MODELO DE CLASSIFICAÇÃO SEMIAUTOMÁTICO PARA SERVIÇOS DE TI

ELLEN GERA DE BRITO MOURA∗, RAIMUNDO SANTOS MOURA∗

∗*Departamento de Computação*
Universidade Federal do Piauí
Teresina, Piauí, Brasil

E-mails: ellengera@gmail.com, rsm@ufpi.edu.br

Abstract— Call-Center services (IT services) are classified in order to categorize, prioritize, and identify solutions for similar cases. The messages can also be used to define the area and technician responsible for servicing the order. Considering enterprise environment, a correct classification optimizes resources and minimizes negative impacts to the business. Services are commonly opened by IT users and written in natural language, through specific software for management of the so-called life cycle. This paper presents a semi-automatic classification model for IT services using natural language processing techniques and supervised machine learning algorithms. A semantic network is used to define conceptual features.

Keywords— Intelligent Systems Applications, Machine Supervised Learning, Natural Language Processing.

Resumo— Serviços de *call-center* (ou *service desk*) são classificados a fim de categorizar, priorizar e identificar soluções para casos semelhantes. As mensagens também podem ser utilizadas para definir a área e o técnico responsável pela manutenção do chamado. Considerando ambiente corporativo, a classificação correta otimiza os recursos e minimiza impactos negativos para o negócio. Serviços normalmente são abertos por usuários de TI e escritos em linguagem natural, através de um software específico para gestão do ciclo de vida do chamado. Este artigo apresenta um modelo de classificação semiautomático para serviços de TI, utilizando técnicas de processamento de linguagem natural e algoritmos de aprendizagem de máquina supervisionada. Uma rede semântica foi utilizada para a definição de *features* conceituais.

Palavras-chave— Sistemas Inteligentes, Aprendizagem de Máquina Supervisionada, Processamento de Linguagem Natural.

1 Introdução

Empresas que fazem uso de tecnologia da informação (TI) necessitam do suporte de equipes especializadas para garantir a disponibilidade e manutenção desses recursos. De acordo com Hochstein et al. (2005), o conjunto mais popular das melhores práticas de gerenciamento de serviços de TI, conhecido pelo acrônimo ITIL (do inglês, *Information Technology Infrastructure Library*), orienta que estas equipes de suporte estejam organizadas em uma central de serviços (*service desk*), definida como um ponto único de contato entre usuários e os prestadores de serviços de TI.

Usuários acionam o *service desk* por meio de chamados técnicos. De acordo com documentos recomendados pelo ITIL, esses chamados podem ser classificados, de forma geral, como: *i*) incidentes, quando provocam uma interrupção ou redução na qualidade de entrega de um serviço; ou *ii*) requisição de serviços, quando o usuário solicita que algo lhe seja fornecido, como por exemplo, instalar um software ou efetuar um cadastro na rede. Em virtude da diversidade e severidade dos problemas de TI, outras subclassificações podem ser aplicadas para garantir uma gestão mais efetiva dos chamados. Uma lista de classes mais específicas é importante, pois permite as seguintes ações: a) identificar o item de serviço afetado pelo chamado; b) selecionar a área (hardware ou software) ou o técnico adequado para atender o chamado; c) contribuir com a redução do tempo de diagnóstico do chamado através de comparações com casos similares; d) ordenar a prioridade dos chamados abertos de acordo com impacto e urgência das classes; e e) contribuir para reduzir o tempo de atendimento do chamado.

Embora e-mail e chamadas por telefone possam ser usados como meio para abertura de chamados, softwares específicos são ferramentas mais adequadas, pois permitem o controle do ciclo de vida completo do chamado. O uso comum destes softwares possibilita que o usuário descreva e registre textualmente o chamado. O *service desk* monitora estas solicitações, analisa os dados e classifica o chamado. Em outra abordagem, é possível que o próprio usuário classifique o chamado. Porém, devido à importância técnica desta ação, é conveniente que esta classificação não seja atribuída aos usuários finais de TI. Destaca-se que a interferência humana na classificação representa um ponto de espera no processo de atendimento e obriga que técnicos especializados sejam destacados para atividades de recepção, classificação, priorização e encaminhamento de chamados, representando um custo adicional à equipe técnica.

Este artigo propõe um modelo para classificação semiautomática de serviços (ou chamados técnicos) de TI, desenvolvido a partir de aprendizagem de máquina supervisionada e processamento de linguagem natural (PLN). Com o classificador é possível reduzir a dependência humana no processo de classificação dos chamados e atividades de priorização, seleção de técnico para atendimento, diagnóstico e listagem de possíveis soluções.

Os experimentos foram realizados sobre um *corpus* de chamados técnicos de TI cedidos por um

órgão público do Estado do Piauí, com *service desk* implantado. Os chamados passaram por técnicas de PLN, até serem representados como *features* conceituais, tendo passado por um processo de anotações semânticas para possibilitar a extensão das informações já existentes. O *corpus* utilizado foi obtido devidamente rotulado. Dessa forma, optou-se por aprendizagem de máquina supervisionada. *Naive Beyes*, SMO e J48 foram os algoritmos aplicados na validação do modelo.

As próximas seções deste artigo estão organizadas da seguinte forma: a Seção II discute trabalhos relacionados à classificação automática de entradas similares a chamados técnicos de TI e aplicação de algoritmos de aprendizagem de máquina supervisionada. A Seção III apresenta a arquitetura do modelo de classificação proposto neste trabalho. Os resultados e detalhamento dos experimentos são descritos na Seção IV, e, por fim, a Seção V conclui o artigo e apresenta trabalhos futuros.

2 Trabalhos Relacionados

Ceci et al. (2014) definem a classificação de documentos como uma área que tem como função a identificação e atribuição de temas, assuntos ou conceitos pré-definidos para um documento. Neste contexto, utiliza-se como base o conteúdo textual do documento para uma possível interpretação e classificação a partir de um conjunto de rótulos. Esta é uma tarefa típica para aplicações de PLN. Existe uma vasta quantidade de trabalhos realizados sobre esse tema. A seguir, destacamos alguns pela estreita relação com a pesquisa apresentada neste artigo.

Um desafio comum nos trabalhos que exploram classificação a partir de entradas textuais em linguagem natural é a necessidade de transformar o texto em representações estruturadas. Para este fim, uma etapa inicial de pré-processamento pode ser utilizada, conforme é observado em Javed et al. (2012). Eles propõem uma abordagem automatizada para classificação de falhas de software registradas textualmente em repositórios abertos na web (do inglês, *open source bug repositories*). A etapa de pré-processamento textual utiliza a abordagem de *bag-of-words* (Salton G. and McGill M, 2013), que pode ser definida como uma representação numérica dos termos da coleção de documentos. Para a seleção das *features* mais relevantes eles utilizam TF-IDF e qui-quadrado e, não mencionam o uso de *n*-gramas. Nagwani et al. (2012) e Thung et al. (2012) também apresentaram trabalhos relacionados com a classificação de falhas de software utilizando uma etapa de pré-processamento textual similar a de Javed et al. (2012) com posterior uso de frequência de termos para a seleção de *features*.

Uma ressalva a essas abordagens de seleção de *features* a partir de frequência de termos é relatada por Chawla et al. (2014). Eles destacam que, em se tratando de falhas de software, podem haver palavras que não aparecem com frequência, mas que são extremamente relacionadas com uma classe. Esses autores citam que sinônimos não podem ser descartados e propõem uma abordagem baseada em informações semânticas para rotular falhas de software.

Após a aplicação de técnicas relacionadas à pré-processamento e seleção de *features*, algoritmos de aprendizagem de máquina são muito utilizados em problemas que exigem classificação de documentos textuais. Dos trabalhos já citados, Javed et al. (2012), e Chawla et al. (2014), fizeram uso dos algoritmos *Naive Bayes* e *Naive Bayes Multinominal*, respectivamente. Os autores consideram esses dois algoritmos como opções imediatas para aplicação em problemas de classificação textuais.

Thung et al. (2012) vão além e apresentam um comparativo entre seis algoritmos de aprendizagem de máquina a partir de *features* definidas por termos textuais: C4.5, *Logistict*, *Naive Bayes*, *Naive Bayes Multinominal*, RBF *Network* e SVM *Multiclass*. Eles avaliaram sua proposta sobre 500 defeitos recolhidos a partir do repositório JIRA sobre 3 softwares específicos.

Adicionalmente, Yelati et al. (2011) utilizam o algoritmo SVM, implementado na biblioteca LIBSVM (Chih-Chung Chang and Chih-Jen Lin, 2001), para classificar e-mails relacionados a chamados encaminhados a *help desk* para marcar e aprender sobre a intenção de seu conteúdo. Uma variante particular do SVM foi utilizada por Knebel et al. (2008) que utilizaram o algoritmo SMO (do inglês, *Sequential Minimal Optimization*). Eles sugerem que esse último algoritmo seja aplicado em problemas de dados esparsos, com significativa presença de elementos com valores iguais a zero na matriz de *features*.

3 Classificação de Chamados Técnicos de TI: Modelo Proposto

O modelo propõe a aplicação de técnicas de pré-processamento textual, semelhantes aos trabalhos citados na seção anterior deste artigo. Entretanto, em relação à extração e seleção de *features*, propõe uma abordagem híbrida, destacando-se a aplicação de padrões linguísticos, incluindo o uso de *n*-gramas, cálculo de relevância de termos e, ainda, uma etapa manual de anotações semânticas. Essa última etapa extrapola a proposta de Chawla et al. (2014), considerando não apenas sinônimos, como também outras associações semânticas.

Os algoritmos de classificação utilizados na classificação são: *Naive Bayes*, SMO e J48. Cada um desses algoritmos apresenta abordagens distintas de classificação o que amplia a possibilidade de observação do comportamento do modelo proposto. Além disso, outros motivos validam o uso desses algoritmos. Mitchell (1997) afirma que o *Naive Bayes* pode, para muitos conjuntos de dados, superar algoritmos

mais sofisticados, inclusive para problemas de classificação de documentos, similares ao deste trabalho. Platt et al. (1999), explicam que o SMO lida particularmente bem com conjunto de dados esparsos, característica detectada na matriz de *features* gerada pelo modelo proposto e utilizada como entrada dos algoritmos de aprendizagem de máquina. Complementarmente, optou-se por utilizar o algoritmo J48, que, por ser baseado em árvore de decisão, permite derivar regras de produção, de decisão ou de classificação com a possibilidade de visualização em árvore dos resultados, conforme relatado por em Patil et al. (2013).

A arquitetura global do modelo de classificação de chamados técnicos de TI é composta de 5 etapas, conforme apresentado na Figura 1. Essas etapas são descritas nas subseções seguintes.

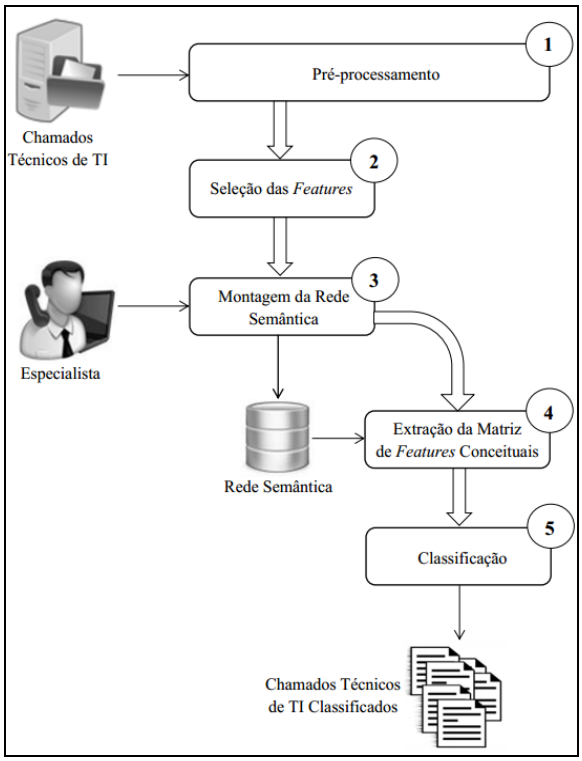

Figura 1. Arquitetura do Modelo de Classificação

3.1 Pré-processamento

A Etapa 1 de Pré-processamento executa algumas intervenções sobre o *corpus* de chamados técnicos, a saber:

- **Tokenizar**: o principal objetivo do uso de *tokens* é quebrar o texto em palavras (Yelati S. and Sangal R., 2011). No modelo proposto, foram extraídos *tokens* (ou termos) compostos por uma palavra (unigrama), duas palavras (bigrama) e três palavras (trigrama).
- **Reduzir tokens em radicais**: o uso de radicais poderá agrupar tokens similares e reduzir a lista final a ser utilizada na seleção de features. Os radicais foram obtidos pelo algoritmo de Orengo, implementado no pacote PTStemmer.
- **Etiquetar**: consiste em definir etiquetas para cada token. Utilizou-se o etiquetador TreeTagger (Schmid H., 2015), pois em testes empíricos realizados em nosso laboratório, ele obteve os melhores resultados.
- **Remover palavras sem relevância**: consiste em remover algumas *stopwords*, tais como: artigos, pronomes e conjunções. Removem-se também símbolos e sinais de pontuação. Destaca-se que as preposições presentes em termos trigramas são preservadas. Por exemplo, em termos como "usuário de rede" a preposição deve ser mantida.

Ao final desta etapa monta-se um *bag-of-words* com os termos (unigramas, bigramas e trigramas) encontrados.

3.2 Seleção das Features

Etapa 2 de Seleção das *Features* aplica dois filtros sobre a lista de termos do *bag-of-words*: um linguístico e outro estatístico. O objetivo é reduzir essa lista, selecionando os termos que tenham maior significado para as classes. Esses filtros também colaboram para reduzir o problema da dimensionalidade da matriz de *features* que será utilizada pelos algoritmos de aprendizagem de máquina na Etapa 5.

Primeiramente, deve-se aplicar um corte relacionado aos padrões linguísticos identificados nos chamados, ou seja, apenas os termos que forem compatíveis com os padrões da Tabela 1 são considerados. Por exemplo, para as palavras simples (unigramas), apenas os substantivos, verbos e adjetivos devem ser considerados. Para bigramas são considerados os substantivos seguidos de adjetivo e os verbos seguidos de substantivo. Para trigramas, apenas substantivos seguidos de preposição e de outro substantivo devem ser considerados. Esses padrões buscam preservar termos de conteúdo, que tenham maior valor de significado em cada chamado.

Tabela 1. Padrões Linguísticos.

N-Grama	**Padrão**
Unigrama	{<SUBS>|<VERB>|<ADJ>}
Bigrama	{<SUBS><ADJ>|<VERB><SUBS>}
Trigrama	{<SUBS><PREP><SUBS>}

Após o filtro dos padrões linguísticos, deve-se realizar uma análise estatística por meio do teste do qui-quadrado para calcular a relevância de cada termo para uma classe. Gera-se um ranqueamento decrescente pelo valor atribuído a cada termo de acordo com sua avaliação individual pelo teste. Ao final, todos os termos que tenham obtido valores menores ou iguais a zero serão eliminados. É obtida uma

lista de *features*, composta pelos demais termos que obtiveram valores maiores que zero.

3.3 Montagem da Rede Semântica

Uma intervenção importante do modelo proposto diz respeito ao uso de relações semânticas que transformam cada *feature* em um conceito mais abrangente. Desta forma, na verificação da presença de uma *feature* em um determinado chamado, considera-se também as relações semânticas da Tabela 2.

Tabela 2. Relações Semânticas

Relações Semânticas
Sinônimo
É um tipo de
Está associado a
Tradução

No processo de anotação semântica, especialistas devem realizar as ações listadas a seguir:

1. Analisar se existem relações semânticas entre as *features* selecionadas na Etapa 2.
2. Agrupar em um único conceito as *features* que possuem relações semânticas. Obtém-se uma lista de *features* conceituais.
3. Recuperar os termos distintos do *bag-of-words* extraída na Etapa 1 de Pré-processamento.
4. Analisar e anotar as possíveis relações semânticas entre as *features* conceituais e os termos do *bag-of-words*.

Ao final dessas ações, tem-se uma rede semântica entre os termos dos chamados e as *features* conceituais. A rede possibilita estender o significado de uma *feature*. A Tabela 3 mostra um recorde da rede semântica da *feature* "Impressora".

Tabela 3. Recorte da Rede Semântica da *Feature* Impressora

Termo	**Relação Semântica**	**Feature**
BROTHER	É um tipo de	Impressora
HP LASERJET	É um tipo de	Impressora
Scanner	Está associado a	Impressora
Printer	Tradução	Impressora

Essa extensão de significado será útil para reduzir a quantidade de valores iguais a zero na matriz que será gerada na Etapa 4 de Extração da Matriz de *Features* Conceituais. Ressalta-se que a Etapa 4 fica a espera da montagem manual da rede semântica. Por conta disso, o modelo de classificação proposto é dito semiautomático.

3.4 Extração da Matriz de Features Conceituais

Nesta Etapa 4 monta-se uma matriz onde cada linha representa um chamado, enquanto as colunas correspondem as *features* conceituais, definidas na Etapa 3. Uma última coluna é adicionada e preenchida com o valor da respectiva classe de cada chamado.

Analisa-se, linha a linha, a presença de uma determinada *feature* em cada chamado. Usa-se frequência binária para o preenchimento da matriz. O valor 1 indica que uma *feature* está presente no chamado, enquanto o valor 0 indica o contrário. Adicionalmente, para minimizar problemas de erros ortográficos, um algoritmo que calcula a similaridade entre termos é executado durante a verificação da presença de uma determinada *feature* em um chamado. Esse algoritmo é uma implementação da distância *Levenshtein*, que utiliza o conceito matemático de matrizes para realizar operações de inserção, exclusão ou substituição de um único caractere nas expressões a serem comparadas, com o objetivo de transformar uma expressão em outra, conforme explica Ristad et al. (2008).

3.5 Classificação

Para o processo de classificação (Etapa 5), um *corpus* anotado deve ser utilizado para o treinamento inicial do modelo, vez que os algoritmos de aprendizagem de máquina são supervisionados. O resultado final do processo, portanto, é o chamado técnico devidamente classificado.

4 Experimentos e Resultados

O *corpus* utilizado nos experimentos consiste de casos reais de chamados técnicos de TI cedido por órgão público do Estado do Piauí. Ao todo foram utilizados 2.064 chamados, todos rotulados manualmente por atendentes de *service desk* entre os meses de agosto a dezembro de 2015, divididos em 9 classes, conforme a Tabela 4 a seguir.

Tabela 4. Chamados Técnicos de TI por Classe

Classes	**Total**
A) BI	29
B) E-Mail	93
C) Equipamento de Informática	320
D) Impressão	253
E) Internet	164
F) Portal Web	215
G) Rede	419
H) Serviços Especiais	71
I) Software	500

Os 2.064 chamados da Tabela 4 foram submetidos ao modelo proposto. Durante a Etapa 2 de Seleção das *Features*, o primeiro corte baseado em

padrões linguísticos (Tabela 1) resultou numa lista de 5.267 termos. Na sequência, o teste de qui-quadrado gerou um ranqueamento, com o auxílio da ferramenta WEKA (Hall M. et. al, 2009), onde 381 termos tiveram valores maiores que zero. Na Etapa 3, esses termos foram agrupados em 202 *features* conceituais, tendo seus significados estendidos a partir de uma rede semântica previamente definida, considerando as relações da Tabela 2. Os testes da Etapa 5 de Classificação foram realizados utilizando como entrada uma matriz gerada na Etapa 4 com 2.064 linhas, cada uma representando um chamado técnico, e 202 colunas de *features* conceituais.

Para avaliar o desempenho da classificação foi utilizada a técnica estatística de validação cruzada com *fold* 10 (do inglês, *10-fold cross validation*). Os resultados foram registrados para três algoritmos de aprendizagem de máquina supervisionada: *Naive Bayes*, SMO e J48, disponíveis na ferramenta WEKA. Ressalta-se que os testes não tiveram como objetivo definir qual o melhor algoritmo. Cada um deles possui características e parâmetros distintos, que podem ser alterados na busca de outros resultados. O foco do experimento foi a validação do modelo proposto. A Tabela 5, a seguir, apresenta o resultado da classificação considerando 4 métricas: precisão (P), cobertura (R), *f-measure* (F) e acurácia (A). Essa última métrica representa o desempenho geral de classificação do modelo.

Tabela 5. Métricas de Avaliação

Algoritmos	**Métricas**			
	P (%)	R (%)	F (%)	A (%)
Naive Bayes	72.8	72.9	72.6	72.9
J48	70.9	70.7	70.1	70.7
SMO	75.8	75.7	75.4	75.8

Todos os algoritmos tiveram acurácia acima de 70%, tendo o SMO se destacado com 75.8%. Em busca de uma linha base, observou-se o *corpus* e detectou-se que 431 chamados, dos 2.064 utilizados nos testes, haviam sido reclassificados em algum momento durante o atendimento. Isso representa uma taxa de acerto manual de 79.1%. Embora esse percentual seja comparativamente maior do que os obtidos pelo modelo proposto, ainda assim o resultado é promissor. Analisando o *corpus*, existem chamados com textos mais complexos que podem ser rotulados em mais de uma classe, que em ambiente de teste podem ser marcados como erro, porém em ambiente de produção pode refletir uma classe possível para o chamado.

Há que se considerar também que a própria rede semântica pode ser evoluída. Nesse contexto, com a finalidade de validar o impacto das *features* conceituais obtidas a partir da rede semântica na Etapa 3, executou-se um teste de classificação utilizando como entrada uma matriz composta apenas por *features* linguísticas, gerada a partir dos 5.207 termos selecionados no primeiro corte da Etapa 2. A Tabela 6 mostra um comparativo da acurácia entre o uso de *features* linguísticas e *features* conceituais (com rede semântica).

Tabela 6. Comparativo entre Features Conceituais e Features Linguísticas

Algoritmos	**Acurácia (%)**	
	Features Conceituais *(Com Rede Semântica)*	*Features* Linguísticas *(Sem Rede Semântica)*
Naive Bayes	72.9	60.5
J48	70.7	65.7
SMO	75.8	66.7

Para todos os algoritmos, a rede semântica trouxe melhores resultados de acurácia. No caso do *Naive Bayes*, a diferença chegou a 12.5% a menor, quando a rede semântica foi desconsiderada.

É importante observar que os algoritmos foram colocados à prova diante de um problema multiclasse. Desta forma, os valores obtidos nas Tabela 5 e Tabela 6 representam uma média dos valores obtidos por cada classe. Nessa análise, a precisão e a cobertura são métricas que permitem analisar a performance do classificador e visualizar as classes que conflitaram entre si. A Tabela 7, a seguir, apresenta a matriz de confusão do algoritmo SMO, que obteve a melhor acurácia considerando como entrada a matriz gerada pelas *features* conceituais proposta pelo modelo.

Tabela 7. Matriz de Confusão com os Resultados de Classificação do Algoritmo SMO

Classes	A	B	C	D	E	F	G	H	I
A	**23**	1	0	0	0	1	0	1	3
B	0	**84**	0	0	0	1	7	0	1
C	0	2	**243**	7	6	2	21	0	39
D	0	0	8	**226**	3	4	5	0	7
E	1	1	2	2	**135**	9	10	0	4
F	2	3	2	5	9	**128**	9	3	54
G	0	7	32	13	12	13	**294**	0	48
H	1	2	1	1	1	3	6	**29**	27
I	0	6	24	14	3	20	26	7	**400**

Cada linha da Tabela 7 representa como o algoritmo classificou os chamados de uma determinada classe. Em um cenário perfeito, com 100% de acerto, só haveriam valores maiores que 0 na diagonal principal da matriz.

A classe, ***B***, por exemplo, possui 93 chamados. Esse número pode ser obtido somando os valores de sua respectiva linha. Para essa classe, 84 chamados foram corretamente classificados como ***B***, enquanto 1 ficou em ***F***, 7 em ***G*** e 1 em ***I***. A cobertura foi, nesse caso, de 90.3%. As colunas da matriz permitem observar os chamados de outras classes que foram marcados como ***B***. Apenas a classe ***D*** não perdeu chamados para ***B***.

As classes que obtiveram os piores desempenhos de classificação para o SMO foram ***F*** e ***H***, que respectivamente representam as classes "Portal Web" e "Serviços Especiais". Ambas tiveram muitos de seus chamados classificados como ***I***, que representa a classe "Software". Esses valores podem ser observados no cruzamento das linhas de ***F*** e ***H*** com a coluna ***I*** da Tabela 8. Analisando o conteúdo dessas 3 classes (***F***, ***H*** e ***I***), observa-se que ambas tratam de regras de negócios similares, que comungam de *features* similares em alguns casos.

5 Conclusão

Este artigo apresentou um modelo para classificação semiautomática de serviços de TI registrados em linguagem natural. Realizou-se um experimento com 2.064 chamados, distribuídos em 9 classes, previamente anotado por especialistas em *service desk*.

O modelo foi divido em etapas, onde técnicas de PLN foram aplicadas sobre os chamados até chegar-se na definição de *features* conceituais montadas a partir de uma rede semântica. Na classificação foram utilizados os algoritmos *Naive Bayes*, J48 e SMO que obtiveram acurácia de 72.9%, 70.7% e 75.8%, respectivamente.

O impacto positivo do uso da rede semântica foi comprovado, quando comparado com o desempenho dos algoritmos de classificação aplicados sobre o mesmo modelo, porém desconsiderando as *features* conceituais. Ressalta-se que em ambientes reais a classificação humana também não está livre de erros. Desta forma, com uma rede semântica apropriada, é possível concluir que o modelo proposto é promissor para o problema explorado.

Destaca-se como trabalho futuro a aplicação do modelo em ambientes de produção, onde a classificação possa ser avaliada de forma online.

Agradecimentos

Agradecemos à Universidade Federal do Piauí – UFPI, pelo suporte da pesquisa, e também à Secretaria Estadual da Fazenda do Piauí, por ceder o *corpus* utilizado neste trabalho.

Referências Bibliográficas

Ceci, Flavio, Cristiane Raquel Woszezenki, and Alexandre Leopoldo Gonçalves. (2014). "O uso de anotações semânticas e ontologias para a classificação de documentos." International Journal of Knowledge Engineering and Management (IJKEM) 3.5.

Chawla I., Singh S. K. (2014). "Automatic Bug Labeling using Semantic Information from LSI". In: Seventh International Conference on Contemporary Computing (IC3).

Chih-Chung Chang and Chih-Jen Lin. (2001). "LIBSVM: A Library for Support Vector Machines". Software available at: http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., and Witten H. (2009). "The weka data mining software: An update; sigkdd explorations". IEEE Transactions on Power Delivery, vol. 11, no. 1.

Hochstein, A., Zarnekow, R., Brenner, W. (2005). "ITIL as common practice reference model for IT service management: Formal assessment and implications for practice".

Javed M. Y., and Mohsin H. (2012). "An Automated Approach for Software Bug Classification". In: IEEE International Conference on Complex, Intelligent and Software Intensive Systems (CISIS).

Knebel, T., Hochreiter, S., & Obermayer, K. (2008). "An SMO algorithm for the potential support vector machine". Neural computation 20.1. pages: 271-287.

Mitchell, T. M. (1997). Machine learning. WCB. [S.l.]: McGraw-Hill Boston, MA.

Nagwani N., Verma S. (2012). "CLUBAS: An Algorithm and Java Based Tool for Software Bug Classification Using Bug Attributes Similarities". Journal of Software Engineering & Applications 5.6.

Patil, Tina R., and S. S. Sherekar. (2013). "Performance analysis of Naive Bayes and J48 classification algorithm for data classification." International Journal of Computer Science and Applications 6.2: 256-261.

Platt, J. et al. (1999). Fast training of support vector machines using sequential minimal optimization. Advances in kernel methods support vector learning, Cambridge, MA, v. 3.

Ristad, Eric Sven, and Peter N. Yianilos. (2008). "Learning string-edit distance." Pattern Analysis and Machine Intelligence, IEEE Transactions on 20.5: 522-532.

Salton G. and McGill M. (1993). "Introduction to Modern Information Retrieval". New York, NY, USA: McGraw-Hill.

Schmid H. (1994). "Probabilistic part-of-speech tagging using decision trees". Proceedings of International Conference on New Methods in Language Processing, Manchester, UK.

Thung F., Lo D., and Jiang L. (2012). "Automatic Defect Categorization". In: 19th Working Conference on Reverse Engineering (WCRE).

Yelati S. and Sangal R. (2011). "Novel approach for tagging of discourse segments in help-desk e-mails". In: Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology-Volume 03, pages 369–372.