineração de Textos (MT) e métodos de aprendizado para apoiar a predição de severidade de relatórios de incidentes a partir das descrições dos mesmos.* ***Método:*** *Um estudo de viabilidade foi conduzido para avaliar a aplicação de técnicas de pré-processamento e métodos de classificação.* ***Resultados:*** *O método de aprendizado semissupervisionado TCBHN apresentou bom desempenho em relação às demais abordagens.* ***Conclusão:*** *Utilização de redes heterogêneas bipartidas e métodos de classificação semissupervisionados para predição de severidade de relatórios de incidentes são promissores.*

1. Introdução

Em todas as fases do ciclo de vida de desenvolvimento de software é comum identificar defeitos, tendo em vista que 50-80% do custo total da manutenção de software está associado com o custo para a correção de defeitos [Xia et al. 2015]. Muitos projetos de software, para apoiar a gestão desses relatórios de incidentes, fazem uso de Sistema de Rastreamento de Incidentes (SRI), tais como o Bugzilla[1], Jira[2], Mantis[3] e outros.

O registro de um relatório de incidente é, em geral, feito por um ser humano ao detectar que existe algum problema com o produto em uso. Como a análise é conduzida de maneira subjetiva por diferentes pessoas, a severidade e/ou prioridade do defeito reportado acaba sendo sub ou superestimada, dificultando a priorização de quais problemas deveriam ser resolvidos primeiro. Devido ao grande número de relatórios de incidentes produzidos diariamente, há um grande desperdício de recursos humanos que devem ser alocados para reorganizar a prioridade e severidade dos diversos defeitos de forma manual.

Diante deste cenário, diversas técnicas automáticas têm sido propostas e usadas com o intuito de reduzir o impacto dos defeitos em softwares. Dentre essas técnicas têm-se: atribuição de severidade/prioridade de relatório de incidente, detecção de relatório de incidente duplicados e predição do tempo de correção de defeitos [Xia et al. 2015]. Comumente para construir os correspondentes modelos de predição dessas técnicas, faz-se necessário primeiramente extrair os dados a partir de um repositório de SRI.

Conforme Shull et al. (2001), o estudo conduzido é classificado como Estudo de Viabilidade [Shull et al. 2001]. No presente estudo, é proposta uma abordagem para viabilizar a predição de severidade de relatório de incidente a partir de técnicas de Mineração de Textos (MT).

Este texto está organizado da seguinte forma. Na Seção 2, apresenta-se a fundamentação teórica. Nas Seções 3 e 4, são apresentadas respectivamente a definição e a implementação do estudo de viabilidade. Na Seção 5, discutem-se os resultados obtidos e as ameaças à validade deste. Os trabalhos relacionados são discutidos na Seção 6. Por fim, as conclusões e os trabalhos futuros são descritos na Seção 7.

2. Mineração de Repositórios de Relatórios de Incidentes - Visão Geral

Nesta seção são apresentados os principais conceitos relacionados a este trabalho.

2.1. Relatório de Incidente

Sistema de Rastreamento de Incidentes (SRI) é uma importante ferramenta que viabiliza o gerenciamento de relatórios de incidentes. Qualquer interessado (*stakeholder*) ao identificar uma falha durante a execução de um software pode fazer uso do SRI para registro do incidente ocorrido. Durante o registro, dependendo do SRI podem ser inseridas algumas informações cadastrais sobre o defeito. Por exemplo, no Bugzilla pode ser fornecido: *Summary* (definição do título), *Product* (produto no qual deu origem ao registro), *Component* (componente relacionado à ocorrência), *Description* (descrição

[1]Bugzilla: `https://www.bugzilla.org/`
[2]Jira: `https://jira.atlassian.com/`
[3]Mantis: `https://www.mantisbt.org/`

detalhada do defeito), *Priority* (define a prioridade de correção de um defeito em relação aos demais, sendo que P1 é considerada a maior prioridade e P5 a menor), *Severity* (descreve o impacto do defeito, situações possíveis são apresentadas na Tabela 1) e *Status* (estado atual).

Tabela 1. Tipos de Severidade de relatório de incidente, adaptada de [Saha et al. 2015]

Severidade	Descrição
blocker – BL	Bloqueia o desenvolvimento e/ou a atividade de teste de software
critical – CR	Provoca perda de dados, *crashes* ou comprometimento da memória
major – MA	Ocasiona maior perda das funcionalidades
normal – NO	Causa alguma perda de funcionalidade em circunstâncias específicas
minor – MI	Resulta em menor perda de funcionalidade
trivial – TR	Problema elementar, tais como texto desalinhado ou erro ortográfico

Na Figura 1 é exibido o ciclo de vida do relatório de incidente no Bugzilla. Após consolidada a submissão do relatório de incidente no referido SRI, as atividades de verificação da validade do mesmo e definição de quem irá tratar o relatório de incidente é do responsável pela triagem [Zhou et al. 2014].

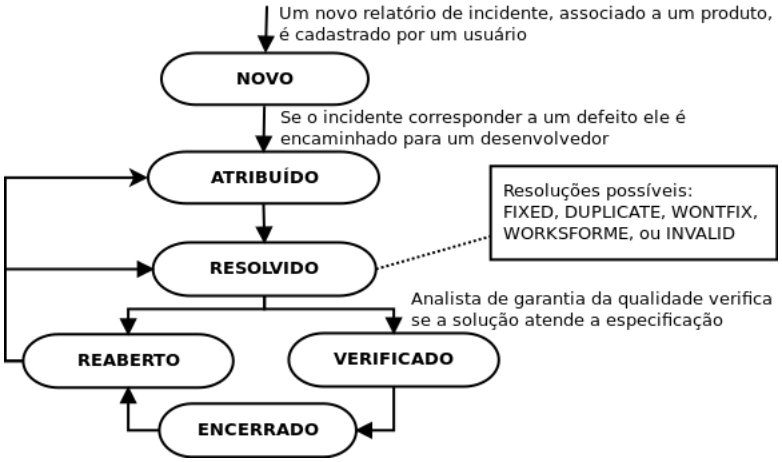


Figura 1. Ciclo de Vida do relatório de incidente no Bugzilla [Zhang et al. 2016].

2.2. Mineração de Repositório de SRI

Mineração de Repositório de SRI (MRSRI) é semelhante ao processo de mineração de dados. As principais atividades e objetos da MRSRI são apresentadas na Figura 2. As atividades correspondem aos retângulos arredondados e os objetos às bases cilíndricas.

A atividade de pré-processamento corresponde à primeira fase da MRSRI, sendo que um ou mais tipos de SRI, podem ser fornecidos como entrada para viabilizar a coleta de dados (relatórios de incidentes) [Jung et al. 2012]. Finalizada a coleta dos dados, atividades de pré-processamento são conduzidas para transformar os dados em uma representação adequada para a extração de padrões. Por exemplo, ao pré-processar os relatórios de incidentes (dados em linguagem natural no formato texto) faz-se necessário conduzir um conjunto de passos:

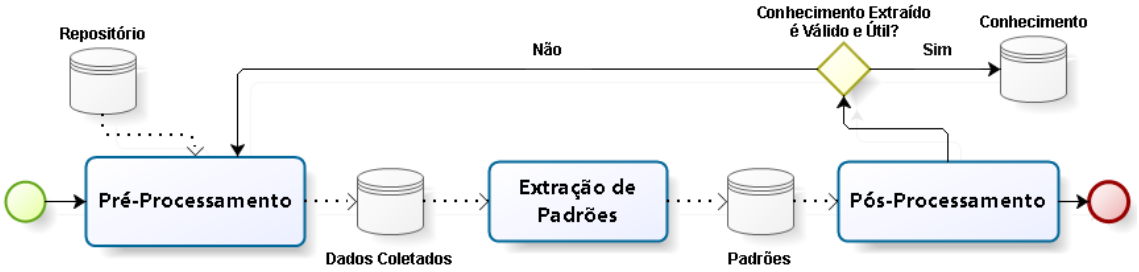


Figura 2. Fluxo Geral da Mineração de Repositório de SRI.

- *Tokenization*: objetiva remover caracteres numéricos e sinais de pontuação.
- Remoção das *stop-words*: preposição, advérbios e outras estruturas (definidas como *stop-words*) comumente utilizadas para apoiar a construção de frases na linguagem humana (a, de, para, até), em geral não agregam informação no contexto de algoritmos de mineração de relatórios de incidentes [Lamkanfi et al. 2011]. Em razão disso, a partir de um conjunto de *stop-words* definido, todas as *stop-words* presentes no texto em processamento são removidas.
- Radicalização (*Stemming*): visa reduzir cada palavra do texto ao correspondente radical (denotação mínima e não ambígua do termo). Por exemplo, as palavras *"mostram"*, *mostrar"* e *"mostrado"* podem ser reduzidas para *mostr"*.
- Definição da representação dos dados: um formato adequado dos documentos textuais necessita ser definido para viabilizar a análise, em geral é construído uma *bag-of-words* (representação no modelo espaço vetorial) para atender esse objetivo. Na qual cada linha representa um documento (relatório de incidente) e cada coluna representa um atributo.

Para extrair padrões utilizam-se métodos de aprendizado de máquina (AM) que podem ser divididos em três tipos: supervisionado, não supervisionado e semissupervisionado. A principal diferença entre eles é o conjunto de exemplos para treinamento. No aprendizado supervisionado os exemplos do conjunto de treinamento são rotulados, ou seja, são classificados previamente. Novas ocorrências serão classificadas com base no que foi apreendido no conjunto de treinamento. Já no aprendizado não supervisionado não existe rótulo pré-definido para os exemplos do conjunto de treinamento. A vantagem desse tipo de aprendizado é não depender de informações rotuladas, porém os erros são maiores. No aprendizado supervisionado a acurácia, em geral, é maior, entretanto existem cenários em que é difícil ter uma quantidade suficiente de exemplos rotulados para gerar um bom modelo de classificação.

O aprendizado semissupervisionado também considera um conjunto de exemplos rotulados, porém a quantidade de exemplos em que os rótulos são conhecidos é bem inferior e, neste tipo de aprendizado, são necessários métodos específicos para tratar este cenário. O objetivo do aprendizado semissupervisionado é fazer uso dos exemplos não rotulados para melhorar o desempenho da classificação. A forma como os exemplos não rotulados são tratados no aprendizado semissupervisionado podem resultar em classificação superior a do aprendizado supervisionado, considerando a mesma quantidade de exemplos rotulados, ou ainda um desempenho equivalente,

considerando uma menor quantidade de exemplos rotulados [Zhu and Goldberg 2009, Chapelle et al. 2006].

Por fim, na atividade de pós-processamento o conhecimento produzido é avaliado de acordo com métricas específicas. Caso o conhecimento produzido não seja adequado para utilização, as atividades de pré-processamento serão retomadas com o intuito de melhorar a qualidade do conhecimento produzido.

2.3. Predição de Severidade de Relatório de Incidente

O processo de atribuição de um rótulo pré-definido à uma instância de dados é conhecido como classificação automática de documentos, que se trata de uma importante subárea da mineração de dados [Lamkanfi et al. 2011]. Por exemplo, em uma determinada empresa, ao receber *e-mail* de um cliente, o mesmo é classificado e encaminhado automaticamente para o departamento (financeiro, assistência técnica ou pessoal) mais adequado para atender à solicitação do cliente. A representação da função de classificação de documentos é definida da seguinte maneira:

$$f : Documento \rightarrow \{r_1, \cdots, r_q\} \quad (1)$$

sendo que no contexto desse estudo, o documento corresponde a um exemplo de relatório de incidente e $\{r_1, ..., r_q\}$ aos rótulos pré-definidos, ou seja, o tipo de severidade. Esse processo de classificação de relatório de incidente com relação ao tipo de severidade, é também conhecido como predição de severidade de relatório de incidente. Existem dois tipos de predição de severidade: binária (por exemplo, severo ou não severo) e não binária (por exemplo, *blocker*, *critical*, *major*, *minor*, *normal* ou *trivial*).

Segundo estudo secundário no contexto da predição de severidade de relatório de incidente conduzido pelos autores, os estudos primários selecionados que investigaram o desempenho de métodos de classificação para apoiar a predição utilizaram, em sua maioria, métodos supervisionados, não existindo, assim, estudos que investigam o impacto de métodos semissupervisionados que sejam de conhecimento dos autores.

3. Definição do Estudo de Viabilidade

Para avaliar a aplicabilidade de métodos semissupervisionados para Predição de Severidade de relatórios de incidentes, foi proposto o referido estudo para um conjunto de textos obtidos de repositórios de software instanciando cada etapa do processo de mineração de textos conforme será apresentado nas subseções seguintes.

3.1. Definição da Questão de Pesquisa

Em particular, pretende-se por meio deste estudo responder à seguinte questão de pesquisa:

- **QP1**: Os métodos de aprendizado semissupervisionado são adequados para apoiar a predição de severidade de relatório de incidente?

3.2. Seleção do *Dataset*

Neste trabalho, utilizaram-se os dados do repositório SRI disponibilizado por [Lamkanfi et al. 2013]. Sendo que foram considerados como dados a serem processados apenas as descrições detalhadas dos relatórios de incidentes. Na Tabela 2 são

apresentadas, de forma resumida, as informações das coleções de textos utilizadas. São apresentadas a descrição do software e a quantidade de documentos (relatório de incidente) de cada um deles. Deste total de documentos, também é apresentada a quantidade de documentos referentes a cada tipo de classe de cada coleção. Essa informação também é importante para avaliar se a quantidade de exemplos rotulados e/ou o desbalanceamento podem impactar no desempenho da classificação. Os textos pré-processados e outras informações estão disponíveis em `http://sites.labic.icmc.usp.br/msr-bsp`.

Tabela 2. Descrições das Coleções de Textos Utilizadas

Dataset	Software	nº docs	Distribuição das Classes por Severidade					
			BL	*CR*	*MA*	*NO*	*MI*	*TR*
$D1$	Eclipse-CTD	5640	78	166	490	275	4547	84
$D2$	Eclipse-JDT	10814	94	274	1000	781	8306	359
$D3$	Eclipse-PDE	5655	47	117	476	208	4693	114
$D4$	Eclipse-Platform	24775	415	989	2718	1088	18891	674
$D5$	Mozilla-Bugzilla	4616	275	176	506	766	2478	415
$D6$	Mozilla-Firefox	69879	233	6603	9486	47635	4145	1777
$D7$	Mozilla-Thunderbird	19237	65	1894	2982	1415	12429	452

3.3. *Design* do Estudo de Viabilidade

3.3.1. Pré-processamento

Na etapa de pré-processamento são realizadas as seguintes tarefas: preparação dos documentos, extração de termos e seleção de atributos e geração de uma representação estruturada da coleção de documentos apropriada para os métodos de extração de padrões utilizados. Neste trabalho, na preparação de documentos, foram realizadas padronização, remoção das *stopwords*, radicalização dos termos (*stemming*) e foram selecionados como termos os unigramas com frequência maior do que um.

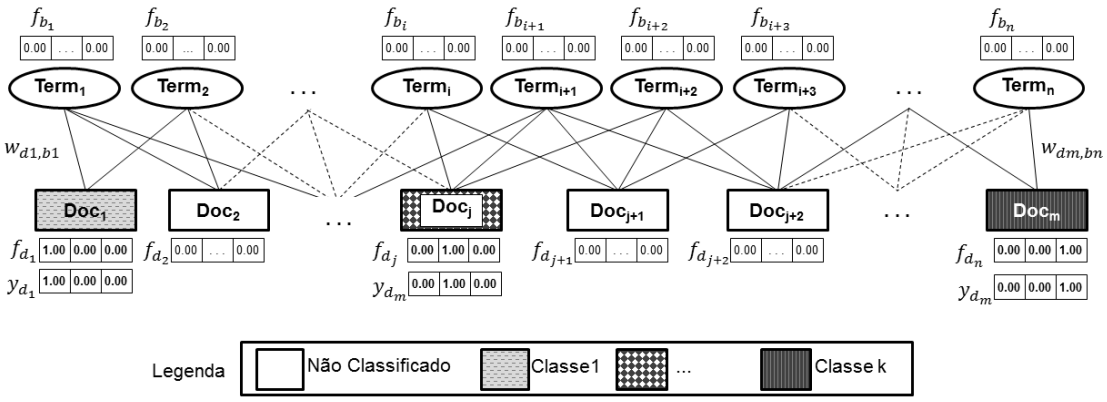
Em relação à representação da coleção de textos foram utilizadas duas representações: (i) modelo espaço vetorial, *bag-of-words* (*bow*) [Tan et al. 2005], comumente utilizada e (ii) modelo de redes heterogêneas bipartidas [Rossi et al. 2016]. *Bag-of-words* é uma matriz de documento-termo, em que cada linha representa um documento, cada coluna representa um termo (palavra) presente na coleção de documentos e cada célula contém uma medida de frequência da palavra no respectivo documento. Um exemplo é apresentado na Figura 3(a).

Uma rede heterogênea bipartida pode ser definida por $N = (\mathcal{O}, \mathcal{E}, \mathcal{W})$, em que $\mathcal{O}$ representa dois conjuntos de objetos da rede (também denominados por nós ou vértices), $\mathcal{E}$ representa o conjunto de conexões (também denominadas relações ou arestas) que ocorrem apenas a partir de objetos de um conjunto para os objetos do outro conjunto, e $\mathcal{W}$ representa os pesos dessas conexões. Neste estudo $\mathcal{O}$ é composto pelos conjuntos de documentos $\mathcal{D}$ e conjunto de termos $\mathcal{T}$. $\mathcal{E}$ é composto por conexões $\mathrm{e_{ij}}$ que representam que o termo $\mathrm{t_j}$ está presente no documento $\mathrm{d_i}$, $0 < i \leq |\mathcal{D}|$ e $0 < j \leq |\mathcal{T}|$. Na Figura 3(b) pode ser observado um exemplo da representação usando rede bipartida heterogênea.

Neste estudo de viabilidade, cada descrição de relatório de incidente extraída dos SRI é considerada um documento e cada palavra é considerada um termo.

	$Term_1$	$Term_2$	...	$Term_{n-2}$	$Term_{n-1}$	$Term_n$	Classe
Doc_1	1	1	...	0	0	0	C_1
Doc_2	0	0	...	1	1	0	C_2
Doc_3	0	1	...	1	0	0	...
...							...
Doc_m	1	1	...	0	1	0	C_k

(a) *Bag-of-words* (*bow*)



(b) Redes heterogêneas bipartidas

Figura 3. Representações de Textos

3.3.2. Extração de Padrões

Nessa etapa, foram utilizados os principais métodos de aprendizado de máquina supervisionados e semissupervisionados. Dos métodos supervisionados foram utilizados um de cada tipo de abordagem: probabilística, baseada aprendizado estatístico, árvores de decisão e distância.

- ***Naïve Bayes*** **(NB)** [Rish 2001]: este método baseia-se no teorema de Bayes [Koch 1990] para identificar a classe para a qual um exemplo x tem a maior probabilidade de estar associado, como dado pela Equação 2.

$$y = arg\,max_i P(y_i|x) \tag{2}$$

Para cada termo x é calculada a probabilidade de pertencer a uma determinada categoria y_i. Essa probabilidade $P(y_i|x)$ é calculada a partir das ocorrências do termo x nos documentos de treinamento nos quais as categorias já são conhecidas. Quando todas essas probabilidades são calculadas, um novo documento pode ser classificado de acordo com a soma das probabilidades para cada categoria de cada termo ocorrido dentro do documento. $arg\,max_i$ retorna a classe com maior probabilidade de estar associada ao termo x. A presença ou ausência de um termo em um documento textual pode determinar a predição da categoria.

- ***Multinomial Naïve Bayes*** **(MNB)**: este classificador é baseado no anterior, porém a categoria não é determinada apenas pela presença ou ausência de um termo no documento, mas também pelo número de ocorrências dos termos no documento.
- ***Support Vector Machines*** **(SVM)**: este método é baseado em aprendizado estatísticos desenvolvido por [Vapnik 1995] que estabelece uma série de princípios que devem ser seguidos na obtenção de classificadores com boa capacidade de generalização. O resultado deste classificador são hiperplanos entre vetores de atributos que separam o espaço em várias categorias.
- **J48**: algoritmo C.45 [Quinlan 1993], este método é baseado em árvores de decisão que usa a estratégia de divisão e conquista para segmentar recursivamente o espaço de busca em subespaços, e cada subespaço é ajustado usando diferentes modelos. As árvores de decisões são simples de entender e interpretar. No entanto, podem criar árvores tendenciosas se algumas classes dominarem outras.
- k***-Nearest Neighbor*** **(**k**NN)**: este método é baseado em distâncias entre objetos, a classificação de um novo objeto é feita considerando os exemplos do conjunto de treinamento mais próximos a ele. A variação nesse algoritmo é a quantidade de vizinhos K a serem considerados.

Os métodos semissupervisionados buscam a regularização da rede considerando duas suposições: (i) dois objetos conectados na rede tendem a ser classificados com o mesmo rótulo e (ii) os rótulos dos objetos devem estar próximos da informação da classe real (conjunto de treinamento). A seguir são apresentados os algoritmos para realizar a regularização em redes heterogêneas bipartidas utilizadas neste estudo de viabilidade, com suas respectivas funções de regularização. Em cada função de regularização o primeiro termo está relacionado com a primeira suposição e, analogamente, o segundo termo descreve a segunda suposição.

- **GNetMine** [Ji et al. 2010]: este é uma extensão do algoritmo *Learning with Local and Global Consistency* (LLGC) [Zhou et al. 2004]. A função de regularização a ser minimizada por GNetMine é definida pela Equação 3.

$$Q(\mathbf{F}) = \sum_{t_i \in \mathcal{T}} \sum_{b_j \in \mathcal{B}} w_{t_i,b_j} \left\| \frac{\mathbf{f}_{t_i}}{\sqrt{\sum_{b_k \in \mathcal{B}} w_{t_i,b_k}}} - \frac{\mathbf{f}_{b_j}}{\sqrt{\sum_{t_k \in \mathcal{T}} w_{t_k,b_j}}} \right\|^2 + \sum_{t_i \in \mathcal{T}^L} \alpha_{t_i} (\mathbf{f}_{t_i} - \mathbf{y}_{t_i}), \quad (3)$$

  em que $0 < \alpha < 1$ dá a importância para cada termo da Equação 3.

- ***Label Propagation based on Bipartite Heterogeneous Network*** **(LPBHN)** [Rossi et al. 2016]: este algoritmo é uma extensão do algoritmo *Gaussian Fields e Harmonic Function* (GFHF) [Zhu and Goldberg 2009] para redes heterogêneas bipartidas. Este algoritmo é livre de parâmetro. A função de regularização a ser minimizada por LPBHN é definida pela Equação 4.

$$Q(\mathbf{F}) = \frac{1}{2} \sum_{t_i \in \mathcal{T}} \sum_{b_j \in \mathcal{B}} w_{t_i,b_j} (\mathbf{f}_{t_i} - \mathbf{f}_{b_j})^2 + \lim_{\mu \to \infty} \mu \sum_{t_i \in \mathcal{T}^L} (\mathbf{f}_{t_i} - \mathbf{y}_{t_i})^2 \quad (4)$$

  há uma restrição que $\mathbf{f}_{t_i} = \mathbf{y}_{\mathbf{t_i}}$, portanto, o segundo termo da Equação 4 tem um valor que tende ao infinito.

- ***Tag-based Model*** **(TM)** [Yin et al. 2009]: este algoritmo foi proposto inicialmente para classificar objetos da web conectados a tags sociais. No contexto

deste trabalho, a função de regularização a ser minimizada por TM é é definida pela Equação 5.

$$Q(\mathbf{F}) = \Big(\beta \sum_{b_i \in \mathcal{B}^L} ||\mathbf{f}_{b_i} - \mathbf{y}_{b_i}||^2 + \gamma \sum_{B_i \in \mathcal{B}^U} ||\mathbf{f}_{b_i} - \mathbf{y}_{b_i}||^2\Big) + \Big(\sum_{b_i \in \mathcal{B}} \sum_{t_j \in \mathcal{T}} w_{b_i,t_j} ||\mathbf{f}_{b_i} - \mathbf{f}_{t_j}||^2\Big), \quad (5)$$

em que cada parâmetro β e γ controla a importância dada ao termo da Equação 5.

- ***Transductive Classification based on Bipartite Heterogeneous Network* (TCBHN)** [Rossi et al. 2016]: este é um algoritmo que executa otimização e propagação de rótulo para minimizar a seguinte função de regularização é definida pela Equação 6.

$$Q(\mathbf{F}) = \frac{1}{2}\left(\sum_{c_k \in \mathcal{C}} \Big(\sum_{t_i \in \mathcal{T}^U} f_{t_i,c_k} - \sum_{b_j \in \mathcal{T}} w_{t_i,b_j} \cdot f_{b_j,c_k}\Big)\right)^2 + \frac{1}{2}\left(\sum_{c_k \in \mathcal{C}} \Big(\sum_{t_i \in \mathcal{T}^L} y_{t_i,c_k} - \sum_{b_j \in \mathcal{B}} w_{t_i,b_j} \cdot f_{b_j,c_k}\Big)\right)^2 \quad (6)$$

3.3.3. Pós-Processamento

Nesta etapa, é conduzida uma avaliação experimental, na qual analisam-se a viabilidade e impacto do uso de aprendizado semissupervisionado na predição de severidade de relatórios de incidentes.

Para comparar os resultados da classificação foi utilizada a medida F^1 que representa a média harmônica das medidas de *precisão* (*Precision*) e *cobertura* (*Recall*), em que ambas as medidas têm o mesmo peso (vide Equação 7).

$$F^1 = 2 * \frac{Precision * Recall}{Precision + Recall}. \quad (7)$$

A *precisão* e a *cobertura* foram calculadas para cada classe na avaliação multi-classe. A fórmula para cálculo da *precisão* e da *cobertura* de uma classe c_i são apresentadas nas Equações 8 e 9, respectivamente:

$$Precision_{c_i} = \frac{TP_{c_i}}{TP_{c_i} + FP_{c_i}}, \quad (8) \qquad Recall_{c_i} = \frac{TP_{c_i}}{TP_{c_i} + FN_{c_i}}, \quad (9)$$

Em que TP (*True Positive*) significa o número de documentos de teste corretamente atribuídos à classe c_i, FP (*False Positive*) significa o número de documentos de teste da classe c_j ($c_j \neq c_i$) mas atribuído à classe c_i, e FN (*False Negative*) é o número de documentos de teste da classe c_i atribuído à classe c_j ($c_j \neq c_i$).

A medida de *precisão* é a porcentagem de documentos corretamente classificados como c_i, considerando todos os documentos classificados como c_i. A medida de *cobertura* é a porcentagem de documentos corretamente classificados como c_i, considerando todos os documentos que realmente pertencem à classe c_i.

Duas estratégias foram utilizadas para resumir os resultados de *precisão* e *cobertura*, calculados para cada classe de uma coleção de texto, são elas: ***micro-averaging*** e ***macro-averaging*** [Sokolova and Lapalme 2009]. A estratégia de

micro-averaging, realiza uma soma dos termos das medidas de avaliação. Portanto, a *precisão* e a *cobertura* com a estratégia de *micro-averaging* são:

$$Precision^{Micro} = \frac{\sum_{c_i \in \mathcal{C}} TP_{c_i}}{\sum_{c_i \in \mathcal{C}} (TP_{c_i} + FP_{c_i})}, \quad (10) \qquad Recall^{Micro} = \frac{\sum_{c_i \in \mathcal{C}} TP_{c_i}}{\sum_{c_i \in \mathcal{C}} (TP_{c_i} + FN_{c_i})}. \quad (11)$$

A estratégia de ***macro-averaging*** realiza a média sobre as medidas de avaliação para cada classe. Portanto, o *precision* e o *recall* com a estratégia de *macro-averaging* são:

$$Precision^{Macro} = \frac{\sum_{c_i \in \mathcal{C}} Precision_{c_i}}{|\mathcal{C}|}, \quad (12) \qquad Recall^{Macro} = \frac{\sum_{c_i \in \mathcal{C}} Recall_{c_i}}{|\mathcal{C}|}. \quad (13)$$

As pontuações de *micro-averaging* são dominadas pelo número de TP. Portanto, classes grandes dominam classes pequenas em pontuações de *micro-averaging*. Por outro lado, a *macro-averaging* atribui igual peso para cada classe. Nesse caso, o número de TP em classes pequenas é enfatizado nas pontuações de *macro-averaging*. Essas duas estratégias atribuem pontuações diferentes e são complementares entre si. Denotam-se F^1 calculado por *micro-averaging*, para *precision* e *recall*, como *Micro-*F^1, e por *macro-averaging* como *Macro-*F^1.

Para obter *Micro-*F^1 e *Macro-*F^1, primeiro é realizado um processo de validação cruzada de 10 execuções. Para cada conjunto de treinamento (9 vezes), foram realizadas 10 execuções para induzir um modelo de classificação, considerando N documentos rotulados, selecionados aleatoriamente, em cada execução. Para analisar o impacto do uso de documentos rotulados, tanto para os algoritmos supervisionados quanto semissupervisionados, considerou-se uma variação absoluta na quantidade de documentos rotulados. Foi considerado $N = \{1, 10, 20, 30, 40, 50\}$ que considera apenas a quantidade exata de 1 exemplo rotulado de cada classe, 10 exemplos rotulados de cada classe, e assim sucessivamente.

Esta variação no número de documentos rotulados possibilitou demonstrar melhor o comportamento dos algoritmos para diferentes números de documentos rotulados, isto é, um *trade-off* entre o número de documentos rotulados e o desempenho da classificação e as diferenças entre os algoritmos de aprendizagem supervisionados indutivos e algoritmos de aprendizagem semissupervisionados à medida que aumenta o número de documentos rotulados.

Os exemplos de treinamento restantes foram considerados como exemplos não rotulados para algoritmos de aprendizagem semissupervisionados. Assim, foram realizadas 100 execuções e em cada execução foi obtido um valor de acurácia. Os valores finais de *Micro-*F^1 e *Macro-*F^1 apresentados na Seção 5 foram uma média dos 100 valores obtidos usando validação cruzada 10-folds.

4. Implementação do Estudo de Viabilidade

4.1. Configuração dos Algoritmos de Aprendizagem Utilizados

Foram executados os algoritmos de aprendizagem supervisionados indutivos, citados na Seção 3.3.2, para análise e comparação com a aprendizagem semissupervisionada.

Isso também permite analisar se o uso de documentos não rotulados, realmente melhora o desempenho da classificação. Para avaliação da execução dos algoritmos supervisionados foram utilizadas as implementações disponibilizadas na ferramenta Weka[4]. Os parâmetros e considerações dos algoritmos indutivos de aprendizagem supervisionada são:

- ***Naïve Bayes* (NB)**: configuração padrão.
- ***Multinomial Naïve Bayes* (MNB)**: configuração padrão.
- ***Support Vector Machine* (SVM)**: foi utilizado o algoritmo *Sequential Minimal Optimization* (SMO) e foram considerados três tipos de *kernel*: Linear, Polinomial (exponente = 2) e RBF (*Radial Basis Function*). Como o parâmetro C é real e positivo, alguns autores definem esses valores como 10^Y. Para cada tipo de *kernel*, foi considerado $Y = \{-5; -4; -3; -2; -1; 0; 1; 2; 3; 4; 5\}$.
- **J48**: neste algoritmo de indução de árvores de decisão, foi utilizado o valor $0,25$ para parâmetro *confidence factor*.
- ***k-NN***: Foi considerado $k = \{7; 17; 37; 57\}$ [Rossi et al. 2016]. E também o algoritmo *k-NN*, com e sem votação ponderada, que atribui para cada um dos vizinhos mais próximos um voto de peso igual a $(1-s)$, em que s é uma medida de similaridade entre vizinhos. Foi utilizado o cosseno como medida de similaridade.

Foram utilizados algoritmos de aprendizagem semissupervisionados baseados no modelo de redes heterogêneas bipartidas. Os algoritmos e seus parâmetros estão em [Rossi et al. 2016]. Foram utilizadas todas as soluções iterativas para todos os algoritmos que possuem soluções iterativas (GNetMine, LPBHN, TCBHN e TM). O número máximo de iterações foi definido para 1000. Os parâmetros e considerações dos algoritmos semissupervisionado utilizados neste estudo de viabilidade são definidos a seguir:

- ***GNetMine***: foi utilizado $\alpha = \{0.1, 0.3, 0.5, 0.7, 0.9\}$.
- ***Label Propagation using Bipartite Heterogeneous Networks* (LPBHN)**: este é um algoritmo de aprendizagem semissupervisionado sem parâmetros.
- ***Tag-based Model* (TM)**: foram utilizados $\beta = \{0.1, 1, 10, 100, 1000\}$, e $\gamma = \{0.1, 1, 10, 100, 1000\}$.
- ***Transductive Categorization based on Bipartite Heterogeneous Networks* (TCBHN)**: foi considerada a solução iterativa de TCBHN, apresentada em [Rossi et al. 2016]. Esta solução utiliza dois parâmetros o η (taxa de correção de erro) e o ϵ (erro quadrático mínimo). Foram utilizados $\eta = \{0.01, 0.05, 0.1, 0.5\}$, $\epsilon = 0.01$, 10 como o número máximo de iterações globais e 100 como número máximo de iterações locais, o que dá um total de 1000 iterações.

5. Resultados Obtidos

Conforme critérios e configurações experimentais definidos anteriormente, os resultados obtidos para cada uma dos conjuntos de dados são apresentados nos gráficos da Figura 4. Nos gráficos as linhas contínuas indicam os resultados dos métodos supervisionados e as linhas tracejadas indicam os métodos semissupervisionados.

Nesse estudo não se destacou um método de aprendizado que produziu resultados superiores em ambas as medidas e em todos os conjuntos de dados. Com exceção, do

[4]Weka: `http://www.cs.waikato.ac.nz/ml/weka/`

método supervisionado SVM que apresentou valores finais de *Micro-*F^1 e *Macro-*F^1 superiores no *dataset* $D7$ (como pode ser observado na Figura 4(g)). Naturalmente, os métodos supervisionados deveriam apresentar melhores resultados, uma vez que possuem mais informações rotuladas. Porém, o processo de rotular tem um custo alto e nem sempre temos dados rotulados suficientes para gerar um bom modelo de classificação. Quando se tem poucos exemplos rotulados, pode-se observar que algoritmos supervisionados não apresentaram um bom desempenho. Nos *datasets* $D2$ (Figura 4(b)), $D3$ (Figura 4(c)) e $D5$ (Figura 4(e)) os métodos semissupervisionados apresentaram resultados de *Micro-*F^1 superiores aos métodos supervisionados quando se tem poucos exemplos rotulados, respondendo assim a questão de pesquisa (QP1).

Os riscos à validade dos resultados desse estudo são classificados em dois tipos:

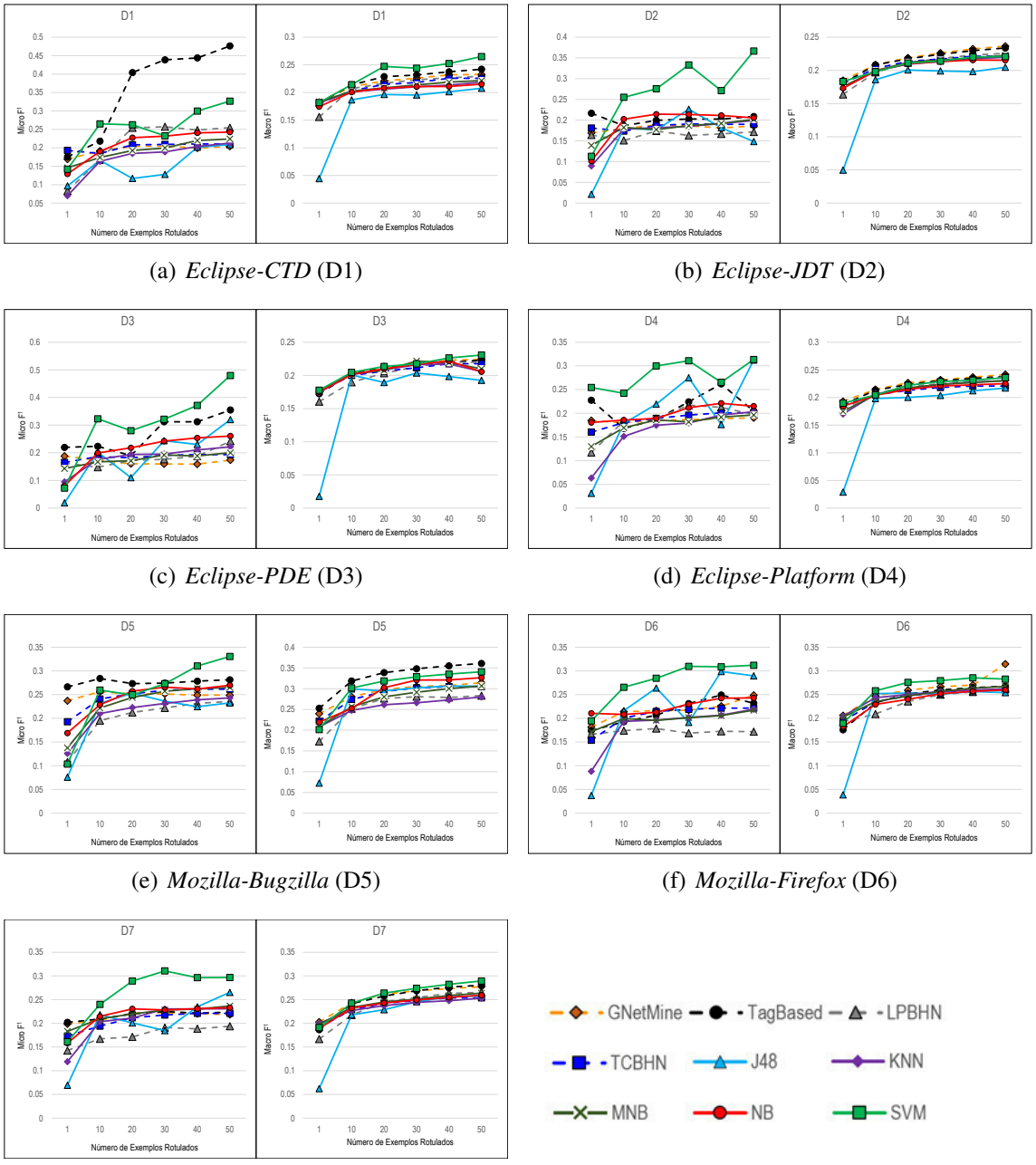
- **Validade interna**: Como foram selecionados apenas relatórios de incidentes com *status* "resolvido" (devido esses representarem relatórios de incidentes que percorreram todo o ciclo de vida), o modelo de predição proposto não considerou relatórios de incidentes imaturos, tais como os recém criados pelos *stakeholders*. Essa restrição pode oferecer impacto no desempenho do modelo proposto.
- **Validade externa**: Os resultados desse estudo não podem ser generalizados para software proprietários, visto que nesse foram analisados somente projetos de software *Open Source*. Tendo em vista que todos os relatórios de incidentes utilizados nesse foram extraídos do repositório Bugzilla, os resultados do presente estudo também não podem ser generalizados para outros SRI.

6. Trabalhos Relacionados

Lamkanfi et al. (2010) propôs o primeiro modelo de predição automática de severidade binária (severo ou não severo) de relatório de incidente, no qual são utilizados algoritmos de mineração de texto que analisam as descrições textuais dos relatórios de incidentes para apoiar a predição de severidade. Os autores concluíram que com pelo menos 500 registros por tipo de severidade pertencentes ao conjunto de treinamento do modelo, é possível predizer a severidade de um relatório de incidente com razoável acurácia [Lamkanfi et al. 2010].

Como evolução do estudo anterior, Lamkanfi et al. (2011) realizaram outro estudo comparativo entre quatro algoritmos de mineração de texto (*Naive Bayes*, *Multinomial Naïve Bayes*, *K-Nearest Neighbor* e *Support Vector Machines*) utilizando a mesma amostra de dados, e identificaram que o algoritmo *Multinomial Naïve Bayes* oferece uma melhor acurácia para a predição binária dos relatórios de incidentes [Lamkanfi et al. 2011]. Neste estudo de viabilidade também utilizou-se a descrição textual dos relatórios de incidentes como dados de entrada para o modelo de predição de severidade, no entanto utilizou-se a representação de dados baseada em redes heterogêneas bipartidas.

Considerando que existem diferentes tipos de severidade de relatório de incidente, Tian et al. (2012) a partir de funções de similaridade de documentos (BM25F e BM25F ext), propuseram um modelo de predição de severidade não binário que considera os cinco principais tipos de severidade (blocker, critical, major, minor e trivial). Por meio do referido modelo proposto, os autores conseguiram obter melhorias comparado com outras propostas de classificação de severidade binária [Tian et al. 2012]. Neste estudo de



(a) *Eclipse-CTD* (D1)

(b) *Eclipse-JDT* (D2)

(c) *Eclipse-PDE* (D3)

(d) *Eclipse-Platform* (D4)

(e) *Mozilla-Bugzilla* (D5)

(f) *Mozilla-Firefox* (D6)

(g) *Mozilla-Thunderbird* (D7)

Figura 4. Resultados

viabilidade avaliou-se apenas a predição de severidade não binária a partir da aplicação de algoritmos de semissupervisionados.

7. Conclusões

O presente trabalho viabilizou um estudo comparativo entre as abordagens de aprendizagem supervisionada e semissupervisionada no contexto da predição de severidade de relatório de incidente. Após analisar os resultados, constata-se que com o uso de redes heterogêneas bipartidas e métodos semissupervisionados foram obtidos resultados promissores no estudo. Considerando que o processo de rotulagem

de relatório de incidente dependendo da quantidade exige muito tempo de dedicação dos desenvolvedores, a combinação entre a representação dos dados e os algoritmos semissupervisionados utilizados demonstrou a aplicabilidade dos mesmos no contexto da predição de severidade de relatório de incidente. Em resumo, ressaltam-se as seguintes contribuições:

- Investigação pioneira ao utilizar algoritmos semissupervisionados para classificar a severidade de relatórios de incidentes.
- Definição de um *baseline* para viabilizar futuras comparações com outras estratégias/algoritmos.
- Disponibilização dos dados e resultados para análise e consulta pública.

No presente estudo, considerou-se o impacto de métodos semissupervisionados apenas na classificação de severidade não binária. Faz-se necessário avaliar o desempenho dos mesmos na classificação binária. Além disso, outras questões podem ser exploradas para melhorar a classificação tais como: (i) avaliar o uso de técnicas de balanceamento das classes; (ii) usar técnicas para enriquecimento das representações de textos; (iii) propor novas representações de textos para o domínio específico na predição de severidade de relatórios de incidentes; (iv) avaliar o tamanho do conjunto de exemplos rotulados em relação ao percentual do tamanho dos *datasets*; (v) avaliar como os exemplos não rotulados podem melhorar o desempenho da classificação; e (vi) utilizar dados de produtos proprietários com o intuito de comparar o comportamento dos modelos de predição utilizados em outro contexto.

Referências

[Chapelle et al. 2006] Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press.

[Ji et al. 2010] Ji, M., Sun, Y., Danilevsky, M., Han, J., and Gao, J. (2010). Graph regularized transductive classification on heterogeneous information networks. In *Proc. of the European Conf. on Machine Learning and Knowledge Discovery in Databases*, pages 570–586. Springer-Verlag.

[Jung et al. 2012] Jung, W., Lee, E., and Wu, C. (2012). A survey on mining software repositories. *IEICE Transactions on Information and Systems*, E95.D(5):1384–1406.

[Koch 1990] Koch, K.-R. (1990). Bayes' theorem. In *Bayesian Inference with Geodetic Applications*, pages 4–8. Springer.

[Lamkanfi et al. 2010] Lamkanfi, A., Demeyer, S., Giger, E., and Goethals, B. (2010). Predicting the severity of a reported bug. In *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*, pages 1–10.

[Lamkanfi et al. 2011] Lamkanfi, A., Demeyer, S., Soetens, Q. D., and Verdonck, T. (2011). Comparing mining algorithms for predicting the severity of a reported bug. In *2011 15th European Conference on Software Maintenance and Reengineering*, pages 249–258.

[Lamkanfi et al. 2013] Lamkanfi, A., Pérez, J., and Demeyer, S. (2013). The eclipse and mozilla defect tracking dataset: a genuine dataset for mining bug information. In *2013 10th Working Conference on Mining Software Repositories (MSR)*, pages 203–206.

[Quinlan 1993] Quinlan, J. R. (1993). *C4.5:Programs for Machine Learning*, volume 1. M.Kaufmann.

[Rish 2001] Rish, I. (2001). An empirical study of the naive bayes classifier. In *IJCAI-Workshop Empirical Methods in Artificial Intelligence*, volume 3, pages 41–46. IBM New York.

[Rossi et al. 2016] Rossi, R. G., Lopes, A. A., and Rezende, S. O. (2016). Optimization and label propagation in bipartite heterogeneous networks to improve transductive classification of texts. *Information Processing & Management*, 52(2):217 – 257.

[Saha et al. 2015] Saha, R. K., Lawall, J., Khurshid, S., and Perry, D. E. (2015). Are these bugs really normal? In *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*, pages 258–268.

[Shull et al. 2001] Shull, F., Carver, J., and Travassos, G. H. (2001). An empirical methodology for introducing software processes. In *Proceedings of the 8th European Software Engineering Conference Held Jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ESEC/FSE-9, pages 288–296, New York, NY, USA. ACM.

[Sokolova and Lapalme 2009] Sokolova, M. and Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427–437.

[Tan et al. 2005] Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining*. Addison-Wesley.

[Tian et al. 2012] Tian, Y., Lo, D., and Sun, C. (2012). Information retrieval based nearest neighbor classification for fine-grained bug severity prediction. In *19th Working Conference on Reverse Engineering*, pages 215–224.

[Vapnik 1995] Vapnik, V. N. (1995). The nature of statistical learning theory.

[Xia et al. 2015] Xia, X., Lo, D., Shihab, E., Wang, X., and Yang, X. (2015). Elblocker: Predicting blocking bugs with ensemble imbalance learning. *Information and Software Technology*, 61:93 – 106.

[Yin et al. 2009] Yin, Z., Li, R., Mei, Q., and Han, J. (2009). Exploring social tagging graph for web object classification. In *Proc. of the Int. Conf. on Knowledge Discovery and Data Mining*, pages 957–966.

[Zhang et al. 2016] Zhang, T., Chen, J., Yang, G., Lee, B., and Luo, X. (2016). Towards more accurate severity prediction and fixer recommendation of software bugs. *J. Syst. Softw.*, 117(C):166–184.

[Zhou et al. 2004] Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In *Advances in Neural Information Processing Systems*, volume 16, pages 321–328.

[Zhou et al. 2014] Zhou, Y., Tong, Y., Gu, R., and Gall, H. (2014). Combining text mining and data mining for bug report classification. In *2014 IEEE International Conference on Software Maintenance and Evolution*, pages 311–320.

[Zhu and Goldberg 2009] Zhu, X. and Goldberg, A. B. (2009). *Introduction to semi-supervised learning*. Morgan and Claypool Publishers.