XI JORNADA DE INICIAÇÃO CIENTÍFICA E TECNOLÓGICA
EVENTO ON-LINE
23 A 26 DE NOVEMBRO
UNIVERSIDADE FEDERAL DA FRONTEIRA SUL

UM ESTUDO SOBRE REUTILIZAÇÃO DE TREINAMENTO EM MODELOS DE PREVISÃO DE VULNERABILIDADE

MATHEUS VINÍCIUS TODESCATO[1], GUILHERME DAL BIANCO[2]

1 INTRODUÇÃO

Com o avanço da tecnologia, os códigos-fonte de *softwares* têm ficado cada vez mais complexos. Analisar em busca de *bugs* um conjunto de grande porte manualmente representa uma tarefa inviável e tediosa. Uma alternativa é utilizar Modelos de Previsão de Vulnerabilidade (MPVs). Os MPVs tem como objetivo detectar arquivos de código-fonte que contenham alguma vulnerabilidade. Os MPVs são baseados em aprendizagem de máquina e o treinamento é feito a partir de erros conhecidos. Sabendo qual arquivo é vulnerável, o desenvolvedor pode focar na resolução do possível erro (Yu *et al.*, 2019).

A tarefa de encontrar todas e somente as vulnerabilidades pertence à classe de problemas de recuperação de informação e é denominada de revocação total (ou *high recall*). O objetivo da revocação total é otimizar o custo na obtenção de altos níveis de revocação (idealmente em torno de 100\%) com um humano assessor no ciclo (Yu *et al.*, 2018). Atingir uma alta revocação é importante para reduzir vulnerabilidades que podem causar impactos significativos quando liberadas aos usuários.

Os MPVs têm se mostrado cada vez mais precisos na identificação de vulnerabilidades, no entanto, ainda é uma tarefa complexa e possui diversas limitações. Como, por exemplo, se o conjunto de códigos não possuir uma sinalização da ocorrência de um possível *bug*, será necessário o auxílio de um humano para validar uma grande quantidade de arquivos de código. Uma alternativa de mitigar o esforço do revisor humano é aplicar abordagens de aprendizagem ativa (Settles, 2009). A aprendizagem ativa tem como objetivo reduzir o esforço do usuário na rotulação, identificando os documentos mais informativos para o processo de aprendizagem (Settles, 2009). Por exemplo, o método KNEE (Cormack *et al*., 2016), explora aprendizagem ativa para determinar quais documentos devem ser rotulados pelo usuário no contexto de textos livres (sem estrutura). Além de diminuir o esforço de rotulação, o método almeja estimar o número de documentos relevantes da coleção, podendo interromper, com segurança, o método antes de analisar todos os

1 Graduado em Ciência da Computação pela Universidade Federal da Fronteira Sul, *campus* Chapecó, **atualmente Mestrando na UFRGS**, contato: mvtodescato@hotmail.com.
2 Doutor,UFFS, Orientador.

documentos.

Um dos desafios que persiste ao adotar a aprendizagem ativa é a geração do conjunto de treinamento inicial (semente). Este conjunto, utilizado pelo algoritmo de aprendizado no início do processo, é primordial para a identificação de padrões que configuram um documento relevante (Seattles, 2009). Tal etapa possui grande impacto no esforço despendido pelo usuário, posto que pode acelerar ou atrasar o aprendizado do algoritmo. Ou seja, se nenhum trecho com *bug* é adicionado ao treinamento, o algoritmo de ranqueamento dificilmente conseguirá produzir um bom ordenamento. Tal problema é conhecido como *cold-start* (Kawase *et al*., 2013), estando presente não apenas no campo da aprendizagem ativa mas também em algoritmos de aprendizado em geral.

2 OBJETIVOS

Este trabalho tem como objetivo avaliar a reutilização de treino para MPVs com a intenção de aprimorar a qualidade do processo no contexto da aprendizagem ativa. A hipótese é que na ausência de trechos marcados como *bugs*, pode-se reutilizar códigos rotulados (treinamento) de outros projetos para acelerar o processo de aprendizado do método ativo, reduzindo assim o esforço de rotulação. Nesse contexto, este artigo busca responder às seguintes questões de pesquisa:

- A reutilização do treinamento de outras bases de dados pode auxiliar no problema da partida fria do método?
- O número de arquivos com *bugs* fornecidos para o método ativo impacta no ganho de informação do modelo ativo?

3 METODOLOGIA

Inicialmente identificou-se os principais trabalhos científicos que exploram técnicas de predição de vulnerabilidades e recuperação da informação. Dessa forma, os principais estudos foram selecionados para se entender o seu comportamento: HARMLESS (Yu *et al.*, 2019), VULPREDICTOR (Zhang *et al*., 2015), KNEE (Cormack *et al*., 2015). Diferentemente dos métodos citados acima, este trabalho tem como objetivo analisar se é possível auxiliar o processo de aprendizagem ativa a partir da reutilização de arquivos de outros projetos, já rotulados. Nesse contexto, foi explorado o uso do método ativo KNEE, voltado para textos livres, no contexto da predição de arquivos com *bugs*.

As bases de dados utilizadas neste trabalho são públicas (Tóth et al., 2013) e estão disponíveis para acesso[3] . O conjunto de dados foi construído com objetivo de contribuir para aplicações de predição de bugs. As bases de dados envolvem códigos utilizando a linguagem de programação Java na plataforma GitHub. Foram selecionadas 105 versões de lançamentos de projetos ao longo de seis meses para a construção do conjunto de dados. Utilizando técnicas de extração de características foram gerados mais de 183 mil arquivos de código.

No intuito de validar o desempenho do experimento proposto, são usadas duas métricas de avaliação: revocação e o *Custo*. A métrica de avaliação, revocação ou *recall* calcula a relação dos documentos relevantes encontrados e o total de relevantes . Já o *Custo* é medido pela porcentagem de arquivos revisados. Para auxiliar na comparação direta das execuções, será reportada a curva (ou a área) envolvendo *recall* vs. *Custo*. Quanto maior a área abaixo da curva, maior será o *recall* e menor será o *Custo*.

Os experimentos foram realizados utilizando a implementação do método KNEE disponibilizado como código aberto. O algoritmo de aprendizagem de máquina utilizado foi o SVM. Originalmente, o método KNEE conta com uma entrada textual do usuário (*query*), porém, no contexto do MPVs, na qual não está disponível a *string* de consulta, serão executados as seguintes configurações: *(i)* "original" onde será fornecido um arquivo com *bugs* selecionado randomicamente na mesma base de dados e rotulado pelo usuário para início do processo de aprendizagem; *(ii)* "1 arq. treino" na qual será utilizado um arquivo de treino de outra base de dados para início do processo; *(iii)* "5 arq. treino", na qual, serão extraídos cinco arquivos de treino de outra base de dados; *(iv)* "10 arq. treino", serão utilizado 10 arquivos de treino de outra base de dados; e por fim, *(v)* "100% arq. treino" será utilizado, como ponto de partida, todo conjunto já rotulado de outra base de dados.

4 EXPERIMENTOS E RESULTADOS

Por meio dos experimentos pode-se perceber que ao adicionar ao treinamento inicial "1 arq. treino" com *bug* de outra base de dados não foi produzido ganho de aprendizado ou redução no custo de rotulação quando comparado às demais áreas de curvatura. Já, ao incrementar para "5 arq. treino" ou "10 arq. treino", foi possível obter os melhores resultados em comparação com os demais nas bases de dados analisadas. Por exemplo, a base de dados "anttlr4" (composta por arquivos de código oriundos de um projeto de processamento de linguagem) obteve um ganho de 11% na área da curva quando adicionado "5 arq. treino" se

3Disponível em:

comparado ao método KNEE original.

Ao adicionar toda a base de dados (“100% arq. treino”), foi obtido um baixo valor de área se comparado aos demais. Acredita-se que esse comportamento seja devido a dominância de exemplos da base de dados reutilizada, isso dificulta a adaptação do modelo ativo de predição para o padrão de *bugs* presente na base de dados alvo.

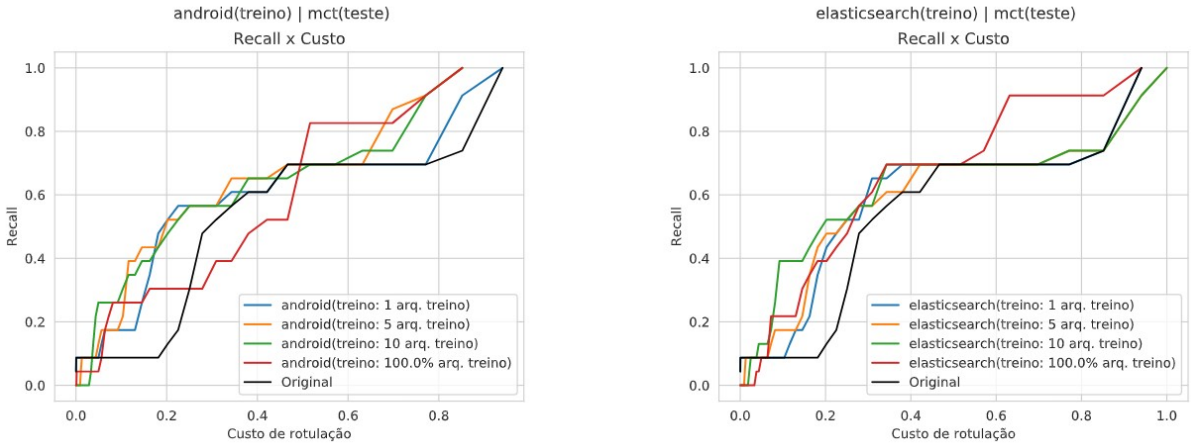

Figura 1. Curva *recall vs custo* aplicando a base “*mct*”.

Os gráficos presentes na Figura 1 detalham as curvas envolvendo o *recall* vs *custo* da base de dados “*mct*” (proveniente de códigos de uma plataforma de monitoramento). Conforme já dito, quanto mais elevada for a curva, maior será o *recall* e menor será o custo de rotulação. A linha preta representa a execução *Original* do método KNEE. Ao utilizar como treino a base de dados rotulada completa (*100% arq. treino*) podemos observar uma curva com baixa aceleração, resultando em um atraso no ganho do *recall*. Com a mesma configuração, a base de dados *mct* apresentou um comportamento divergente, já que sua curva mostrou uma melhora acentuada no *recall* nos pontos finais (em torno de 60% do *recall*). As curvas em “verde” e “amarelo” (*treino: 5 arq. treino e treino: 10 arq. treino*) mostram uma aceleração promissora no início do processo ativo em ambos os gráficos. Ao utilizar um conjunto de treino de “5 arq. treino” ou “10 arq. treino”, podemos observar as melhores curvas quando comparado aos demais.

5 CONCLUSÃO

Este artigo teve como objetivo analisar experimentalmente a reutilização de treinamento no contexto da previsão de vulnerabilidades, ou seja, na identificação de *bugs* em códigos. Para isso, foram desenvolvidos experimentos em arquivos de código com uma baixa prevalência de vulnerabilidades. A experimentação permitiu identificar que a

reutilização de uma base de dados completa (contendo todos os arquivos rotulados) pode prejudicar o aprendizado do modelo de previsão de vulnerabilidades. Já, ao se adicionar 5 ou 10 arquivos contendo *bugs*, foram observados ganhos promissores em relação à execução original. Em trabalhos futuros o objetivo é comparar o método ativo KNEE com outros utilizados para revisões da literatura, aplicando no contexto da previsão de vulnerabilidade.

REFERÊNCIAS BIBLIOGRÁFICAS

YU, Zhe, Christopher Theisen, Laurie Williams, and Tim Menzies. "Improving vulnerability inspection efficiency using active learning." **IEEE TSE** (2019).

SETTLES, Burr. "Active learning literature survey." (2009).

CORMARK, Gordon V., and Maura R. Grossman. "Engineering quality and reliability in technology-assisted review." In Proceedings of the **ACM SIGIR**, pp. 75-84. 2016.

KAWASE, Ricardo, Marco Fisichella, Bernardo Pereira Nunes, Kyung-Hun Ha, and Markus Bick. "Automatic classification of documents in cold-start scenarios." In Proceedings of the **3rd International Conference on Web Intelligence**, Mining and Semantics, pp. 1-10. 2013.

TÓTH, Zoltán, Péter Gyimesi, and Rudolf Ferenc. "A public bug database of github projects and its application in bug prediction." In: **ICCSA**, pp. 625-638. Springer, Cham, 2016.

ZHANG, Yun, David Lo, Xin Xia, Bowen Xu, Jianling Sun, and Shanping Li. "Combining software metrics and text features for vulnerable file prediction." In: **2015 ICECCS**, pp. 40-49. IEEE, 2015.

Palavras-chave: predição de vulnerabilidade; *High-recall Information Retrieval*; aprendizado ativo.

Nº de Registro no sistema Prisma: PES-2020-0256

Financiamento: UFFS