Avaliação de Aprendizagem de Máquina para Classificação de Requisitos em Termos de Qualidade

Carla A. M. Vieira[1]

[1]Instituto de Ciências Exatas e Informática
Pontifícia Universidade Católica do Minas Gerais (PUC Minas)
Caixa Postal 30.535-901 – Belo Horizonte, MG – Brazil

camvieira@sga.pucminas.br

Abstract. *Requirements Engineering is a fundamental step for the software development, as well as the developed requirements quality directly influence the development success. Generally, the quality assessment of the requirements is done manually by the engineers themselves, making the process slower, subjective and susceptible to errors. As an alternative to this evaluation, Machine Learning algorithms can be used to evaluate the requirements quality written in natural language in an automated way. This article proposes a comparative analysis between the combination of two vectorization techniques (Bag of Words and Term Frequency-Inverse Document Frequency) and four supervised learning algorithms (k-Nearest Neighbor, Support Vector Machines, Logistic Regression and Multinomial Naive Bayes) for the evaluation of the unambiguous quality characteristic of requirements. For each combination, their performance values are analyzed and compared with each other.*

Keywords— *Requirements Engineering, Requirements Quality, Machine Learning, natural language*

Resumo. *A Engenharia de Requisitos é uma etapa fundamental para o desenvolvimento de um software, assim como a qualidade dos requisitos desenvolvidos influenciam diretamente no sucesso do desenvolvimento. Geralmente a avaliação da qualidade dos requisitos é feita de forma manual pelos próprios engenheiros, trazendo maior lentidão para o processo, subjetividade e suscetibilidade a erros. Como uma alternativa para a avaliação, algoritmos de Aprendizagem de Máquina podem ser utilizados para a avaliação da qualidade dos requisitos escritos em linguagem natural de forma automatizada. Este artigo propõe uma análise comparativa entre a combinação de duas técnicas de vetorização ("Saco de palavras" e Frequência do Termo-Inverso da Frequência nos Documentos) e quatro algoritmos de aprendizagem supervisionada (k-Vizinho Mais Próximo, Máquinas de Vetores de Suporte, Regressão Logística e Multinomial Naive Bayes) para a avaliação da característica de qualidade de não ambiguidade dos requisitos. Para cada combinação são analisados seus valores de desempenhos e comparados entre si.*

Keywords— *Engenharia de Requisitos, Qualidade de Requisitos, Aprendizagem de Máquina, Linguagem natural*

Bacharelado em Engenharia de *Software* - PUC Minas
Trabalho de Conclusão de Curso (TCC)

Orientador de conteúdo (TCC I): Laerte Xavier - laertexavier@pucminas.br
Orientador acadêmico (TCC I): Lesandro Ponciano - lesandrop@pucminas.br
Orientador do TCC II: (A ser definido no próximo semestre)

Belo Horizonte, 14 de novembro de 2021.

1. Introdução

A Engenharia de Requisitos é uma importante função interdisciplinar da Engenharia de Software, responsável por identificar as necessidades das partes interessadas [Souza et al. 2020] e definir e manter os requisitos a serem atendidos pelo sistema, software ou serviço de interesse [ISO/IEC 2018]. Essa área desempenha um papel fundamental no processo de garantia de qualidade no desenvolvimento de software uma vez que ter requisitos de qualidade é o ponto inicial para o desenvolvimento de um software de qualidade. Paralelamente, a integração de tecnologias de Inteligência Artificial na Engenharia de Software tem como objetivo de otimizar o processo de desenvolvimento de produtos de software e automatizar tarefas intensivas de esforço, a fim de obter sistemas com alta qualidade [Gramajo et al. 2020b]. Sendo assim, a utilização da Inteligência Artificial apresenta-se como uma alternativa para a evolução da qualificação de requisitos de forma objetiva e automatizada.

Como os requisitos são escritos em linguagem natural, sua avaliação da sua qualidade geralmente se limita ao conhecimento e experiência dos engenheiros responsáveis, o que dificulta a padronização e análise rápida da qualidade dos requisitos. Atualmente, se estuda a possibilidade de automatização desse processo por meio das diferentes técnicas de Aprendizagem de Máquina disponíveis, porém não é claro qual método pode ser mais eficaz para esse objetivo. Os diferentes algoritmos de Aprendizagem de Máquina podem apresentar diferentes resultados para as métricas de desempenho dos modelos, sendo necessário compreender qual se adequaria melhor com contexto de classificação dos requisitos por qualidade. Sendo assim, o problema que este artigo procura resolver é a **falta de análises empíricas do emprego de algoritmos de classificação automática de requisitos de software escritos em linguagem natural em diferentes classes de qualidade de requisitos utilizando técnicas de Aprendizagem de Máquina**.

O processo de definição e validação de requisitos demanda grande consumo de tempo e esforço [Gramajo et al. 2020a], destacando a importância da evolução dessa etapa do desenvolvimento de um software. Além disso, como os requisitos são escritos em linguagem natural, a flexibilidade e as características inerentes da linguagem, como inconsistências, redundâncias e ambiguidades, podem levar a erros durante a especificação e, consequentemente, podem influenciar negativamente as fases posteriores do ciclo de vida do software [Gramajo et al. 2020b]. Somado a importância que a etapa de Engenharia de Requisitos tem para a garantia de qualidade do software desenvolvido, os pontos levantados garantem a necessidade do aprimoramento das práticas de avaliação da qualidade dos requisitos, de forma mais objetiva e automatizada ou o melhor desempenho possível.

Para analisar a melhor técnica de vetorização, são analisadas as duas técnicas mais conhecidas, sendo elas "Saco de palavras" (BoW, do inglês *Bag of Words*) e Frequência do Termo-Inverso da Frequência nos Documentos (TF-IDF, do inglês *Term Frequency—Inverse Document Frequency*). Essas técnicas são analisadas em combinação a quatro algoritmos de aprendizagem supervisionada, sendo eles: k-Vizinho Mais Próximo (k-NN, do inglês *k-Nearest Neighbor*), Máquinas de Vetores de Suporte (SVM, do inglês *Support Vector Machines*), Regressão Logística (LR, do inglês *Logistic Regression*), Multinomial Naive Bayes (MNB, do inglês *Multinomial Naive Bayes*)

Desse modo, este trabalho tem como objetivo **avaliar o emprego da classificação automática de requisitos de software expressos em linguagem natural, em termos de qualidade, utilizando técnicas de Aprendizagem de Máquina**. Para isso, os três objetivos específicos deste trabalho são: 1) classificar uma base de dados de requisitos pela característica de qualidade; 2) treinar os modelos de classificação automática; e 3) analisar as métricas de desempenho dos modelos, comparando-as entre si.

É esperado como resultado do trabalho uma análise comparativa entre os diferentes modelos para classificação dos requisitos por qualidade, apresentando sua diferença de desempenho em cada possível combinação de extração de recurso e algoritmo de Aprendizagem de Máquina. Para isso, são avaliadas métricas de desempenho como precisão, revocação e *F1-Score*.

Este trabalho está dividido da seguinte forma: A Seção 2 apresenta a fundamentação teórica. A Seção 3 aborda trabalhos relativos à classificação da qualidade dos requisitos usando técnicas de aprendizagem de máquina. Por fim, a Seção 4 relata a metodologia e materiais utilizados.

2. Fundamentação Teórica

Nesta seção são detalhados os principais conceitos e técnicas que estão envolvidos na solução do problema apresentado. São eles: Engenharia de Requisitos, Inteligência Artificial, Aprendizagem de Máquina e aprendizagem supervisionada e técnicas, algoritmos e métricas utilizados no desenvolvimento do estudo.

2.1. Engenharia de Requisitos

A Engenharia de Requisitos (ER) é a área da Engenharia de Software que fornece o mecanismo apropriado para entender o que o cliente deseja, analisar necessidades, avaliar a viabilidade, negociar uma solução razoável, especificar a solução, validar a especificação e gerenciar os requisitos à medida que são transformados em um sistema funcional [Thayer et al. 1997]. As atividades desse processo incluem: obtenção, análise, especificação, validação, verificação e gerenciamento de requisitos [Pohl 2010].

Os requisitos são definidos por sentenças em linguagem natural e para garantir a qualidade de um requisito individual é necessário que ele siga as seguintes características: necessário, implementação livre, não ambíguo, consistente, completo, singular, viável, rastreável e verificável [ISO/IEC 2018]. A definição de requisitos individuais com qualidade é a etapa inicial para garantia da qualidade de um software. Especialmente, a característica "não ambíguo", explorada nesta pesquisa, é a habilidade do requisito de poder ser interpretado por todos os envolvidos em um projeto de uma única maneira.

2.2. Inteligência Artificial

O termo Inteligência Artificial pode ser considerado um conceito relativamente amplo, considerando que não há um consenso final sobre seu significado, colecionando um conjunto de definições. Uma das definições mais utilizadas, apesar de não universalmente aceita, é a qual AI é o estudo de como fazer os computadores realizarem tarefas que no momento as pessoas são melhores [Rich 1985].

As diferentes vertentes relacionas à AI são definias como estudos das formas de estabelecer comportamentos "inteligentes" nas máquinas. Com o surgimento de novas estratégias de aprendizagem e o aumento do poder de processamento dos computadores, a AI tornou-se uma ferramenta mais acessível ao mercado e ao aprimoramento do desenvolvimento de sistemas de software [Feldt et al. 2018].

2.3. Aprendizagem de Máquina

Aprendizagem de Máquina (ML, do inglês *Machine Learning*) é uma disciplina da AI composta por um conjunto de técnicas que permitem aos computadores aprender dados e fazer generalizações e previsões a partir de deles, o que facilita a tomada de decisões [Gramajo et al. 2020a]. Embora não seja um conceito novo, a disponibilidade de grandes volumes de dados e maior capacidade de processamento em computadores permitiu experimentar e investigar ainda mais sobre seus usos e aplicações em vários domínios.

Uma das primeiras definições do conceito ML foi o campo estudo que dá aos computadores a capacidade de aprender algo, para o qual não foram programados explicitamente [Samuel 1959]. Uma definição mais formal é aquela expressa por Mitchell (1997), que afirma que um computador aprende uma tarefa específica T, considerando as experiências do tipo E, com relação a um medida de desempenho P, se o computador realmente melhorar seu desempenho P, na tarefa T, a partir da experiência E.

A aprendizagem de máquina pode ser classificado como supervisionada ou não supervisionada. É considerada uma aprendizagem supervisionada se os dados utilizados estão rotulados, classificados ou categorizados previamente. Caso contrário, a aprendizagem é considerada não supervisionada. Neste trabalho, é utilizada a aprendizagem supervisionada.

2.4. Aprendizagem Supervisionada

O aprendizagem supervisionada é uma técnica de ML que usa dados rotulados, chamados de dados de treinamento, para construir um modelo preditivo para prever o rótulo de dados não rotulados [Gramajo et al. 2020a]. O conjunto de dados de treinamento inclui dados de valores de entrada e resposta. A partir disso, o algoritmo de aprendizagem supervisionada busca criar um modelo que possa fazer previsões sobre os valores de resposta para um novo conjunto de dados. Normalmente é utilizado um conjunto de dados de teste para validar o modelo. Se são usados maiores conjuntos de dados de treinamento, é possível gerar modelos cuja capacidade preditiva seja maior, obtendo bons resultados em novos conjuntos de dados [Darnstädt et al. 2014].

2.5. Técnicas e Algoritmos

Nesta subseção são apresentados os conceitos de técnicas e algoritmos que serão utilizados no desenvolvimento do estudo.

2.5.1. Extração de Recurso

Algoritmos de aprendizagem de máquina operam em um espaço de recurso numérico, esperando a entrada como um *array* bidimensional onde as linhas são instâncias e colunas são recursos [Dias Canedo and Cordeiro Mendes 2020]. Para realizar aprendizagem de máquina em texto, como é o caso dos requisitos analisados, é preciso transformar as sentenças em representações vetoriais para aplicação do aprendizagem de máquina numérico. O processo de codificação de documentos em um formato numérico de recurso é chamado de extração de recurso ou, mais simplesmente, vetorização e é uma primeira etapa essencial para a análise da linguagem natural [Dias Canedo and Cordeiro Mendes 2020].

Para a finalidade comparativa entre as alternativas para extração de recurso, são utilizadas neste trabalho as técnicas "Saco de palavras" (BoW, do inglês *Bag of Words*) e Frequência do Termo-Inverso da Frequência nos Documentos (TF-IDF, do inglês *Term Frequency—Inverse Document Frequency*)

BoW: é uma forma de representar o texto de acordo com a ocorrência das palavras nele. A definição de "saco de palavras" é relacionado ao fato de não levar em conta a ordem ou a estrutura das palavras no texto, apenas se ela aparece ou a frequência com que aparece.

TF-IDF: utiliza de medidas estatísticas para medir o quão importante uma palavra é em um documento (texto) por meio de uma pontuação. O TF-IDF de uma palavra em um texto é feito multiplicando duas métricas diferentes: a frequência do termo (TF, do inglês *Term Frequency*), que mede a frequência com que um termo ocorre num documento, e o inverso da frequência nos documentos (IDF, do inglês *Inverse Document Frequency*), que mede o quão importante um termo é no contexto de todos os documentos.

2.5.2. Algoritmos de Aprendizagem de Máquina

Neste trabalho são comparadas a utilização de quatro diferentes algoritmos de aprendizagem supervisionada, sendo eles k-Vizinho Mais Próximo (k-NN, do inglês *k-Nearest Neighbor*, Máquinas de Vetores de Suporte (SVM, do inglês *Support Vector Machines*), Regressão Logística (LR, do inglês *Logistic Regression*) e Multinomial Naive Bayes (MNB, do inglês *Multinomial Naive Bayes*.

k-NN: baseia-se no princípio de que as instâncias dentro de um conjunto de dados geralmente existirão em estreita proximidade com outras instâncias que têm propriedades semelhantes. O algoritmo classifica um novo dado calculando a distância desses dados com as instâncias já existentes dentro da base de dados. A seguir, seleciona as k instâncias mais próximas e calcula sua média, para problemas de regressão, ou obtém a moda.

SVM: é um modelo de aprendizagem de máquina capaz de realizar classificação linear ou não linear, regressão e até detecção de *outliers*. O algoritmo faz a classificação criando um hiperplano linear de margem máxima que separa duas classes. Essa margem faz com que haja pouca possibilidades de separar os dados da amostra e, portanto, haja pouca chance de classificação incorreta de novas instâncias.

LR: A regressão logística é comumente usada para estimar a probabilidade de uma instância pertencer a uma classe específica. Abdul et al. (2019) definem a regressão logística como uma classe de regressão na qual uma variável independente é usada para prever a variável dependente. É chamada de regressão logística binária quando a variável dependente possui duas classificações. É chamada de regressão logística multinomial quando a variável dependente possui mais de duas classes. Neste trabalho, utilizamos regressão logística binária para classificar os requisitos entre ambíguo ou não ambíguo.

MNB: é um modelo generativo que estima a probabilidade condicional de uma classe com dados de entrada. Em particular, *Naive Bayes* assume que os recursos de entrada são independentes entre si (independência condicional). O Multinomial Naive Bayes é uma versão especializada do Naive Bayes usado principalmente para a classificação de documentos e textos.

2.6. Métricas

As métricas de avaliação são usadas principalmente para avaliar o desempenho de um classificador. O desempenho é verificado por meio de fórmulas matemáticas que irão comparar as previsões obtidas por um modelo com os valores reais do banco de dados. As métricas analisadas para efeito de comparação entre os métodos implementados serão a equação de precisão, a equação de revocação (*recall*) e a equação de *F1-Score*. Para o melhor entendimento da definições dessas métricas é levado em conta a Tabela 1 que indica os erros e acertos do modelo, comparando com o resultado esperado, chamada de matriz de confusão. Nela, o conceito de Verdadeiro Positivo, Verdadeiro Negativo, Falso Positivo e Falso Negativo são definidos de acordo com a resposta da classificação feita e a resposta real na base de dados.

Tabela 1. Matriz de Confusão

		Classificação feita	
		Sim	Não
Real	Sim	Verdadeiro Positivo (VP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (VN)

2.6.1. Equação de Precisão

A precisão mede a porcentagem entre a quantidade de amostras classificadas positivamente de forma correta em relação ao total de amostras positivas. A precisão é calculada usando a razão entre o total de classificações positivas corretas e o total de classificações da classe positivo realizadas. Esse cálculo também pode ser visto como a razão da quantidade de verdadeiros positivos (VP) para a quantidade de positivos detectados pelo modelo (VP + FP), como definido na Equação 1.

$$Precisão = \frac{VP}{VP + FP} \tag{1}$$

Uma precisão igual a 1 indica uma porcentagem de 100% do classificador e que todas as instâncias foram classificadas corretamente.

2.6.2. Equação de Revocação (*Recall*)

O *Recall* é uma proporção de instâncias positivas que são detectadas corretamente pelo classificador. Como definido na Equação 2, ele é calculado através do número de vezes que uma classe positivo foi prevista corretamente (VP), dividido pelo número de vezes que a classe aparece nos dados do teste (VP + FN). Em outras palavras, calcula entre todas as situações de classe positivo como valor esperado, quantas estão corretas.

$$Recall = \frac{VP}{VP + FN} \tag{2}$$

2.6.3. Equação de *F1-Score*

O *F1-Score* é considerado a média harmônica entre precisão e a revocação, levando em conta as duas métricas simultaneamente e seu cálculo é definido na Equação 3. Enquanto a média regular trata todos os valores igualmente, a média harmônica dá muito mais peso aos valores baixos. Como resultado, o classificador só obterá um *F1-Score* alto se a revocação e a precisão forem altos [Dias Canedo and Cordeiro Mendes 2020].

$$F1 - score = \frac{2 * (Precisão * Recall)}{Precisão + Recall} \tag{3}$$

3. Trabalhos Relacionados

Nesta seção são discutidos estudos que apresentam propostas relacionadas à utilização de aprendizagem de máquina para a avaliação de qualidade de requisitos. O primeiro artigo apresenta uma análise comparativa com as mesmas combinações de técnicas avaliadas neste trabalho, porém para classificação de entre tipos de requisitos. Em seguida é abordado um artigo que explora uma abordagem completa de avaliação e sugestão automática de aprimoramento de requisitos. Posteriormente, os dois artigos seguintes exploram a análise de características da qualidade, sendo elas a incerteza e a singularidade, respectivamente. Por fim, os estudos seguintes apresentam um estudo utilizando Redes Bayesianas e, por fim, a identificação de ambiguidades nocivas.

Dias et al. (2020) apresentam uma proposta muito semelhante em relação a análise comparativa de técnicas de aprendizagem de máquina para melhor desempenho na classificação de requisitos. O artigo apresenta a mesma combinação de duas técnicas de vetorização (BoW e TF-IDF) e quatro algoritmos de aprendizagem supervisionada (k-NN, SVM, LR e MNB) para análise de desempenho, entretanto, tem como objetivo classificar os requisitos entre Requisitos Funcionais e Não-Funcionais e subclassificar esse segundo grupo. O melhor resultado encontrado foi a utilização do TF-IDF seguido pelo uso do LR. O diferencial deste artigo em relação ao Dias et al. (2020) é o foco na classificação da qualidade, mais precisamente na não ambiguidade dos requisitos.

Ter um conjunto de requisitos da mais alta qualidade possível é de grande importância e os benefícios incluem melhorar a qualidade do projeto, entender melhor as necessidades do cliente, reduzir custos e prever cronogramas e resultados do projeto com maior precisão [Adanza Dopazo et al. 2021]. Assim, o objetivo principal nesse artigo é criar uma metodologia que possa modificar requisitos pobres, extraindo as principais características de cada requisito, avaliando sua qualidade com alto nível de especialidade e, então, aprimorando o qualidade dos requisitos. Na primeira etapa, foi implementado um algoritmo de aprendizagem de máquina para classificar os requisitos com base na qualidade e identificar aqueles que são mais prováveis de serem problemáticos. E na segunda etapa, o algoritmo genético gerou soluções para melhorar a qualidade dos requisitos identificados como inferiores. O algoritmo genético obteve resultados que podem ser comparados com a solução teoricamente ótima, definindo essa ferramenta como promissora para a engenharia de requisitos.

As partes interessadas frequentemente usam linguagem especulativa quando precisam transmitir seus requisitos com algum grau de incerteza [Yang et al. 2012]. Devido à imprecisão intrínseca da linguagem especulativa, os requisitos especulativos correm o risco de serem mal interpretados e a incerteza relacionada negligenciada, Assim, podem se beneficiar de um tratamento cuidadoso no processo de engenharia de requisitos. Neste artigo, é apresentada uma abordagem orientada linguisticamente para a detecção automática de incerteza em requisitos de Linguagem Natural (NL). Primeiro, são identificadas sentenças especulativas aplicando um algoritmo de aprendizagem de máquina chamado Campos Aleatórios Condicionais (CRFs, do inglês *Conditional Random Fields*) para identificar pistas de incerteza. O algoritmo explora um conjunto de recursos lexicais e sintáticos extraídos de sentenças de requisitos. Em segundo lugar, a abordagem tenta determinar o escopo da incerteza. É utilizada uma abordagem baseada em regras que se baseia em um conjunto de heurísticas linguísticas feitas à mão para determinar o escopo da incerteza com a ajuda das estruturas de dependência presentes na árvore de análise sintática da frase. Os resultados aparentam promissores para a implementação da avaliação da incerteza de requisitos reais. Assim como este trabalho, Yang et al. (2012) estudam o campo da análise automática da incerteza de requisitos, porém avaliam apenas a utilização do algoritmo CRFs e não têm a intenção de análise comparativa entre os algoritmos.

Um dos desafios para a avaliação da qualidade de requisitos é lidar com um texto escrito em linguagem natural [Gramajo et al. 2020b]. Nesse artigo, Gramajo et al. (2020) propõe outro estudo da qualidade de requisitos desenvolvidos por meio de redes neurais. Utilizando da definição de qualidade definida pela norma ISO/IEC/IEEE 29148:2018, o estudo foca em um dos tópicos de qualidade do requisito, a singularidade. Para isso, é proposto a utilização de redes neurais, do tipo Memória Longa de Curto Prazo (LSTM, do inglês *Long Short-Term Memory*), para prever se os requisitos são singulares ou não. Cada requisito é submetido a um processo de tokenização e rotulagem gramatical, a fim de obter uma sequência representativa que atua como uma entrada para o modelo neural. E então o modelo neural proposto é treinado com um conjunto de dados de 1000 requisitos, sendo 80% para realizar o treinamento da rede neural, 10% para teste e 10% para validação. Os resultados apresentados pelo modelo são definidos como promissores e se destaca o interesse em prosseguir a pesquisa com os demais tópicos de qualidade de um requisito. Gramajo et al. (2020) estudam a avaliação da característica de requisitos definida como

singularidade, enquanto este trabalho lhe complementa com o estudo da característica de não ambiguidade.

A área de Engenharia de Requisitos é crucial para o desenvolvimento de um software que não se afaste das necessidades do cliente, mas que muitas vezes é refém da expertise e experiência do profissional responsável [Wiesweg et al. 2020]. Isso é apontado como uma abertura para falhas e, portanto, riscos. Com isso, Wiesweg et al. (2020), na tentativa de auxiliar esse gerenciamento de risco, desenvolveram solução utilizando Redes Bayesianas. Para isso, foram avaliadas diferentes versões de Redes Bayesianas que modelam as relações entre causas, problemas e efeitos em ER. Os modelos foram treinados com os dados que foram coletados por meio de duas pesquisas com respostas de 228 e 488 profissionais, sobre problemas, causas e efeitos encontrados em projetos reais. Os modelos tinham como objetivo realizar uma análise *post-mortem* (dado o problema, diagnosticar a causa) e preditiva (dado a causa, diagnosticar possíveis problemas). Ambos modelos apresentaram bons valores de *recall* e precisão, demonstrando uma possível abertura para utilização em gerenciamento de riscos reais. Para este artigo, o trabalho de Wiesweg et al. (2020) é utilizado como referência na utilização de aprendizagem de máquina na análise consequências de requisitos com baixa qualidade para o desenvolvimento de software.

Em um outro artigo, Yang et al. (2010) propõe identificar automaticamente ambiguidades potencialmente nocivas dos requisitos, que ocorrem quando o texto é interpretado de forma diferente por diferentes leitores. Para isso, foi extraído um conjunto de ambiguidades junto de vários julgamentos humanos sobre suas interpretações. A distribuição de julgamento foi usada para determinar se a ambiguidade é nociva ou inócua. Utilizando técnicas de aprendizagem de máquina, foi desenvolvida uma ferramenta automatizada para prever a preferência antecedente de candidatos a sintagmas nominais, que por sua vez é usada para identificar ambiguidades nocivas. O estudo mostrou resultados que permitem explorar e destacar ambiguidades realistas e potencialmente problemáticas em documentos de requisitos reais. Esse artigo apresenta um estudo aprofundado sobre a identificação da nocividade ambiguidades dos requisitos, característica a ser avaliada nos algoritmos utilizados este artigo.

4. Materiais e Métodos

Este trabalho é classificado como uma pesquisa quantitativa e descritiva, uma vez que objetiva avaliar e trazer uma análise comparativa entre as técnicas de Aprendizagem de Máquina na classificação de qualidade automática de requisitos de software expressos em linguagem natural. Assim, são utilizados dois diferentes métodos de extração de recurso e quatro diferentes algoritmos de aprendizagem de máquina no contexto de classificação de qualidade dos requisitos. Nesta seção são apresentado os materiais, métodos, métricas de avaliação e etapas de execução da pesquisa.

4.1. Materiais

Para a execução da pesquisa, é utilizada a versão 3.10 da linguagem Python (`https://www.python.org/`) e a base de dados utilizada será a PROMISE_exp [Lima et al. 2019] disponível em `https://tinyurl.com/PROMISE_exp`. O PROMISE_exp é uma versão expandida do original conjunto PROMISE, um repositório

público inspirado no *UCI Machine Learning Repository*, e criado para encorajar modelos preditivos repetíveis, verificáveis, refutáveis e/ou improváveis de engenharia de software. O repositório original consiste em um conjunto pré-rotulado em classificação (Requisitos Funcionais e Não-Funcionais) de 255 RFs e 370 RNFs. A versão expandida consiste em 969 requisitos. Para utilização dessa base de dados, é necessária a pré-rotulação desses requisitos de acordo com o valor da característica de qualidade a ser avaliada, no caso deste trabalho, a não ambiguidade.

4.2. Métodos e Métricas

Na Seção 2, relativa a fundamentação teórica, são apresentados os conceitos das técnicas, algoritmos de aprendizagem de máquina supervisionados e equações das métricas utilizada no desenvolvimento da pesquisa.

Os requisitos são classificadas duas classes, sendo ”Ambíguo”e ”Não Ambíguo”. Seguindo a definição de identificação de ambiguidade definida por Yang et al. (2010), o classificador deve identificar se há pelo menos uma resposta negativa nos recursos de linguagens apresentados na Tabela 2 para ser considerado ambíguo.

Tabela 2. Recursos para Classificador para Ambiguidade

Tipo de Recurso	Recurso
Linguística	Acordo de número
	Definição
	Frase não preposicional
	Restrição sintática
	Paralelismo sintático
	Frase não associada
	Restrição semântica
	Paralelismo semântico
Contexto	Centrando
	Recência da frase
	Proximal

Para extração de recurso, são utilizadas as duas técnicas mais conhecidas, BoW e TF-IDH, definidas anteriormente na Seção 2.5.1. Em relação aos algoritmos de aprendizagem de máquina supervisionados, são utilizados k-NN, SVM, LR, e MNB. Para cada combinação entre técnica de extração de recurso e um algoritmo, é utilizada a base de dados rotulados com o intuito de promover um processo de aprendizagem supervisionada. A técnica de extração de recurso será responsável pelo processo de codificação do requisito em um formato numérico e o algoritmo é alimentando por pares de entradas e saídas conhecidas, então em forma de vetores. Para cada resultado encontrado, são utilizadas as métricas de precisão, *recall* e *F1-Score* para a análise comparativa entre o desempenho de cada combinação.

4.3. Etapas de Execução

Para execução do trabalho são consideradas as seguintes etapas:

1. Coleta da base de dados de requisitos;

2. Classificação requisitos da base de dados por característica de qualidade e incrementação da base de dados. A base será classificada entre requisitos ambíguos ou não ambíguos;
3. Normalização: limpeza dos dados, onde todas as palavras irrelevantes, como pronomes e artigos, são removidas. Verbos e substantivos flexionados são convertidos para sua forma original;
4. Extração de Recurso: transformação do texto em uma informação numérica. Neste estudo, são utilizado BoW e TF-IDF para realizar a conversão;
5. Treinamento: vetores obtidos na fase de Extração de Recurso são usados para treinar e predizer os modelos de classificação dos quatro algoritmos usados neste trabalho: SVM, MNB, k-NN e LR;
6. Avaliação: resultados das previsões dos rótulos dos requisitos e os rótulos verdadeiros desses requisitos são usados para calcular as medidas de desempenho descritas na Seção 2.6.

4.4. Cronograma

A Tabela 3 apresenta o cronograma para desenvolvimento da pesquisa entre a primeira quinzena de fevereiro de 2022 e a primeira quinzena de maio de 2022.

Tabela 3. Cronograma de Execução 2022.1

Início da Atividade	Atividade
Primeira quinzena de fevereiro	Revisão de metodologia e coleta de dados
Segunda quinzena de fevereiro	Classificação e incrementação da base de requisitos
Primeira quinzena de março	Normalização
Segunda quinzena de março	Extração de Recurso (BoW e TF-IDF)
Primeira quinzena de abril	Treinamento dos modelos (SVM e MNB)
Segunda quinzena de abril	Treinamento dos modelos (k-NN e LR)
Primeira quinzena de maio	Avaliação

Referências

Adanza Dopazo, D., Moreno Pelayo, V., and Génova Fuster, G. (2021). An automatic methodology for the quality enhancement of requirements using genetic algorithms. *Information and Software Technology*, 140:106696.

Darnstädt, M., Simon, H. U., and Szörényi, B. (2014). Supervised learning and co-training. *Theoretical Computer Science*, 519:68–87. Algorithmic Learning Theory.

Dias Canedo, E. and Cordeiro Mendes, B. (2020). Software requirements classification using machine learning algorithms. *Entropy*, 22(9).

Feldt, R., de Oliveira Neto, F. G., and Torkar, R. (2018). Ways of applying artificial intelligence in software engineering. In *2018 IEEE/ACM 6th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE)*, pages 35–41.

Gramajo, M., Ballejos, L., and Ale, M. (2020a). Seizing requirements engineering issues through supervised learning techniques. *IEEE Latin America Transactions*, 18:1164–1184.

Gramajo, M. G., Ballejos, L. C., and Ale, M. (2020b). Hacia la evaluación automática de la calidad de los requerimientos de software usando redes neuronales long short term memory. In *Workshop on Requirements Engineering (WER)*.

ISO/IEC (2018). *ISO/IEC/IEEE International Standard - Systems and software engineering – Life cycle processes – Requirements engineering*. ISO/IEC.

Lima, M., Valle, V., Costa, E., Lira, F., and Gadelha, B. (2019). Software engineering repositories: Expanding the promise database. In *SBES 2019: Proceedings of the XXXIII Brazilian Symposium on Software Engineering*, pages 427–436.

Pohl, K. (2010). *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer Publishing Company, Incorporated, 1st edition.

Rich, E. (1985). Artificial intelligence and the humanities. *Computers and the Humanities*, 19(2):117–122.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3):210–229.

Souza, J. H. J., Marques, L. C., Conte, T. U., and Zaina, L. A. M. (2020). Descrevendo requisitos de user experience em critérios de aceitação de user stories. In *Workshop on Requirements Engineering (WER)*.

Thayer, R. H., Bailin, S. C., and Dorfman, M. (1997). *Software Requirements Engineerings, 2nd Edition*. IEEE Computer Society Press, Washington, DC, USA, 2nd edition.

Wiesweg, F., Vogelsang, A., and Mendez, D. (2020). Data-driven risk management for requirements engineering: An automated approach based on bayesian networks.

Yang, H., De Roeck, A., Gervasi, V., Willis, A., and Nuseibeh, B. (2012). Speculative requirements: Automatic detection of uncertainty in natural language requirements. In *2012 20th IEEE International Requirements Engineering Conference (RE)*, pages 11–20.