Extração de insights sobre dúvidas em questões do Stack Overflow usando Mapas Auto-Organizáveis

Vagner Sargiani[1], **Leandro A. Silva**[1]

[1]Instituto Presbiteriano Mackenzie (MACK)
Rua da Consolação, 930 – 01302-907 – Consolação, São Paulo – SP – Brasil

vsargiani@gmail.com, leandroaugusto.silva@mackenzie.br

Abstract. *The discipline Programming Language in Computer Science courses has always been a great challenge for students and teachers. The interest in observe this discipline and analyze the interest of community can be used as a strategy of improving teaching and learning. This paper proposes to use the data from Stack Overflow forum, text mining techniques, and Self-Organizing Maps neural networks in order to discover insights of doubts from students and professionals who work with programming. This analysis can became a source of information to discovery, among other possibilities, to identify specific needs for programming disciplines, how demands of specialization courses, statistics about language use and doubts, and serve as a repository for searching information about programming languages.*

Resumo. *A disciplina Linguagem de Programação em cursos de computação sempre foi um grande desafio para alunos e professores. O interesse em monitorar e avaliar o processo de ensino desta disciplina, e identificar vulnerabilidades, é de interesse amplo como estratégia de aprimoramento de ensino e aprendizagem. Este trabalho propõe que se use os dados disponíveis no fórum de dúvidas Stack Overflow, técnicas de text mining, e redes neurais do tipo Mapas Auto-Organizáveis, a fim de obter insights de dúvidas frequentes de estudantes e profissionais de programação. A análise destas informações podem, entre outras possibilidades, identificar carências no aprendizado de linguagem de programação, necessidades de cursos de especialização, sensu sobre uso de linguagens e dúvidas de linguagens especificas, ou seja, servir como repositório de informações sobre linguagens de programação.*

1. Introdução

O processo ensino-aprendizagem para as disciplinas ligadas a linguagens de programação varia de acordo com as características de cada linguagem. Algumas dessas características são:

- Paradigma: orientadas a objetos (java, C++, Eiffel), eventos (visual basic e outras linguagens visuais), aspectos (AspectJ), estruturadas (C), funcionais (Pyton);
- Transparência: As linguagens podem possuir estruturação similar a linguagem humana (pascal, basic), ou possuir uma notação especifica para o uso ao qual se destina (LISP, Prolog);
- Abrangência: Existem linguagens de uso geral (C++, Java), linguagens de nichos específicos (R, LISP)

DOI: 10.5753/cbie.wcbie.2017.614

- Grau de Dificuldade: Baixo (Basic, Scratch), Média (PHP, HTML), Alto (C, Java)

Conforme citado em [Oliveira et al. 2016], a disciplina mais difícil em cursos ligados a tecnologia da informação é a de lógica de programação, que é apresentada tanto como uma disciplina ligada a Algoritmos como a Linguagens de Programação. A associação entre características distintas e a dificuldade apresentada pela lógica de programação podem fazer com que a quantidade de dúvidas que surgem no início do aprendizado de uma linguagem e a dificuldade que o aluno tem em encontrar as respostas resultem em frustrações, culminando inclusive na evasão de cursos.

Na literatura, os trabalhos correlatos a esta proposta que será apresenta a seguir citam diferentes abordagens para coleta e análise de dados; e uso dos mesmos para o aprimoramento no ensino de linguagem de programação. [Blikstein 2011], por exemplo, apresenta uma técnica para coletar dados dos alunos no momento da aprendizagem de programação, usando dados de acompanhamento ocular (*eye tracking*), tipos de erro em compilação etc; e com esse repositório faz-se análises para avaliar o desempenho dos alunos, com interesse sempre no aprimoramento da aprendizagem dos alunos. [Piech et al. 2012], por outro lado, se interessam na criação de agentes robôs que coletam dados de alunos no momento da solução de uma atividade, envolvendo tarefas especificas de linguagem de programação e gerando indicadores que mensuram como os alunos chegam a uma solução.

[Mayer et al. 1986], no entanto, apresentam uma discussão sobre as habilidades de pensamento computacional que são componentes cognitivos para os alunos aprenderem a programar, o que deve ser mais relevante a sua capacidade intelectual. E nesse artigo discute se o problema está na aprendizagem da linguagem de programação ou então no pensamento computacional. E no trabalho de [Barua et al. 2014], é discutido como é possível utilizar websites como StackOverflow.com para identificar padrões de tópicos, definidos pelos autores como grupos de palavras relacionadas que se aproximam de um conceito do mundo real. Desta forma, é possível realizar análises estatísticas na base textual de perguntas, avaliando os mais diversos aspectos, como assuntos com crescimento ou de popularidade, dúvidas mais frequentes ou mesmo tópicos com maior número de contribuições. O trabalho de [Zhang and Chow 2015] apresenta como é possível utilizar redes neurais (no caso particular dos Mapas Auto Organizáveis) para classificação e organização de conteúdo encontrado em formato texto para geração de conhecimento.

Neste trabalho, a proposta é verificar a possibilidade de usar dados de programação disponíveis em mídias sociais, onde as pessoas compartilham dificuldades no momento em que elas acontecem e as soluções são apresentadas coletivamente e com abordagens distintas. Existem websites, por exemplo, que permitem aproximar pessoas que possuem dúvidas com profissionais que podem auxiliar a esclarecê-las. Exemplos de websites com esta característica são: stackoverflow.com, informit.com, experts-exchange.com, codeproject.com e tek-tips.com. Outras mídias que podem ser utilizadas em discussões também podem servir de referência para esta nossa proposta como é o caso de listas de e-mails, como Yahoo Groups e Google Groups.

No aspecto de análise dos dados, diferente dos trabalhos supracitados com ênfase em explorações quantitativas, pretende-se aqui usar uma técnica comum de visualização de dados textuais, a qual é conhecida como nuvens de palavras (*word cloud*) [Silva et al. 2016]. Esta é uma técnica que permite identificar as palavras mais frequen-

tes até as menos frequentes de forma gráfica. No entanto, a técnica não possibilita que seja feita interpretações semântica sobre as visualizações. Como proposta de fazer essa composição, e aqui emerge a segunda contribuição deste trabalho, a ideia é criar nuvens de palavras com possibilidade de análise semântica. Para isso propõe-se aqui explorar o uso de Mapas Auto Organizáveis, um tipo de redes neurais artificiais que mantém as propriedades topológicas dos dados em um mapa bidimensional [Kohonen 2013]. A cada unidade do mapa, a ideia é gerar nuvens de palavras dos neurônios, permitindo assim que sejam feitas relações das nuvens de palavras, descobrindo relações de dúvidas.

O objetivo deste trabalho é avaliar o uso das perguntas e respostas disponíveis no Stack Overflow com a finalidade de descobrir insights sobre dúvidas relacionadas a Linguagem de Programação. Como contribuição adicional, propõe-se também a construção de nuvens de palavras semânticas através das publicações organizadas no mapa gerado pela rede SOM.

A estrutura do artigo está composta da seguinte maneira. Além da introdução, que apresenta o problema, motivação e objetivo da pesquisa, na seção 2 segue uma breve introdução teórica dos conceitos utilizados nesta pesquisa. Na seção 3 a metodologia utilizada para a validação dos objetivos do trabalho. Na seção 4 os resultados e discussões são apresentados e, por fim, na última seção a conclusão e trabalhos futuros.

2. Referencial teórico

2.1. Rede Social para Linguagem de Programação

O site Stack Overflow, como definido por [Adaji and Vassileva 2015], é uma plataforma para **Q&A** (Questions and Answers), onde usuários podem efetuar perguntas e responder questões voltadas a áreas específicas de TI. O formato é de fórum e o usuário, que pode estar cadastrado ou utilizar a modalidade convidado, pode realizar uma pergunta; e os demais usuários, profissionais especialistas, podem auxiliar na resposta à pergunta. O formato geral de uma questão para qualquer site mantido pela Stack Exchange, do qual faz parte o Stack Overflow, segue a estrutura definida por [Walk et al. 2016]:

- Pergunta (Questão): contém um título e um corpo (body);
- Respostas: Quando um usuário tem informações que podem auxiliar na resposta a uma pergunta, essas são registradas como Respostas. Conforme o site solicita, estes campos não devem conter outras dúvidas, mas somente informações a respeito da dúvida original;
- Comentários: As respostas podem ser comentadas pelos usuários, indicando sua efetividade em resolver o problema proposto na pergunta;
- Visualizações: Contagem sobre quantas vezes a pergunta foi visualizada;
- Votos: Neste contador, os usuários votam nas perguntas que acharam mais interessantes ou nas respostas que resolvem a dúvida. O site utiliza este sistema de votos para qualificar seus usuários. O usuário recebe badges (marcadores virtuais) conforme recebe ou atribui votos, permitindo assim identificar os usuários de acordo com a efetividade da contribuição dentro do site.

Neste modelo é possível perceber que a postagem inicial da questão pode dar origem a respostas ou comentários. Uma resposta é uma tentativa de resolver o problema proposto na questão, que pode ou não ser a resposta definitiva. Já os comentários, esses

podem ter as seguintes finalidades: apoio, crítica, complemento (da questão ou da resposta), agradecimento, resposta (apesar de não ser o local correto para isso), entre outros.

Neste trabalho, o foco principal será apenas na pergunta, devido ao fato dessa ser a origem de uma questão e fundamental para a identificação de insights sobre dúvidas comuns. O formato lógico de uma pergunta normalmente tem presente os seguintes tópicos:

- Título: Texto breve, geralmente com uma prévia do problema. Nas telas de navegação e de busca, é o primeiro texto a aparecer;
- Introdução: É uma breve contextualização sobre o que gerou a dúvida. Pode ser algo particular sobre um desenvolvimento, um estudo etc;
- Definição de Contexto: Opcional, é um detalhamento maior, normalmente ocorre quando o objeto da dúvida só ocorre em determinadas situações;
- Dúvida: É a pergunta em si. Como houve uma pré-contextualização, normalmente a dúvida é objetiva, por exemplo: "Como divido um objeto por um inteiro? " ou "Como formato uma data? ";
- Exemplo: Também opcional, normalmente contém um trecho de código onde o objeto da dúvida ocorre. O local deste item é variável em qualquer ponto da dúvida, pode existir mais de um, e é facilmente identificado pelas tags HTML `<code></code>` (utilizada pelo site para permitir uma formatação diferenciada do resto do conteúdo).

2.2. Mineração de Textos

Como as postagens do fórum estão em formato de texto, para a realização de análises é necessário estruturar esse tipo de dados por meio de um modelo de representação, que transforma os termos de cada publicação em um valor de relevância (peso). A análise de dados não estruturados é efetuada em três fases distintas.

A primeira fase do processo consiste na construção do vocabulário que se dá pelo processo de Mineração de Textos (text mining). Ou seja, o texto com todas as perguntas selecionadas representa o *Corpus* inicial. Para geração da representação (*Corpus* representado) é necessário seguir os seguintes passos [Goker and Davies 2009, Silva et al. 2016]:

i *Tokenization*: A partir do caractere espaço, os comandos das instruções são separados em tokens. Os caracteres especiais como vírgulas (","), pontos de exclamação ("!") e de interrogação ("?") e números são removidos. Padronização de capitalização para minúsculas (ou maiúsculas) são realizadas nessa fase.

ii *Stopwords*: Palavras como preposições, pronomes, artigos, advérbios que são comuns em diferentes contextos, e por isto não apresentam significado relevante, são removidos do processo;

iii *Stemming*: As palavras resultantes das etapas anteriores passam por um processo de normalização, sendo reduzidas ao radical. Este processo é importante, pois palavras com o mesmo radical podem ser consideradas como semelhantes.

Com esses passos os textos são limpos e integrados, resultando etapa chamada "Dados Limpos". Na fase "Dados Preparados" está o *Corpus* gerado em formato de matriz, onde nas linhas se tem as perguntas e nas colunas as palavras mais relevantes. O preenchimento dessa matriz representa a etapa de representação, que transforma os termos

de cada publicação em um valor de relevância (peso). A representação dos documentos pode ser feita usando o modelo chamado espaço vetorial [Goker and Davies 2009]. O modelo espaço vetorial busca representar documentos em forma de vetor, composto por termos (palavras) representados por um peso ponderado. Uma das formas de se fazer a ponderação é combinando a frequência que um termo aparece em um documento (**tf** do inglês, *term frequency*) com a frequência invertida do documento (**idf** do inglês, *inverse document frequency*). A combinação de **tf** x **idf** define a importância (peso) de um termo dentro do documento [Goker and Davies 2009, Silva et al. 2016]. Assim, depois de calculado o peso de cada termo do documento, tem-se como resultado uma base de dados numérica formada por vetores de documentos e, assim, consequentemente, realiza-se as análises desejadas. Neste trabalho a análise será feita com uso de Mapas Auto-Organizáveis.

2.3. Redes Neurais do tipo Mapas Auto Organizáveis

Segundo [Kohonen 2013], uma rede neural SOM (do inglês *Self-Organized Maps*) é um método de análise de dados não supervisionado. É baseado na forma de análise de sinais chamada "Vector Quantization", introduzido por [Lloyd 1982] para dados escalares, e por [Forgy 1965] para forma vetorial. A arquitetura SOM é organizada em duas camadas, sendo a primeira a de recebimento dos dados e a segunda com nós organizados em um grid em formato bidimensional (usualmente).

Segundo as definições de [Kohonen 2013], dados similares se aproximam no grid da rede, enquanto os menos similares se afastam no mesmo grid. Desta forma, SOM é uma rede que auto organiza os dados de acordo como as entradas são apresentadas e os pesos com os quais os modelos (neurônios) são inicializados. Esta inicialização comumente é feita de forma aleatória, mas levando em consideração a posição espacial assumida pelo mesmo no grid.

A implementação de modelos para o SOM pode utilizar dois algoritmos: processo de aproximação gradual recursivo (Os dados de entrada são aplicados ao algoritmo, um de cada vez, em uma sequência aleatória ou periódica, para quantos passos forem necessários para se chegar em uma configuração razoavelmente estável) ou processo tipo Batch (todos os dados são aplicados ao algoritmo em lote, e todos os modelos são atualizados em uma única operação simultânea. Este processo precisa ser executado algumas dezenas de vezes, até que os modelos se estabilizem).

Ainda segundo [Kohonen 2013], a seguinte equação representa o processo de definição do neurônio (modelo) vencedor de índice c, ou seja, o neurônio com peso mais próximos de $x(t)$.

$$c = arg\ min(||(x(t) - m_i)||), \qquad i \in \{1, 2, 3, ..., \mathbb{N}\} \tag{1}$$

E o cálculo de pesos do neurônio vencedor e seus vizinhos seguem a seguinte equação:

$$m_{j+1} = \begin{cases} m_j + h_{jc}(t)\,(x(t) - m_j(t)) & \forall\ j \in \mathbb{N}_c \\ m_j(t) & Caso\ Contrario \end{cases} \tag{2}$$

Nas formulas, m representa o neurônio de índice i ou j do vetor que contém o

grid, e o valor h_{jc} é chamado **Função de Vizinhança**, e define o quanto cada vizinho de M_c será atualizado. O resultado final do treinamento do SOM significa que dados semelhantes sejam mapeados em um mesmo neurônio ou, então, ficam em neurônios vizinhos do mapa, mantendo assim a topologia dos dados.

3. Metodologia

3.1. Obtenção dos Dados

O site Stack Overflow oferece uma plataforma de consulta a todo o conteúdo chamada de Stack Exchange. Após a consulta, os resultados podem ser exportados no formato CSV[1]. Caso se deseje a base completa, também é possível baixar um arquivo dump, disponibilizado para Download. Esta é uma opção interessante para avaliações históricas, o que não é o foco deste trabalho.

3.2. Construção do vocabulário

Os dados foram coletados do Stack Overflow no mês de setembro de 2016, resultando em uma massa de dados total de 479.300 registros. Esses dados serão manipulados com o uso da ferramenta R Studio.

Para realizar a análise dos dados utilizando text mining foram utilizados os seguintes pacotes: **tm** (análise e mineração de dados em conteúdo de textos), **wordcloud** (visualização de nuvem de palavras), **SnowballC** (realiza o *stem* dos termos).

Após a construção do *Corpus* representado, gera-se uma matriz de termos. Existem duas configurações possíveis para esta matriz [Feinerer and Hornik 2015]: **DocumentTermMatrix** (Produz uma matriz onde cada linha é um documento, e cada coluna é um termo) e **TermDocumentMatrix** (Produz uma matriz onde cada linha é um termo, e cada coluna é um documento).

Outra atividade que faz parte da montagem da matriz é a remoção de registros esparsos (número baixo de frequência). A matriz gerada terá um grande número de documentos com termos cuja frequência é baixa a ponto de impactarem negativamente nas análises. Para este estudo, a frequência mínima para que um termo seja incorporado na análise será de 0,98.

3.3. Treinamento do SOM

A estrutura da rede SOM terá apenas uma saída bidimensional. Para implementação, será utilizada a biblioteca R chamada `kohonen`, que possui um modelo fiel ao definido originalmente por [Kohonen 2013].

Para cálculo do tamanho do grid que será utilizado, utilizou-se a proposta de [Vesanto and Alhoniemi 2000]. Para este artigo o o grid terá dimensão 9×9.

4. Experimentos e Avaliação dos Resultados

O conjunto de dados utilizados foi o descrito na seção 3.1, e será filtrado para permitir as análises. O objetivo da criação de um subconjunto é a redução de um problema que ocorre

[1] do inglês Comma-separated values, arquivo de valores separados por vírgulas.

em grandes volumes de dados textuais, que é ocorrência de termos esparsos. Trabalhando com a fragmentação por termos, é possível efetuar a análise de termos comuns ao tema.

Como prova de conceito para as análises aqui propostas, foi feita a seleção de 178 questões. Os filtros utilizados no Stack Overflow para a geração do subconjunto de dados para a análise seguiram os requisitos abaixo:

- Palavras de busca para análise: "java", "maven", "ant", "jenkins". A escolha destes termos (palavras) se deve ao fato de: "java" ser uma linguagem muito popular na atualidade, segundo lugar no ranking IEEE Spectrum (2017); "maven" e "ant" são ferramentas que permitem efetuar a compilação e geração de dependências dos projetos de software; e "jenkins" é uma ferramenta muito utilizada para automação do processo de compilação e publicação. O objetivo foi concentrar em dúvidas de compilação, gerenciamento de módulos e publicação da linguagem Java;
- Período considerado: de 1 a 20 de setembro. Foram testados períodos menores dentro do mês de setembro de 2016, e este período permitiu obter uma amostra com tamanho necessário para o experimento;
- Quantidade de visualizações da dúvida pelos membros do Stack Overflow como sendo acima de 50. O objetivo foi recuperar registros que foram acessados, indicando se tratar de perguntas relevantes (comuns a outras pessoas);
- Resposta da pergunta deve ter pelo menos uma resposta: Ao menos um especialista respondeu à pergunta.

A escolha dos termos "maven" e "ant" possuem relevância e foram escolhidos devido aos seguintes fatores: são gerenciadores de dependências para linguagens de programação, seus scripts são arquivos com formato específicos e que envolvem conhecimento, o processo de compilação pelas interfaces de desenvolvimento padrão utilizam em sua maioria um destes dois programas, e normalmente sua utilização não faz parte das disciplinas ligadas ao ensino de programação, fazendo parte de disciplinas ligadas a automação de ciclo de vida de desenvolvimento de softwares. "jenkins" é um gerenciador de ciclo de vida de aplicação, e sua utilização foi com o objetivo de criar clusteres com o uso dos termos "maven" e "ant" com uma linguagem ("java") separados do uso com gerenciadores de ciclo de vida.

Com os dados carregados, pode-se efetuar a seleção da amostra desejada e iniciar a limpeza inicial. Esta limpeza removeu código identificado pelas tags `<code></code>`, e em seguida todas tags de marcação HTML dentro do campo body. A remoção destas informações permite que a análise se concentre apenas no texto referente a pergunta.

Nesse sentido foi criado um *Corpus*, utilizando os recursos de text mining. Nesse *Corpus* estão todos os termos e documentos associados, porém ainda existe a necessidade de efetuar a limpeza do mesmo, conforme discutido na subseção 2.2. Na sequência, cria-se a matriz de termos no formato **DocumentToText** (como discutido na subseção 2.2).

O principal problema encontrado neste trabalho foi o alto número de termos esparsos, frutos de diversos fatores, tais como: escrita livre dos usuários, que utiliza abreviações e comete erros ortográficos, e formação de palavras inexistentes durante o tratamento do texto (como remoção de hífens e pontuação). Esses termos ocorrem em um número muito baixo no total da amostra e podem interferir negativamente no tempo de processamento. A solução adotada foi aplicando um filtro prévio para reduzir a amostra e ajusta-la apenas para refletir alguns termos chave. Adicionalmente, no ciclo de remoção

de *stopwords* foi incluído o termo "java" (presente no filtro inicial), pois todas questões apresentavam este termo, o que comprometeria o trabalho de análise.

Para o treinamento da rede SOM os parâmetros utilizados foram dimensão do mapa de 9 por 9 com formato retangular, número de épocas em 80 e a taxa de aprendizado entre 0,09 e 0,01. O tamanho do grid foi definido dentro do que foi discutido na subseção 3.3. A raiz do número de questões utilizado ficou em 13,34, então o número a ser escolhido deveria ser maior que 2 e menor que 13. O número 9 foi definido para permitir uma melhor visualização gráfica e pelo tempo de treinamento (quanto maior o grid, mais tempo é consumido no treinamento da rede). Já o formato retangular permite a localização mais simples de um determinado neurônio dentro do grid. A rede se estabilizou próximo de 45 iterações, justificando a escolha de 80 épocas para treinamento.

O próximo passo foi definir como analisar os dados do grid SOM. A distribuição dos neurônios dentro do grid, com topologia retangular, se organizam da esquerda para direita, e de baixo para cima. Para efetuar essa análise, pode-se montar um conjunto de nuvens que reflita a composição de termos de cada neurônio, selecionando o neurônio e seus vizinhos adjacentes. O resultado está ilustrado na Figura 1.

Pode-se perceber da Figura 1 que existem termos que se repetem entre as nuvens devido a manutenção topológica do mapa, permitindo analisar semanticamente as nuvens combinadas. Assim, é possível navegar pelos termos entre neurônios, para explorar as dúvidas associadas.

Como um exemplo de análise a se fazer para demonstrar o uso da proposta com o mapa semântico gerado, considere o termo "pomxml" do neurônio 31. Este termo ocorre, pois, o arquivo Pom.xml é um arquivo central de configuração de um projeto Maven (neurônio 40). O ponto do "Pom.xml" foi eliminado após os tratamentos de text mining. O neurônio onde o termo aparece com maior relevância é o 31, mas é possível perceber a presença nos neurônios adjacentes 32, 39, 40, 48 e 49, e é possível perceber associações desse termo com outros, por exemplo: "depend", "spring", "error" e "except". Portanto, é possível inferir que as dúvidas nessa área se concentram em como configurar dependências dentro do arquivo Pom.xml. As respostas originais sem a representação e mapeadas nestes neurônios foram recuperadas e a interpretação acima apresentada foi validada. Isso mostra um tipo de dificuldade que pode ser resolvida com a explicação em mais detalhes do processo de configuração do Maven ou então, a proposta de cursos que ajudam a realizar esse processo.

O processo de descobertas é cíclico, onde análises similares, focando em outros termos, devem ser realizadas. Desta forma é possível identificar novos agrupamentos de termos pela análise visual dos neurônios e, assim, obter novos insights.

E ainda, a análise pode ser feita considerando individualmente cada neurônio, considerando que há maior similaridade nesse caso, ou seja, perguntas mais similares. Como exemplo dessa outra abordagem de análise, o neurônio 49 possui diversas palavras em destaque, como "access", "edit", "file", "project", "directori", e outros termos que remetem a estruturas de arquivos. É possível inferir assim, que as dúvidas concentradas neste neurônio têm com o objetivo acesso de arquivos em um projeto. Ao acessar o texto integral destas perguntas, foi possível confirmar que as dúvidas eram referentes a como configurar o acesso a um arquivo para ser utilizado.

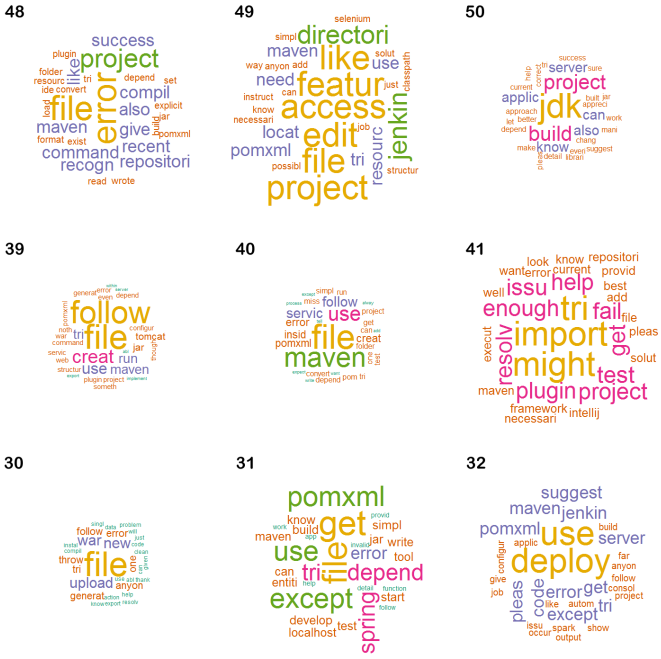

Figura 1. Nuvem do neurônio 40 e dos neurônios adjacentes

O neurônio 40, por outro lado, apresentou o maior volume de perguntas, ou seja, é o grupo de maior densidade. Analisando os termos em seu interior pode-se identificar a presença de palavras como “servic”, “convert”, “create”, “inside”. Em conjunto com os termos “maven” e “file”, a dedução possível é que são perguntas sobre gerenciamento de arquivos dentro de projeto. Assim como feito nas demais análise, as perguntas foram recuperadas para confirmar a dedução realizada.

5. Conclusão

A linguagem de programação em cursos de computação é sempre um grande desafio para as Instituições de Ensino, pois é disciplina de muita dificuldade dos alunos, resultando em altas taxas de reprovação e, consequentemente, é considerada como responsável pela evasão nesse tipo de curso. Contudo, como apresentado nos trabalhos correlatos, existe uma serie de tentativas para tentar mensurar as dificuldades dos alunos. Neste trabalho introduzimos e avaliamos a viabilidade de usar dados de perguntas e respostas do Stack Overflow como fonte de dados para descoberta de dúvidas relacionadas a conceitos associados a linguagem de programação Java.

As perguntas e respostas consistem em dados não estruturados que foram preparados usando conceitos de text mining e análises com o uso de Mapas Auto Organizáveis (SOM). Assim, como contribuição deste trabalho, apresentamos o uso do Stack Overflow como fonte de dados sobre programação e o uso de nuvens de palavras a partir dos neurônios do mapa, permitindo que se faça análise semântica dos resultados, o que possibilita a descoberta de insights sobre os tipos de dúvidas.

O uso de text mining é interessante para poder identificar padrões dentro de um conjunto de textos. Por exemplo, criando-se dicionários apropriados é possível em trabalhos futuros agrupar a ocorrência dos termos ou mesmo efetuar uma análise de senti-

mentos (“positivo” e “negativo”). Também como trabalho futuro, deve-se pensar em uma maneira usual de fazer as análises através do mapa SOM para tornar a aplicação mais prática.

Referências

Adaji, I. and Vassileva, J. (2015). Predicting Churn of Expert Respondents in Social Networks Using Data Mining Techniques: A Case Study of Stack Overflow. *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 182–189.

Barua, A., Thomas, S. W., and Hassan, A. E. (2014). What are developers talking about? An analysis of topics and trends in Stack Overflow. *Empirical Software Engineering*, 19(3):619–654.

Blikstein, P. (2011). Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st International Conference on Learning Analytics and Knowledge - LAK '11*, LAK '11, page 110, New York, New York, USA. ACM Press.

Feinerer, I. and Hornik, K. (2015). Package 'tm': Text Mining Package. *Hadoop help*, pages 1–58.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, 21:768–769.

Goker, A. and Davies, J. (2009). *Information Retrieval: Searching in the 21st Century*. John Wiley & Sons.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks*, 37:52–65.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Mayer, R. E., Dyck, J. L., and Vilberg, W. (1986). Learning to program and learning to think: what's the connection? *Communications of the ACM*, 29(7):605–610.

Oliveira, M. V. D., Rodrigues, L. C., and Paula, A. (2016). Material didático lúdico : uso da ferramenta Scratch para auxílio no aprendizado de lógica da programação. In *XXII Workshop de Informática na Escola*, pages 359–368.

Piech, C., Sahami, M., Koller, D., Cooper, S., and Blikstein, P. (2012). Modeling how students learn to program. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education - SIGCSE '12*, page 153, New York, New York, USA. ACM Press.

Silva, L. A., Peres, S. M., and Boscarioli, C. (2016). *Introdução À Mineração de Dados - Com Aplicação Em R*. CAMPUS - GRUPO ELSEVIER, 1ª edição edition.

Vesanto, J. and Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3):586–600.

Walk, S., Helic, D., Geigl, F., and Strohmaier, M. (2016). Activity dynamics in collaboration networks. 10(2).

Zhang, H. and Chow, T. W. S. (2015). Organizing Books and Authors by Multilayer SOM. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–14.