Classificação de Ransomware utilizando Algoritmos de Machine Learning

Adriana Medeiros Pinheiro
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Pará (UFPA)
Belém, Brasil
adrianamprf@gmail.com

George Tassiano Melo Pereira
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Pará (UFPA)
Belém, Brasil
george.melo7@gmail.com

Caio Carvalho Moreira
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Pará (UFPA)
Belém, Brasil
caiomoreira@ufpa.br

Claudomiro de Souza Sales Júnior
Programa de Pós-Graduação em Engenharia Elétrica
Universidade Federal do Pará (UFPA)
Belém, Brasil
cssj@ufpa.br

***Resumo*—*Ransomware* é um subconjunto de *malwares* que vem crescendo como uma grave ameaça cibernética. Ele impede ou limita os usuários de acessar seu sistema e/ou seus arquivos até que o resgate seja pago. Algoritmos de *Machine Learning* (ML) vêm sendo usados nos modelos de classificação automatizada. Neste artigo, exploramos 11 algoritmos de ML, juntamente com a aplicação da técnica *Principal Component Analysis* (PCA) com o objetivo encontrar os melhores algoritmos classificadores para o problema de classificação de *ransomware*. Foram abordados cinco métodos de classificação usados pela literatura. Dentre os destaques, o classificador Naive Bayes se mostrou eficiente, pois obteve até 100% de acurácia em oito famílias em um dos métodos.**

***Palavras-chave*—ransomware, machine learning, classificação**

I. Introdução

De acordo com um relatório da empresa de segurança cibernética *Cybersecurity Ventures*, foi previsto que em 2021 o *ransomware* trará danos globais estimados em 6 trilhões de dólares [1]. Gangues de *ransomware* estão cada vez mais se aperfeiçoando e alcançando um alto número de vítimas, seja usuários finais comuns, organizações empresariais, federais, bancos, prefeituras, centros médicos, etc.

O *Ransomware* é um *malware* projetado para restringir o acesso a um sistema ou dados até que um valor de resgate solicitado pelo cibercriminoso seja satisfeito. Ele pode ser classificado em dois tipos: criptográfico ou de bloqueio. O primeiro criptografa os arquivos da vítima, e *ransomware* de bloqueio evita que as vítimas acessem seus sistemas, mas independentemente de qual for o tipo, ambos exigem um pagamento de resgate para liberar os arquivos ou acesso ao sistema [2]. Ainda de acordo com [2], o ataque de *ransomware* consiste em cinco fases conforme mostrado na Figura 1.

O *ransomware* gera uma grande quantidade de dados de alta dimensão, e fazer uma classificação sem usar uma inteligência computacional consome um alto custo de tempo e processamento. Processar e analisar dados de alta dimensão tornou-se um desafio para pesquisadores que trabalham em

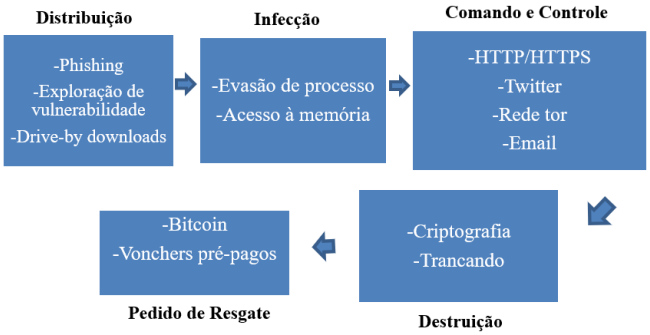

Fig. 1. Fases do Ransomware.

várias disciplinas, especialmente aprendizado de máquina e mineração de dados [3].

Nessa perspectiva, a análise de componentes principais (PCA) pode ser usada para fazer uma extração de atributos, reduzindo o espaço dimensional, pois em problemas de classificação com uma quantidade muito alta de atributos, pode ocorrer uma série de efeitos negativos conhecidos na literatura como a maldição da dimensionalidade. Por essa razão, métodos de redução de dimensionalidade têm sido aplicados amplamente em problemas reais [4].

O aprendizado de máquina é uma aplicação de Inteligência Artificial (IA) que fornece ao sistema habilidades para aprender automaticamente e melhorar a experiência sem programá-la implicitamente. Este processo de aprendizagem começa com a observação (dados), com o objetivo de reconhecer padrões e, em seguida, tomar uma decisão melhor com base no conhecimento aprendido. Isso permite que os computadores aprendam automaticamente sem intervenções humanas, assim como mostra [5].

Neste artigo, exploramos 11 algoritmos de ML, juntamente com a aplicação da técnica PCA para a redução de dimen-

sionalidade dos dados com o objetivo encontrar os melhores algoritmos classificadores para o problema de detecção de *ransomware*. Foram abordados cinco métodos de classificação usados pela literatura. Dentre os destaques, o classificador Naive Bayes entre todos os métodos, obteve até 100% de acurácia em oito famílias em um dos métodos.

II. Revisão da Literatura

O aprendizado de máquina tem sido muito eficaz na detecção de *malwares*, como mostram os trabalhos de [5], [6] [7], [8] e [9]. O *ransomware* tem sido um tópico de pesquisa muito ativo e vários pesquisadores propuseram pesquisas que se concentram em diferentes aspectos da pesquisa de *ransomware*.

Para uma melhor análise do *ransomware* e toda sua evolução, o trabalho de [2] realizou uma pesquisa abrangente sobre *ransomware* e soluções de defesa em relação a PCs / estações de trabalho, dispositivos móveis e plataformas *Cyber-Physical Systems* (CPS) e *Internet of Things* (IoT). A pesquisa cobriu 137 estudos durante o período de 1990-2020, apresentou uma visão geral detalhada da evolução do *ransomware*, analisou de forma abrangente os principais blocos de construção do *ransomware*, apresentou uma taxonomia de famílias de *ransomware* mais notáveis e listou uma série de questões em aberto para pesquisas futuras de *ransomware*.

O trabalho de [5] realizou uma pesquisa sobre a evolução da detecção de *ransomware* usando aprendizado de máquina e técnicas de aprendizado profundo. O artigo avaliou 19 trabalhos, fazendo a abordagem algorítmica, o processo de engenharia de recursos, bem como a avaliação de cada resultado. O artigo é de extrema relevância, pois explora as novas direções do *ransomware* e como é esperada a evolução do *ransomware* nos próximos anos.

O trabalho de [10] propôs um método de detecção de *ransomware* que pode distinguir entre *ransomware* e arquivos benignos, bem como entre *ransomware* e outros tipos de *malware*. Nos experimentos foram usados seis algoritmos de aprendizagem de máquina, tais como: RF *(Random Forest)*, RL (Regressão Logística), NB *(Naive Bayes)*, SGD *(Stochastic Gradient Descent)*, KNN *(K-Nearest Neighbours)* e SVM *(Support Vector Machine)*.

O trabalho de [11] propôs um método de seleção de recursos que usa otimização por enxame de partícula (PSO) para detecção e classificação de *ransomware* usando dados de análise de comportamento de alta dimensão de *ransomware* e *goodware*. Os resultados do artigo mostram que o desempenho de classificação do modelo depende do número de recursos selecionados de cada um dos grupos de recursos no conjunto de dados. Os autores alcançaram uma acurácia média de 50% para as famílias de *ransomware*.

O trabalho de [12] realizou uma classificação chamada ERAND *(Ensemble Ransomware Defense)* para defesa contra *ransomware*. Os autores usaram o NSGA-II para calcular os pesos de cinco classificadores *(ExtraTree, Gradient Boosting, AdaBoost, XGBoost e Random Forest)* e alcançaram uma precisão elevada, encontrando precisões para cada família acima de 95%. Entretanto a metodologia usada pelos autores ficou bastante obscura, e causou várias interpretações distintas.

O trabalho de [9] realizou uma predição de *ransomware* usando aprendizado supervisionado com os algoritmos *Support Vector Machines* (SVM), *Random Forest* (RF), *Decision Tree* (DT), Rede Bayesiana (BN), Rede Neural Artificial (AN), Regressão Logística (RL). A pesquisa obteve um melhor desempenho com SVM com uma precisão de 88,2%.

Já o trabalho de [8] abordou o problema do ajuste fino dos classificadores Naive Bayes (NB) para dois conjuntos de dados desequilibrados, e um deles foi o que utilizamos para nosso experimento. Ele propôs o (Fine Tuning Naive Bayes) FTNB-ID e comparou com Naive Bayes e FTNB original.

O trabalho de [3] propôs um modelo de detecção de *ransomware* baseado em métodos de aprendizado de máquina usando dados de tráfego de rede. Os pesquisadores monitoraram a comunicação de rede entre a máquina da vítima e o comando e controle (C&C) para detectar e impedir a entrega da chave de criptografia necessária para a criptografia dos arquivos da vítima sem a qual o processo de criptografia não foi iniciado. Os autores usaram técnicas de redução de dimensionalidade para encontrar os oito atributos que mais contribuem para a detecção de *ransomware* no tráfego de rede. No entanto, a solução sofre por ter uma taxa de falsos positivos de 12,5% , o que pode gerar muitos alarmes falsos.

O trabalho de [13] realizou uma análise dinâmica automatizada de *ransomware*. Os autores criaram o sistema EldeRan baseado na observação de que o *ransomware* executa certas ações que são únicas ou significativamente diferentes daquelas executadas por software benigno. O sistema monitora um ambiente em *sandbox (Cuckoo Sandbox)* e extrai recursos nas seguintes classes: chamadas de API do Windows, operações de chave de registro, operações de sistema de arquivo, operações de diretório, o conjunto de operações feitas por extensão de arquivo, arquivos descartados e as strings do executável. O componente de seleção de características do sistema usa o critério de informação mútua, que permite que as características mais discriminantes sejam obtidas. Os recursos são fornecidos ao classificador de regressão logística regularizado para classificar os executáveis como benignos ou maliciosos. O principal problema com essa abordagem, é que esse método se esforça para detectar *ransomware* que permanece inativo por um longo período ou requer a entrada do usuário para ativar o executável devido à dependência de técnicas de *sandbox*. A regressão logística regularizada não é muito propícia com limites de decisão não lineares, o que significa que o modelo pode achar difícil encontrar relacionamentos complexos entre recursos.

O trabalho de [14] propôs que o ransomware de muitas famílias poderia ser detectado por meio do monitoramento eficaz da atividade do sistema de arquivos. Os autores realizaram uma análise dinâmica em grande escala de 1359 *ransomware* provenientes de 15 famílias diferentes, encontradas entre os anos de 2006 e 2014. Posteriormente, [15] propôs uma solução de análise e detecção de *ransomware* chamada *UNVEIL*, o qual utilizou padrões de rastreamentos de entrada e

saída (E/S) deixados por processos de nível de usuário como comportamento de assinatura para detecção de *ransomware*. Infelizmente, o *UNVEIL* foi capaz de detectar *ransomware* somente depois que alguns arquivos na máquina da vítima já haviam sido criptografados, portanto, a solução deixou as vítimas parcialmente desprotegidas. Além disso, os próprios autores mencionaram que o *UNVEIL* pode ser impedido por qualquer *ransomware* executado com privilégios no nível do *kernel*.

Usaremos para nossas comparações os trabalhos de U. Adamu, et al. (2019), M. S. Abbasi, et al. (2020) e P. Borah, et al.(2021) [9], [11] e [12] nos quais os autores trabalharam com o mesmo *dataset* e algoritmos que utilizamos para nossos experimentos.

III. Materiais e Métodos

A. Dataset

As amostras de *ransomware* do *dataset* utilizado para este artigo foi baixado no VirusShare, um site que mantém um banco de dados continuamente atualizado de *malware* para vários sistemas operacionais. Ele foi disponibilizado por [13] em que propôs uma solução anti-ransomware, chamada Elderan como foi mencionado na revisão de literatura. As amostras foram executadas por 30 segundos em um ambiente de análise dinâmica para registrar seu comportamento em termos de operações do programa utilizado. No total, 30.967 atributos categorizados em sete grupos foram registrados. O conjunto de dados foi analisado no final de fevereiro de 2016, e é composto por 582 amostras de trabalho de *ransomware* pertencentes a 11 diferentes classes e 942 de *goodwares*. De acordo com [16], nos últimos anos, os cibercriminosos começaram a projetar novas famílias de *ransomware* que visam vítimas específicas. Um exemplo é *Ryuk*, visto em 2019, que foi projetado para atingir apenas empresas.

As amostras de *ransomware* coletadas são representativas das versões e variantes mais populares atualmente encontradas na natureza (a maioria delas pertence ao tipo *cryptoransomware*). Para treino e teste foi realizada uma validação cruzada do tipo divisão aleatória estratificada com instâncias aleatoriamente distribuídas em quatro pacotes. Essa divisão é uma variação de *Shuffle Split*, que retorna divisões estratificadas, ou seja, que cria divisões preservando a mesma porcentagem para cada classe de destino como no conjunto completo. A tabela I mostra o resumo das instâncias do conjunto de dados utilizado.

Durante a fase de pré-processamento do *dataset* foi feito um relatório usando o pandas profiling da biblioteca pandas da linguagem *Python*. O *Pandas profiling* é um módulo *Python* de código aberto que permite fazer uma análise exploratória de dados. No relatório da análise, verificou-se que há 110 duplicadas, correspondente 7.2% do *dataset*, o que pode impactar o desempenho da classificação, já que há instâncias que tem o mesmo comportamento e são de diferentes classes. A Tabela II apresenta um resumo do grupo de features no conjunto de dados.

TABELA I
Resumo das instâncias do conjunto de dados.

Família	# Instâncias
Critroni	50
CryptLocker	107
CryptoWall	46
Kollah	25
Kovter	64
Locker	97
Matsnu	59
Pgpcoder	4
Reverton	90
TeslaCrypt	6
Trojan-Ransom	34
Goodware	942
Total	1524

TABELA II
Resumo do grupo de features do conjunto de dados.

Grupo	Descrição	# Features
API	Chamadas de API	232
DROP	Extensões de arquivos removidos	346
REG	Operações de Registro Chave	6622
FILES	Operações de arquivos	4141
FILES_EXT	Extensões de arquivos manipulados	935
DIR	Operações em diretório de arquivos	2424
STR	Strings embutidas	16267
Total	-	30967

B. Extração de atributos via redução de dimensionalidade

Algoritmos de mineração de dados e aprendizado de máquina podem enfrentar a maldição dos problemas de dimensionalidade ao lidar com dados dimensionais elevados [17]. Além disso, os modelos de aprendizagem podem se encaixar na presença de um grande número de recursos que podem levar à degradação do desempenho, além de aumentar os requisitos de memória e o custo computacional. Portanto, é necessário remover recursos irrelevantes. Uma vez que o conjunto de dados têm um grande número de recursos (30.967), realizamos a aplicação do PCA com o objetivo de encontrar um subconjunto ideal de recursos.

O PCA é o método mais antigo, simples e robusto de realizar redução de dimensionalidade. A técnica foi redescoberta muitas vezes em diversas áreas da ciência, sendo conhecida também como Transformação de Karhunen-Loeve, Transformação de Hotelling ou decomposição em valores singulares. O principal objetivo da técnica é encontrar uma representação com menos redundância entre os dados, eliminando as dependências entre as variáveis que compõe as diferentes dimensões do vetor aleatório X.

Ainda de acordo com [17] existem dois critérios para encontrar as componentes principais, como: problema de maximização usando na função objetivo a variância dos dados retidos, ou a minimização do erro quadrático médio (MSE). Para este trabalho aplicamos o PCA com 99% de variância, o que resultou em 866 componentes principais.

C. Algoritmos de machine learning

Em nossos experimentos, onze algoritmos de aprendizagem de máquina foram usados: Regressão Logistica (RL), *Nearest*

Neighbours (NN), Linear SVM (LSVM), RBF SVM, *Gaussian Process* (GP), *Decision Tree* (DT), *Random Forest* (RF), *AdaBoost*, *Naive Bayes* (NB), *Naive Bayes Bernoulli* (NBB), e *Quadratic Discriminant Analysis* (QDA). O seguinte explica resumidamente cada algoritmo. A seguir uma breve explicação de cada algoritmo.

- Regressão Logística (RL)

A regressão logística é o método usado para problemas de classificação binária (problemas com dois valores de classe), utilizando conceitos de estatística e probabilidade. É um algoritmo que lida com questões e problemas de classificação, analisando diferentes aspectos ou variáveis de um objeto para depois determinar uma classe na qual ele se encaixa melhor. A função logística parece um grande S e transformará qualquer valor no intervalo de 0 a 1. Isso é útil porque é possível aplicar uma regra à saída da função logística para ajustar valores para 0 e 1 e prever um valor de classe.

- Nearest Neighbours (NN)

Um dos algoritmos de decisão mais simples que podem ser usados para classificação é a regra do vizinho mais próximo (NN). Ele classifica uma amostra com base na categoria de seu vizinho mais próximo.

- Linear SVM (LSVM)

Linear SVM é um modelo de aprendizado supervisionado amplamente utilizado para reconhecimento de padrões ou análise de dados. O LSVM gera um modelo de classificação linear binária não probabilística que determina a qual categoria os novos dados pertencem.

- Radial basis function kernel support vector machine (RBF-SVM)

RBF-SVM é uma função de kernel popular usada em vários algoritmos de aprendizagem kernelizados. Em particular, é comumente usada na classificação de máquinas de vetores de suporte.

- Gaussian Process (GP)

Em teoria da probabilidade e estatística, um processo gaussiano é um modelo estatístico em que as observações ocorrem em um domínio contínuo, por exemplo, tempo ou espaço. Em um processo gaussiano, cada ponto em algum espaço de entrada contínua está associada com uma variável aleatória com distribuição normal. Além disso, cada conjunto finito dessas variáveis aleatórias tem uma distribuição normal multivariada. A distribuição de um processo gaussiano é a distribuição conjunta de todas as infinitas variáveis aleatórias, e, como tal, é uma distribuição de funções com um domínio contínuo.

Visto como um algoritmo de aprendizado de máquina, um processo gaussiano utiliza "aprendizagem preguiçosa" e uma medida da similaridade entre os pontos (a função *kernel*) para prever o valor de um ponto invisível a partir de dados de treinamento

- Decision Tree (DT)

O algoritmo da árvore de decisão pertence à família dos algoritmos de aprendizado de máquina supervisionado. Ele pode ser usado tanto para um problema de classificação, bem como para o problema de regressão. O objetivo deste algoritmo é criar um modelo que preveja o valor de uma variável de destino, para o qual a árvore de decisão usa a representação em árvore para resolver o problema em que o nó folha corresponde a um rótulo de classe e os atributos são representados no nó interno da árvore.

- Random Forest (RF)

Random Forest é um algoritmo de aprendizagem de máquina flexível e fácil de usar que produz excelentes resultados a maioria das vezes, mesmo sem ajuste de hiperparâmetros. É também um dos algoritmos mais utilizados, devido à sua simplicidade e o fato de que pode ser utilizado para tarefas de classificação e também de regressão. O algoritmo *random forest* adiciona aleatoriedade extra ao modelo, quando está criando as árvores. Ao invés de procurar pela melhor característica ao fazer a partição de nodos, ele busca a melhor característica em um subconjunto aleatório das características. Este processo cria uma grande diversidade, o que geralmente leva a geração de modelos melhores.

- AdaBoost

AdaBoost é melhor usado para aumentar o desempenho das árvores de decisão em problemas de classificação binária.O algoritmo *AdaBoost* foi um dos primeiros algoritmos de Boosting a ser amplamente usado em sistemas de tempo real, por causa da sua simplicidade e adaptabilidade.

- Naive Bayes (NB)

Naive Bayes é um algoritmo de classificação simples baseado no teorema Bayesiano. Um elemento pode ser classificado em uma classe entre várias classes.

- Naive Bayes Bernoulli (NBB)

Naive Bayes Bernoulli é uma variante do Naive Bayes. Ele é usado para dados discretos e funciona na distribuição Bernoulli. A principal característica do Bernoulli Naive Bayes é que ele aceita recursos apenas como valores binários como verdadeiro ou falso, sim ou não, sucesso ou falha, 0 ou 1 e assim por diante.

- Quadratic Discriminant Analysis (QDA)

A análise discriminante quadrática (QDA) está intimamente relacionada à análise discriminante linear (LDA), onde se assume que as medidas de cada classe são normalmente distribuídas. Ao contrário do LDA, no entanto, no QDA não há suposição de que a covariância de cada uma das classes é idêntica.

D. Avaliação dos resultados

Para uma avaliação eficaz do nosso método classificador, quatro métricas de desempenho de aprendizado de máquina foram usadas, tais como: Acurácia, Precisão, F *Measure (F1)*, e *Recall*. Os valores de cada métrica são relatados na tabela de resultados no tópico resultados e discussões. Para avaliar o desempenho do método proposto, utilizou-se a validação cruzada *K-Fold* onde definimos o valor de $K = 4$. A acurácia é a média global do acerto do modelo ao classificar as classes; A precisão traz a informação de quantas observações o

modelo classificou corretamente; A F1 ou *Measure* é a média harmônica entre o *Recall* e Precisão; E o *recall* é a métrica utilizada para indicar a relação entre as previsões positivas realizadas corretamente e todas as previsões que realmente são positivas *(True Positives e False Negatives)*. Essas métricas podem ser calculadas a partir das equações 1, 2, 3 e 4.

$$Acurácia = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

$$Precisão = \frac{TP}{TP + FP} \quad (2)$$

$$F1 = 2 * \frac{Precisão * Recall}{Precisão + Recall} \quad (3)$$

$$Recall = \frac{TP}{TP + FN} \quad (4)$$

em que:

TP = Verdadeiro Negativo
TN = Verdadeiro Positivo
FN = Falso Negativo
FP = Falso Positivo

IV. Resultados e Discussões

Para mostrar o desempenho da classificação de *ransomware* utilizando algoritmos de *machine learning*, propomos cinco métodos existentes na literatura. O primeiro método faz uma classificação binária com todas as classes do *dataset*. Para essa abordagem o melhor desempenho foi para o algoritmo regressão logística (RL) com uma acurácia de 97.69%, com desvio padrão (DV) de 0.156%, como mostrado na Tabela III. Quando comparado com [9], nossa metodologia se destaca, pois os autores alcançaram 88,2% de acurácia.

TABELA III
Classificação do método 1. Métricas em %.

Algoritmos	Acurácia ± DP	F1	Precisão	Recall
RL	97.69 ± 0.156	97.19	97.40	96.82
NN	93.39 ± 0.714	91.14	87.37	95.25
LSVM	97.21 ± 0.442	96.37	95.56	97.19
RBF SVM	95.68 ± 0.832	94.86	96.05	93.75
GP	97.67 ± 0.284	97.09	96.99	97.19
DT	92.24 ± 2.230	90.34	90.57	90.30
RF	94.07 ± 0.826	93.29	97.03	89.87
AdaBoost	95.21 ± 0.624	93.99	93.60	94.39
NB	50.59 ± 0.409	55.13	38.31	98.27
NBB	80.06 ± 0.934	75.28	75.02	75.86
QDA	95.12 ± 1.480	94.17	95.61	92.88

O segundo método faz uma classificação de família incluindo os goodwares. Para essa abordagem o melhor desempenho foi de 51.84% com o algoritmo Linear SVM (LSVM), como mostrado na Tabela IV. Quando comparado com o trabalho de [11], a metehurística PSO teve melhores resultados, alcançando uma média de 54,65% de acurácia.

O terceiro método faz uma classificação de família excluindo os goodwares. Para este experimento o algoritmo regressão logística (RL) teve o melhor desempenho, alcançando 51,47% de acurácia, conforme mostrado na Tabela V.

TABELA IV
Classificação do método 2. Métricas em %.

Algoritmos	Acurácia ± DP	F1	Precisão	Recall
RL	51.47 ± 1.71	53.14	56.16	51.47
NN	42.09 ± 1.93	42.29	44.48	42.09
LSVM	51.84 ± 1.65	52.10	53.77	51.84
RBF SVM	40.56 ± 2.48	42.89	57.03	40.56
GP	46.77 ± 2.62	47.60	49.92	46.77
DT	43.12 ± 2.74	43.28	44.61	43.12
RF	39.16 ± 1.94	46.77	69.86	39.16
AdaBoost	10.13 ± 3.12	0.08	0.09	10.13
NB	28.93 ± 7.57	19.08	27.48	28.93
NBB	48.72 ± 7.33	50.01	58.14	48.72
QDA	45.60 ± 4.17	40.84	53.02	45.60

TABELA V
Classificação do método 3. Métricas dadas em %.

Algoritmos	Acurácia ± DP	F1	Precisão	Recall
RL	51.47 ± 5.80	51.89	55.21	51.47
NN	36.46 ± 4.57	36.12	38.91	36.46
LSVM	48.70 ± 7.42	48.92	50.96	74.20
RBF SVM	39.21 ± 2.34	39.57	48.65	39.21
GP	34.17 ± 1.61	29.12	26.59	34.17
DT	40.66 ± 6.40	39.33	39.35	40.66
RF	49.57 ± 6.14	50.47	54.62	49.57
AdaBosst	16.22 ± 2.03	12.96	14.57	16.22
NB	34.13 ± 6.52	41.88	48.17	34.13
NBB	42.91 ± 4.31	41.88	48.17	42.91
QDA	41.14 ± 8.06	37.89	55.00	41.14

O quarto método faz uma classificação abordada por [12], onde uma classe de família é unida com os goodwares. Há uma comparação de uma classe por vez. Para este experimento nosso melhor resultado foi de 96,16% de acurácia com o algoritmo Linear SVM (LSVM) na família 11 de acordo com a Tabela VI. Os autores [12] tiveram uma maior eficácia neste experimento, alcançando até 98% de acurácia.

TABELA VI
Classificação do método 4. Acurácia em %.

Fam.	RL	NN	LSVM	RBF SVM	GP	DT	RF	Ada Boost	NB	NBB	QDA
F1	86.25	83.48	92.30	85.00	86.25	83.55	81.25	89.93	57.28	82.43	77.50
F2	94.04	85.78	94.17	92.26	94.04	86.64	70.23	94.11	51.78	78.30	52.97
F3	88.88	83.06	93.91	61.11	94.31	75.79	63.88	84.52	51.05	77.38	54.16
F4	77.49	82.30	82.36	50.00	50.00	66.90	57.23	79.60	68.45	79.53	50.00
F5	93.26	88.82	92.93	93.20	93.07	86.07	56.73	9012	52.97	88.39	56.73
F6	88.09	86.37	90.38	78.88	88.02	79.73	63.15	90.12	50.06	79.13	50.65
F7	91.66	87.61	95.30	77.08	94.79	81.16	69.79	85.15	52.05	83.00	61.45
F8	50.00	50.00	50.00	50.00	50.00	50.00	50.00	50.00	47.15	49.93	50.00
F9	93.75	87.10	96.13	81.94	93.68	85.84	79.16	91.26	53.43	67.36	71.52
F10	75.00	49.73	74.93	50.00	50.00	62.36	50.00	75.00	53.37	74.86	50.00
F11	89.28	78.10	96.16	66.07	80.29	78.24	60.71	87.10	56.21	80.15	50.00

O quinto método também faz uma classificação abordado por [12], em que uma família inteira é retirada, e em seguida é aplicado treinamento sobre todo o *dataset* restante e faz a validação coma familia retirada. Esse foi o único método que não foi feito crosvalidation. Quando comparada com [12], nosso resultado se destacou pois alcançou 100% de acurácia em 8 famílias, como mostra a Tabela VII, enquanto de [12] ficou em 98% de acurácia. Para esse método o autor

também utliza o termo variante. É possível observar que nesta abordagem, as famílias 8 (PGPCODER) e 10 (TeslaCrypt) obtiveram resultados de até 100% de acurácia na classificação com o algoritmo Naive Bayes (NB), o que são resultados eficientes, haja vista que estas famílias possuem 4 e 6 amostras, respectivamente.

TABELA VII
CLASSIFICAÇÃO DO MÉTODO 5. ACURÁCIA EM %.

Fam.	RL	NN	LSVM	RBF SVM	GP	DT	RF	Ada Boost	NB	NBB	QDA
F1	98.00	90.00	98.00	92.00	98.00	82.00	92.00	96.00	100.0	90.00	94.00
F2	97.19	98.13	97.19	93.45	98.13	90.65	85.04	97.19	99.06	50.46	37.38
F3	76.08	93.47	97.82	73.91	97.82	84.78	82.60	93.47	100.0	52.17	71.73
F4	72.00	52.00	72.00	92.00	72.00	72.00	84.00	92.00	100.0	36.00	60.00
F5	98.43	93.75	98.43	93.75	98.43	65.62	57.81	81.25	96.87	81.25	01.56
F6	92.78	90.72	93.81	86.59	93.81	86.59	77.31	82.47	93.81	71.13	05.15
F7	96.61	77.96	89.83	91.52	93.22	91.52	86.44	94.91	100.0	45.76	72.88
F8	100.0	100.0	100.0	100.0	100.0	50.00	25.00	75.00	100.0	75.00	50.00
F9	88.88	90.00	78.88	83.33	88.88	86.66	60.00	91.11	100.0	46.66	01.11
F10	83.33	83.33	83.33	83.33	83.33	83.33	83.33	100.0	100.0	83.33	100.0
F11	100.0	91.17	100.0	91.17	100.0	85.29	82.35	100.0	100.0	67.64	91.17

V. CONCLUSÃO

Para este trabalho propomos uma classificação de *ransomware* abordando algoritmos classificadores de machine learning, utilizando cinco métodos existentes na literatura. As classificações foram feitas com 11 algoritmos de classificação usando a biblioteca *Scikit-learn* de aprendizado de máquina de código aberto para a linguagem de programação Python. Os classificadores se mostraram eficientes alcançando resultados até de 100% de acurácia, como o Naive Bayes, superando alguns resultados existentes na literatura. Não houve mudanças nos classificadores em nenhum dos *datasets* pra treino e teste aplicados, também não foi utilizado nenhuma aplicação de técnica de balanceamento de classes. Nos próximos trabalhos iremos testar essas observações, além de comparar o PCA com outras técnicas de redução de dimensionalidade, a fim de encontrar melhores resultados nos outros métodos que não chegaram perto de 100% de acurácia.

AGRADECIMENTOS

Agradecemos ao Programa de Pós Graduação em Engenharia Elética (PPGEE) da UFPA ao apoio aos estudantes neste período delicado de pandemia.

REFERÊNCIAS

[1] L. F. Freedman, "Ransomware attacks predicted to occur every 11 seconds in 2021 with a cost of $20 billion — data privacy + cybersecurity insider," 2020. [Online]. Available: https://www.dataprivacyandsecurityinsider.com/2020/02/ransomware-attacks-predicted-to-occur-every-11-seconds-in-2021-with-a-cost-of-20-billion/

[2] H. Oz, A. Aris, A. Levi, and A. S. Uluagac, "A survey on ransomware: Evolution, taxonomy, and defense solutions," 2021. [Online]. Available: https://doi.org/10.1145/nnnnnnn.nnnnnnn

[3] G. Cusack, O. Michel, and E. Keller, "Machine learning-based detection of ransomware using sdn," in *SDN-NFVSec 2018 - Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks and Network Function Virtualization, Co-located with CODASPY 2018*, vol. 2018-January. Association for Computing Machinery, Inc, 3 2018, pp. 1–6.

[4] A. Levada, "Análise de componentes principais (pca) e análise discriminante linear (lda)," *Universidade Federal de São Carlos – Departamento de Computação, Notas da aula de Reconhecimento de Padrões*, 5 2021.

[5] D. W. Fernando, N. Komninos, and T. Chen, "A study on the evolution of ransomware detection using machine learning and deep learning techniques," *IoT*, vol. 1, pp. 551–604, 12 2020.

[6] N. Milosevic, A. Dehghantanha, and K. K. R. Choo, "Machine learning aided android malware classification," *Computers and Electrical Engineering*, vol. 61, pp. 266–274, 7 2017.

[7] J. Hwang, J. Kim, S. Lee, and K. Kim, "Two-stage ransomware detection using dynamic analysis and machine learning techniques," *Wireless Personal Communications*, vol. 112, pp. 2597–2609, 6 2020.

[8] F. Alenazi, K. Hindi, and B. AsSadhan, "Fine-tuning naïve bayes for imbalanced datasets," *International Conference on Data Science*, 7 2019.

[9] U. Adamu and I. Awan, "Ransomware prediction using supervised learning algorithms," in *Proceedings - 2019 International Conference on Future Internet of Things and Cloud, FiCloud 2019*. Institute of Electrical and Electronics Engineers Inc., 8 2019, pp. 57–63.

[10] S. I. Bae, G. B. Lee, and E. G. Im, "Ransomware detection using machine learning algorithms," vol. 32. John Wiley and Sons Ltd, 9 2019, p. e5422. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/cpe.5422 https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.5422 https://onlinelibrary.wiley.com/doi/10.1002/cpe.5422

[11] M. S. Abbasi, H. Al-Sahaf, and I. Welch, "Particle swarm optimization: A wrapper-based feature selection method for ransomware detection and classification," vol. 12104 LNCS, pp. 181–196, 4 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-43722-0_12

[12] P. Borah, D. K. Bhattacharyya, and J. K. Kalita, "Cost effective method for ransomware detection: An ensemble approach," vol. 12582 LNCS. Springer Science and Business Media Deutschland GmbH, 2021, pp. 203–219.

[13] D. Sgandurra, L. Muñoz-González, R. Mohsen, and E. C. Lupu, "Automated dynamic analysis of ransomware: Benefits, limitations and use for detection," 9 2016. [Online]. Available: http://arxiv.org/abs/1609.03020

[14] A. Kharraz, W. Robertson, D. Balzarotti, L. Bilge, and E. Kirda, "Cutting the gordian knot: A look under the hood of ransomware attacks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9148. Springer Verlag, 2015, pp. 3–24.

[15] A. Kharaz, S. Arshad, C. Mulliner, W. Robertson, E. Kirda, and A. Kharraz, *UNVEIL: A Large-Scale, Automated Approach to Detecting Ransomware*, 2016. [Online]. Available: https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/kharaz

[16] C. Security, "Fall 2019 threat of the quarter: Ryuk ransomware," 2019. [Online]. Available: https://www.cisecurity.org/white-papers/fall-2019-threat-of-the-quarter-ryuk-ransomware/

[17] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, J. M. Benítez, and F. Herrera, "A review of microarray datasets and applied feature selection methods," *Information Sciences*, vol. 282, pp. 111–135, 10 2014.