Parametrização dos Algoritmos MLP e SVM em um Computador de Baixo Desempenho: Uma Análise com a Base de Dados CICIDS2017

Pedro Ely S. Mourão[1], Flávio Henrique B. de Souza[1]

[1]Instituto de Engenharia e Tecnologia
Centro Universitário de Belo Horizonte, Belo Horizonte – MG – Brazil

pedroelymourao@gmail.com, flabasouza@yahoo.com.br

Abstract: *The Intrusion Detection System establishes some guidelines to identify attacks with different forms of action in a system . In this context, pattern recognition algorithms as MLP and SVM can assist in the detection of these events. This research presents a tuning process of MLP (Multilayer Perceptron) and SVM (Support Vector Machine) structures and evaluates their performance in a low performance computer. The tests had two different resampling methods. In 100 hours of experiments and datasets of up to 215,000 records, processing times and accuracy were compared (with a parametrization between 0 to 1), achieving at maximum values of 10,000 seconds and 0.999962 respectively, in addition to the designer's control over the structure used.*

Resumo: *O Sistema de detecção de Intrusão estabelece algumas diretrizes para identificação de ataques em um sistema, que possuem diferentes formas de ação. Nesse contexto, algoritmos de reconhecimento de padrões como MLP (Multilayer Perceptron) e SVM (Support Vector Machine) podem auxiliar na detecção deste evento. Com este foco, este artigo apresenta um processo de ajuste das estruturas MLP e SVM e avalia o seu desempenho em um computador de baixa performance. Os testes contaram com dois métodos distintos de reamostragem. Em 100 horas de experimentos, com uma base de dados de até 215 mil registros, foram comparados os tempos de processamento e a acurácia (parametrizada entre 0 a 1), chegando a valores máximos de 10 mil segundos e 0,99962 respectivamente, além do domínio do projetista sobre o dimensionamento do algoritmo utilizado.*

1. Introdução

Segundo Ardad et al. (2019), os sistemas de detecção de intrusão (IDS, do inglês *Intrusion Detection System*) têm sido utilizados nas organizações para manter a integridade e disponibilidade das informações presentes em seu sistema. O IDS não possui mecanismos para se adaptar dinamicamente às diferentes situações a ele apresentadas, pois identifica ataques comparando-os aos dados em sua base. Com isso, um ataque pode ser classificado como uma atividade normal, principalmente se a base de dados não estiver atualizada. Diversos estudos sobre essa questão têm sido abordados na literatura [Netto, 2006; Ardad et. al. 2019; Kumar e Reddy, 2019].

A base de dados CICIDS2017 é um conjunto de dados rotulados gerados através do software CICFlowMeter, que, assim como o CICIDS2017, está disponível publicamente no site do Canadian Institute for Cybersecurity [Lashkari et al., 2017]. A base de dados contém diferentes tipos de ataques para IDSs [Sharafaldin et al. 2018].

O uso de técnicas de reconhecimento de padrões como a Multilayer Perceptron (MLP) e a Support Vector Machine (SVM) para detecção de intrusões, pode auxiliar na detecção de atividades anômalas não catalogadas, ou não registradas na base de dados do IDS, devido a capacidade dessas técnicas de identificar padrões [Ardad et al., 2019]. Tais técnicas são consideradas duas das mais importantes e relevantes estruturas da literatura na atualidade [Osowski et al. 2004; Afzal et al. 2020]. Uma métrica de classificação que é utilizada para referenciar o desempenho da estrutura dessas técnicas sobre um problema de classificação é a curva ROC (do inglês, *Receiver Operating Characteristic*) e sua AUC (do inglês, *Area Under Curve*). A curva ROC é uma representação gráfica de um método para avaliação, organização e seleção de sistemas de diagnóstico e/ou predição. A AUC, que é a área abaixo da curva ROC, contribui para a avaliação de modelos em aprendizado de máquina e mineração de dados de forma normalizada [Prati et al. 2008; Ardad et. al. 2019].

Mesmo com os algoritmos MLP e SVM e a utilização da AUC para uma avaliação de desempenho, a determinação do tipo de algoritmo, bem como os parâmetros a serem utilizados para tal análise, ainda é uma atividade que demanda pesquisas [Almseidin et al., 2017]. Ainda são tomadas por relevantes as análises que possam conduzir o processo de IDS de forma mais assertiva e eficiente [Ardad et al., 2019]. Além disso, a questão do custo computacional, em relação ao tempo de processamento, motiva pesquisas para ferramentas otimizadas e eficientes [Lisboa et al., 2019], além de comparações de estruturas, como a da PassMark [Passmark, 2019]; uma vez que os recursos computacionais para pesquisas, nem sempre terão os melhores conjuntos de *hardware* (memória e processador), principalmente em instituições com orçamento limitado. Assim, a questão norteadora deste trabalho é: como é o processo de ajuste dos parâmetros dos algoritmos MLP e SVM para o problema proposto? Qual o comportamento desse processo em um computador de baixo desempenho?

Assim, o objetivo geral deste artigo é apresentar o processo de ajuste dos parâmetros das ferramentas MLP e SVM, a fim de otimizar os possíveis resultados para a base de dados utilizada, analisando o seu desempenho em um computador de baixa performance. Com relação à premissa de focar em computadores de baixo desempenho, pode-se citar o levantamento realizado pela plataforma Quero Bolsa (2020), que reuniu os dados socioeconômicos de candidatos que tentaram vaga no curso superior do ano de 2016 ao ano de 2020, através do Exame Nacional do Ensino Médio (ENEM).

Os dados foram recolhidos pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anisio Teixeira (Inep), vinculado ao Ministério da Educação (MEC). Segundo o levantamento, 37,45% dos candidatos não possuem computador em casa, sendo assim, não há como garantir o nível de desempenho dos computadores que podem ser colocados à disposição dessas pessoas dentro ou fora das universidades. Portanto, pode-se considerar que, embora os computadores atuais apresentam *hardware* de alta performance, essa tecnologia não é amplamente acessível em todas as situações ou camadas sociais, tornando pertinente tal pesquisa.

Esse trabalho se justifica pela necessidade de identificar como se dá a execução dos algoritmos MLP e SVM em computadores de baixa performance (sistema Windows 7 de 32 bits, processador Intel Pentium® Dual-Core CPU E5700 com *clock* de 3,00GHz e 2GB de memória RAM). Além da apresentação do processo de ajuste dos parâmetros sob domínio do desenvolvedor, através das ferramentas MLP e SVM.

2. Revisão Bibliográfica

2.1 MLP e SVM

O algoritmo de treinamento mais utilizado para o modelo MLP é o *Backpropagation*, que se baseia na aprendizagem por correção de erros. O algoritmo de *Backpropagation* pode ser classificado como um tipo de aprendizado supervisionado que calcula o erro gerado nos valores de saída e os propaga para a entrada de modo a ajustar os pesos e calcular os valores novamente [Guezzaz et al., 2019]. Em uma arquitetura de uma rede MLP, um vetor é apresentado aos nós das unidades de entrada x da rede e seu efeito se propaga através desta, passando por todas as camadas até chegar nas unidades de saída y que é produzido como resposta da rede [Silva, 2005]. A Figura 1 demonstra um exemplo da estrutura.

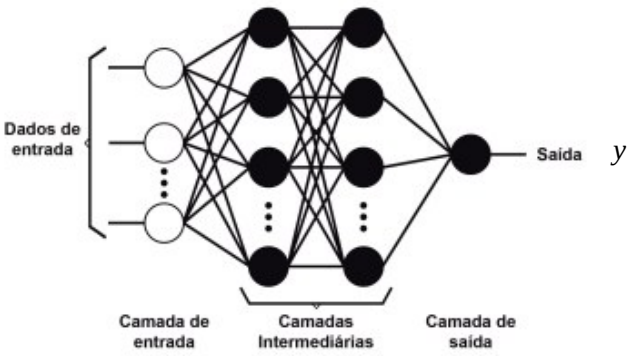

Figura 1. Exemplo de estrutura MLP. Fonte: Silva et al. (2019)

Já a SVM x za a Teoria de Aprendizado Estatístico (TAE) para construir um classificador bin a partir de um conjunto de padrões, intitulados de exemplos de treinamento, em que a classificação é conhecida. Os problemas de classificação binária (duas classes), podem ser estendidos para classificação de múltiplas classes [Chaves, 2006; Liu e Lang, 2019].

Como mostra a Figura 2, as SVMs não lineares projetam dados de entrada em um espaço de características de dimensões maiores, o que aumenta o poder computacional de máquina lineares e se torna viável devido a utilização de kernels. Padrões não linearmente separáveis podem ser transformado em um novo espaço de características em que os padrões são linearmente separáveis [Chaves, 2006; Liu e Lang, 2019].

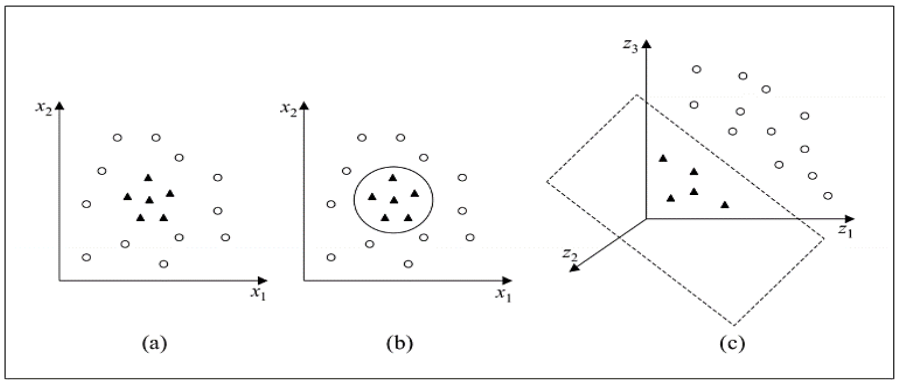

Figura 2. Separação Não Linear. (a) Conjunto de dados não linear; (b) Fronteira não linear no espaço de entradas; (c) Fronteira linear no espaço de características. Fonte: Müller et al. (2001).

2.2. Curva ROC e AUC

Para medir a acurácia das estruturas expostas, MLP e SVM, foi utilizada a curva ROC (*Receiver Operating Characteristic*), que é formada em um plano cartesiano, onde o eixo *y* representa a sensibilidade e o eixo *x*, 1 menos a especificidade (1-E), ambos em valores decimais. Para cada ponto de intersecção utilizado pelo teste, são calculadas a sensibilidade e a especificidade. Feito isso, é inserido um ponto no gráfico, como mostra a Figura 3. A união desses pontos forma a curva ROC [Lopes et al., 2014].

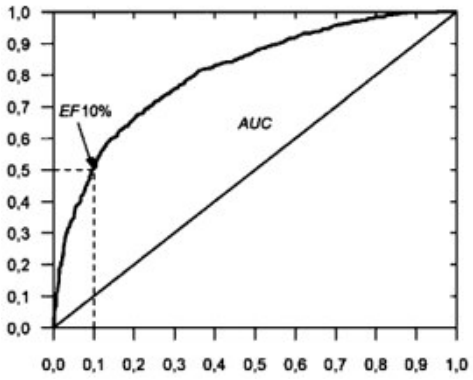

Figura 3. Exemplo de Curva ROC e AUC. Fonte: Piccirillo e Amaral (2018)

A área sob a curva ROC (AUC) representa a performance global do teste. Este parâmetro, leva em consideração todos os valores de sensibilidade e especificidade para cada valor da variável do teste. Quanto melhor o teste, mais a área sob a curva ROC se aproxima de 1 [Lopes et al., 2014]. Na matriz de confusão as linhas representam os casos reais e as colunas as previsões efetuadas pelo modelo. O número de acertos para cada caso está indicado na diagonal principal da matriz. A matriz de confusão de um classificador ideal possui todos os elementos restantes iguais a zero [Pereira et al., 2007].

2.3. Trabalhos Relacionados

Pesquisas relacionadas ao assunto têm sido apresentadas recentemente. Alguns trabalhos focam em técnicas para obter a melhor acurácia, enquanto outros buscam apresentar novas técnicas.

A Tabela 1 demonstra um comparativo de alguns dos trabalhos mais recentes da literatura, que: utilizaram a base CICIDS2017 [Aksu et al. 2018; Alrowaily et al.,2019; Ahmim et al., 2019; Lopez et al. 2019]; e experimentaram algoritmos de reconhecimento de padrões, inclusive os dois utilizados neste artigo, MLP e SVM [Aksu et al. 2018; Alrowaily et al.,2019; Ahmim et al., 2019].

Ahmim et al. (2019) apresentam um estudo para identificar o melhor algoritmo de aprendizado de máquina para classificar o tráfego de ataques específicos, utilizando MLP e SVM e a base de dados de referência.

Cabe a ressalva de que a MLP e SVM são algoritmos muito explorados na literatura, porém, poucos trabalhos demonstram o processo de ajuste de ambos (vide Tabela 1), o que é uma das questões do trabalho proposto.

Apenas um deles, Almseidin et al. (2017), mostra os parâmetros utilizados na configuração dos algoritmos de aprendizado de máquina, porém o autor utiliza a base de dados KDD, que data de 1999.

Tabela 1. Comparação com Trabalhos Relacionados

Autores	**Ano do dataset**	**Apresentação do Processo de calibração de MLP ou SVM**	**Parâmetros das técnicas de Reconhecimento de Padrões**	**Conjunto de Validação**	**Conjunto de Treinamento**	**Hardware Utilizado**
[Aksu et al.,2019]	2017	-	-	10%	90%	-
[Alrowaily et al. 2019]	2017	-	-	30%	70%	-
[Lopez et al, 2019].	2017	-	-	-	-	Intel core i7
[Almseidin et al., 2017]	1999	-	Sim	-	-	Ubuntu 13.10, Intel R, Core(TM) i5-4210U CPU @ 1.70GHz (4CPUs), 6 GB RAM
[Ardad et al., 2019]	1999	-	-	-	-	-
[Ahmim et al., 2019]	2017	-	-	50%	50%	Windows 10 - 64 bits, processor Intel(R) I5 2.7 GHz e 8 GB RAM

Autores do Presente Projeto	**2017**	**Sim**	**Sim**	**25 e 35%**	**75 e 65%**	**Windows 7 - 32 bits, processador Intel Pentium(R) Dual-Core CPU E5700 @ 3,00GHz e 2GB RAM**

3. Metodologia

Para os testes com as ferramentas MLP e SVM, será utilizado a base de dados CICIDS2017, que segundo *Sharafaldin et al.* (2018) "*abrange todos os critérios necessários com ataques atualizados comuns"*, sendo esses ataques: DoS, DDoS, Força Bruta, XSS, Injeção de SQL, Infiltração, Varredura de Porta e Botnet. Os criadores da base de dados analisaram o conjunto de dados gerado, para assim, selecionar os melhores conjuntos de recursos para detectar diferentes ataques. Também utilizaram sete algoritmos de aprendizado de máquina para avaliar o conjunto de dados gerado. Os algoritmos utilizados foram: *KNN*, *RF*, *ID3*, *Adaboost*, *MLP*, *Naive-Bayes* e *QDA* [Sharafaldin et al. 2018].

Os dados para a construção da base de dados foram capturados durante 5 dias, tendo o seu início na manhã de segunda-feira, 3 de julho de 2017, e seu término na tarde de sexta-feira, 7 de julho de 2017. Os ataques foram distribuídos por dia da semana, com exceção de segunda-feira que teve um tráfego normal de rede. Os ataques, assim como descritos na Tabela 2, foram distribuídos na manhã e tarde de terça-feira, quarta-feira, quinta-feira e sexta-feira, e os dados foram distribuídos em 8 arquivos *CSV* [Sharafaldin et al. 2018].

Tabela 2. Rótulos diários da base de dados

Dias	***Rótulos***
Segunda-feira	Benign
Terca-feira	BForce, SFTP and SSH
Quarta-feira	DoS and Hearbleed Attacks slowloris, Slowhttptest, Hulk and GoldenEye
Quinta-feira	Web Infiltration Attacks, Web BForce, XSS and Sql Inject. Infiltration Dropbox Download and Cool disk
Sexta-feira	DDoS LOIT, Botnet ARES, PortScans (sS, sT, sF, sX, sN, sP, sV, sU, sO, sA, sW, sR, sL and B)

Os dados serão analisados a fim de identificar padrões na construção da base de dados utilizada e, assim, a análise obtida sera empregada para a verificação da acurácia das técnicas a serem aplicadas.

3.1. Tratamento da Base de dados

Para execução dos experimentos, um tratamento prévio das bases foi necessário. A base de dados CICIDS2017 contém 2.830.743 de observações, divididas em 8 arquivos e

possui 79 características. Entre os 8 arquivos da base de dados foi selecionado o arquivo de número 8, que foi coletado na tarde do último dia da construção do CICIDS2017. Os registros utilizados são referentes a atividades normais de uso e ataques DDoS.

Todos valores da base de dados que não eram números reais (ou possuíam valores infinitos) foram removidos da tabela. Também foram removidos os rótulos que possuíam o mesmo valor em todas as linhas. Os rótulos são: *Bwd PSH Flags, Bwd URG Flags, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk/Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, Bwd Avg Bulk/Rate and Fwd URG Flags.*

Ao final do tratamento da base restaram um total de 70 características e 225.745 observações, 97.718 observações de atividade normal de uso e 128.025 observações de ataque DdoS, as quais foram utilizadas para realizar os treinamentos. Foram estabelecidas 3 bases de dados para treinamento, sendo elas: B1 utilizando todas as 225.745 observações, B2 com 112.872 observações e B3 com 80.000 observações.

3.2. Metodologias de Experimentos

Para realizar os experimentos de ajuste, foram selecionados alguns valores prévios. Algumas prerrogativas foram tomadas por referência, de acordo com a estrutura avaliada. A Tabela 3 as apresenta as considerações referentes ao MLP e a Tabela 4 referentes a de SVM.

Tais referências foram utilizadas com as bases previamente tratadas, para os experimentos a serem descritos.

Tabela 3. Parâmetros de Análises do MLP

Parâmetro Analisado	***Análise***	***Métricas***
Número de épocas	Quanto menos épocas, menor o tempo de processamento. Porém, um número insuficiente de tais iterações, pode ocasionar em perdas na qualidade do aprendizado.	50, 100, 500, 1000, 2000, 3000, 4000
Reamostragem: Quantidade de amostras do conjunto de teste e validação.	A quantidade de amostras para treinar tais estruturas interfere na quantidade de ocasiões para as quais a MLP está preparada para analisar ou referenciar.	Teste (75%) Validação (25%) Teste (65%) Validação (35%)
Quantidade de Neurônios da Camada Oculta	Quanto maior a quantidade de neurônios na camada oculta, mais complexa a estrutura resultante pode se apresentar, porém este número pode influenciar diretamente na capacidade de aprendizado da MLP. Nem sempre uma grande quantidade de neurônios garante o melhor aprendizado na MLP.	3, 5, 7, 10
Tipos de MLPs	Apesar de uma estrutura base de referência, técnicas foram integradas às estruturas MLP para otimizar o processo de aprendizado. Suas melhorias buscam trazer maior eficiência as funções de ativação e de busca do melhor valor dos pesos das conexões entre os neurônios.	*Backpropagation Standard* (BS), *Backpropagation Momentum* (BM), *Resilient propagation* (Rp), *Backpropagation*

		with Weight Decay (BWD), *Quick propagation* (Qp).

Tabela 4. Parâmetros de Análises SVM

Parâmetro Analisado	***Análise***	***Métricas***
Funções de Kernel	De acordo com a função de kernel escolhida para avaliação da base de dados, a SVM pode ter uma maior acurácia. Assim, quatro tipos de funções de kernel foram elencadas na literatura para exploração daquela que mostrar maior acurácia para a base analisada.	Radial Basis, Polinomial, Tangente Hiperbólica, Linear
Reamostragem: ***O valor de k da técnica k-fold cross validation***	O processo de reamostragem divide a base de dados em k pastas, onde é feito o processo de teste e validação em separado com cada uma das k pastas, depois elas são comparadas entre si. Porém o processo de separação e avaliação demanda processamento, dependendo da quantidade de k. Assim, foi proposta uma busca do melhor valor de AUC, de acordo com o k que realmente é necessário.	k = 1, 2, 3, 7
Valor C de Violação de Restrições	Durante o processo de separação, um grau de violação das avaliações das funções de kernel pode permitir uma avaliação de melhor acurácia. Porém essa permissão de violação deve ser ajustada, pois pode comprometer a qualidade do processo de separação.	C = 1, 2, 3, 7

4. Resultados

Durante os experimentos, foram realizados: o tratamento da base de dados, treinamento com os diferentes tipos de algoritmos, uma análise dos experimentos, e por fim, um uma análise comparativa entre a estrutura utilizada neste artigo e as que foram usadas na literatura.

4.1. Análise de Desempenho por MLP

Para analisar o desempenho das técnicas de reconhecimento de padrões selecionadas, primeiramente definiu-se os experimentos a serem realizados para cada tipo de técnica, neste caso uma *MLP* e uma *SVM*. Para cada uma delas foram estabelecidas diferentes regras de experimento. Após obter os resultados, realizou-se uma análise de cada uma delas relacionando-as com o tempo de execução e a AUC de cada treinamento.

4.1.1. Análise da MLP com a Quantidade de Neurônios Fixa

Tomado como referência a quantidade de 10 neurônios na camada oculta, foi feita a variação dos demais parâmetros da Tabela 3, com a base B1 (totalizando 70 experimentos). O treinamento das primeiras MLPs objetiva aferir a influência sobre o tempo e a AUC, da variação de quantidade de épocas e porcentagem de validação para este experimento.

Os gráficos das Figuras 4 a 8 mostram o tempo em segundos de treinamento (eixo à esquerda do gráfico) para todas as épocas com 25% e 35% de validação. Estes mesmos gráficos contém um eixo secundário (à direita) com o valor de AUC para cada

época com 25% e 35% de validação. A maior AUC obtida foi de 0,99962 no treinamento com 2.000 épocas, 35% de validação e com a MLP Rp. A menor foi de 0,98450 no treinamento com 50 épocas, 35% de validação e na MLP Qp.

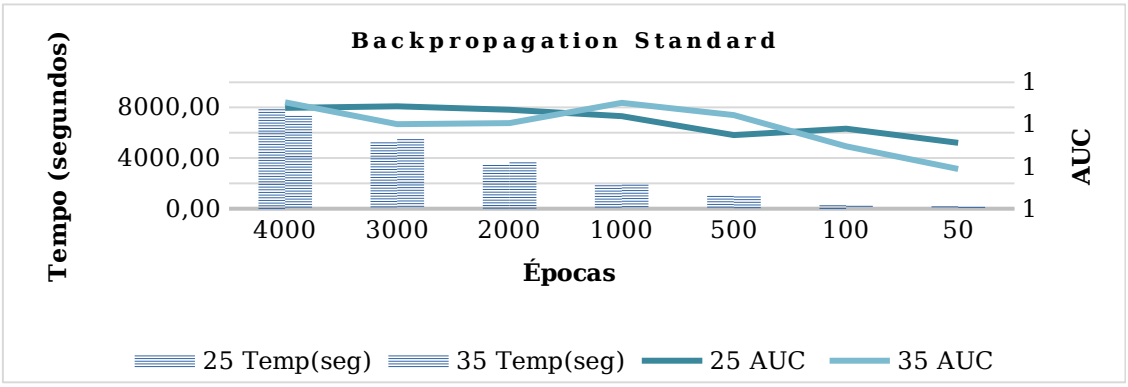

Figura 4. Variação de Tempo e AUC por Épocas do Backpropagation

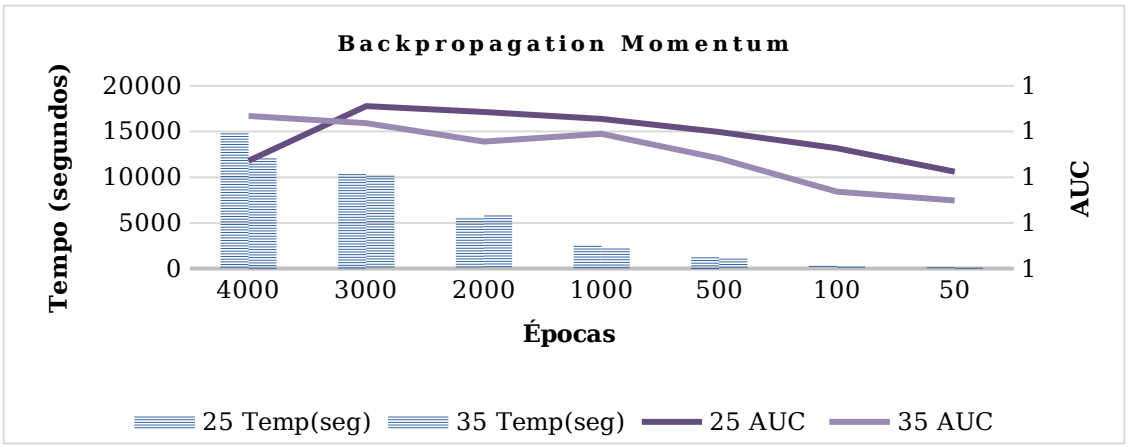

Figura 5. Variação de Tempo e AUC por Épocas do Backpropagation Momentum

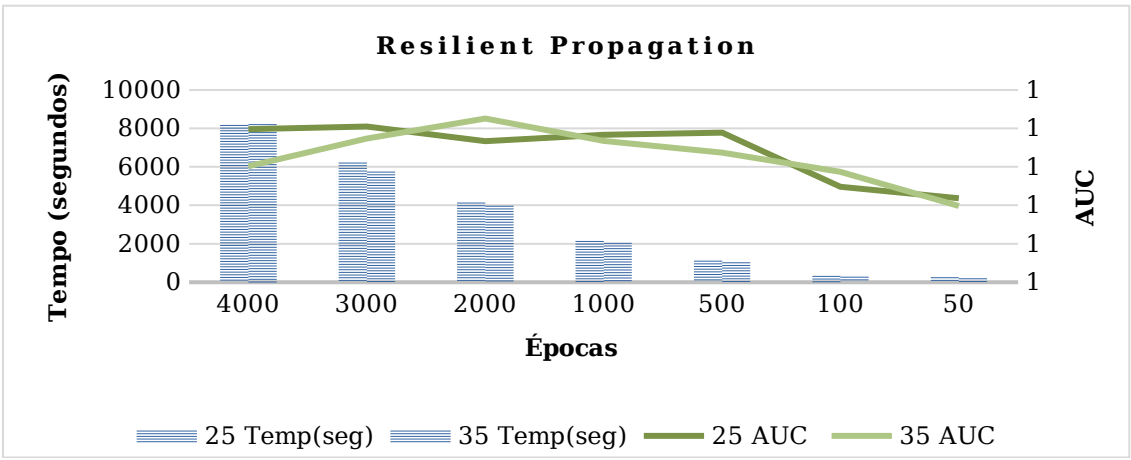

Figura 6. Variação de Tempo e AUC por Épocas do Resilient Propagation

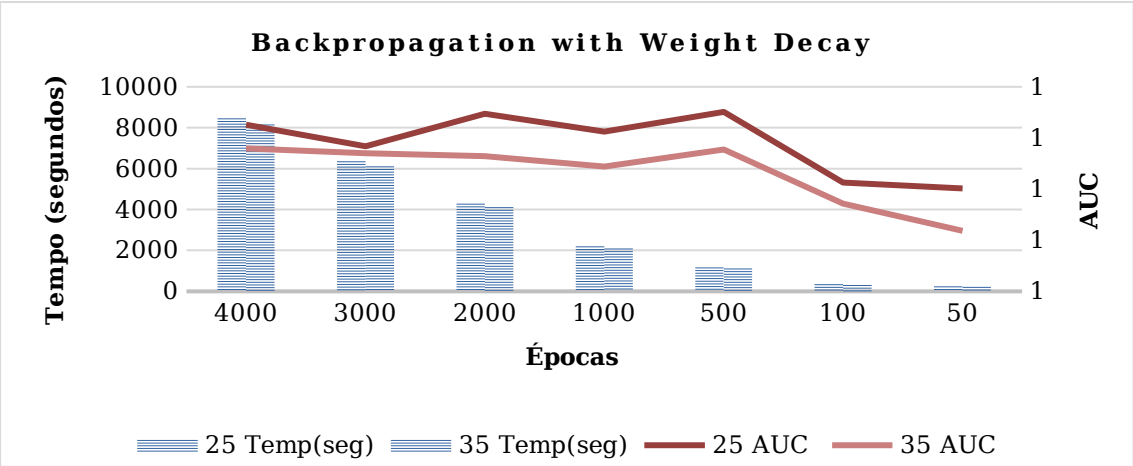

Figura 7. Variação de Tempo e AUC por Épocas do Backpropagation with Weight Decay

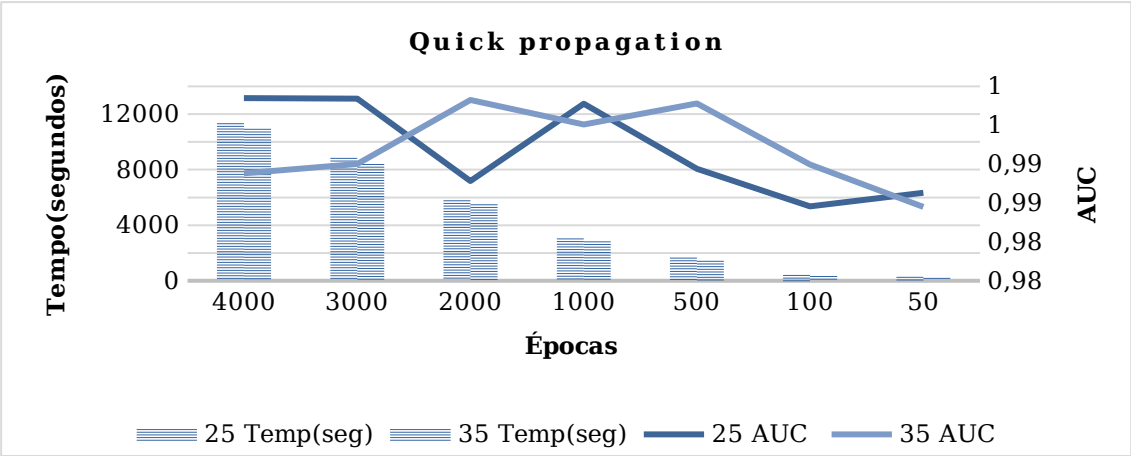

Figura 8. Variação de Tempo e AUC por Épocas do Quick propagation

O treinamento que obteve o maior custo de tempo foi o do tipo BM, com 15.014,12 segundos, com as medidas de 4.000 épocas e 25% de validação. O de menor custo foi também da BM com 293,72 segundos, com as medidas de 50 épocas e 35% de validação. O BM obteve a melhor média de AUC com 25% e 35% de validação, sendo seu valor médio de 0,99936 e 0,99921 respectivamente. Já a Qp teve a pior média para 25% e 35% de validação, com os valores de 0,99183 e 0,99206.

O tipo de MLP que obteve a melhor média geral de AUC foi a BM com 0,99929 e o tipo com a pior média foi a Qp com 0,99194. Apesar de o tipo BM obter a melhor média de AUC e o maior custo de tempo, a variação de sua melhor AUC para a pior foi de apenas 0,00082, enquanto a variação de tempo entre esses valores selecionados de treinamento foi de 10.244,09 segundos. Ou seja, realizou-se um treinamento com 2 horas e 50 minutos a mais que a pior AUC para obter apenas 0,00082 a mais de aproveitamento.

O tipo Qp obteve a maior variação entre sua pior e melhor AUC, sendo a diferença de 0,01400, e apresentou a maior variação de tempo, sendo de 11.139,42 segundos. Apesar de ter a maior variação entre AUCs o custo para isso é consideravelmente grande, correspondendo a aproximadamente 3 horas.

A MLP BS registrou a menor variação entre a pior e melhor AUC com o valor de 0,00063, e o custo de tempo dessa variação foi de 7.195,91 segundos, correspondendo a um gasto de aproximadamente 2 horas para uma variação pequena.

O algoritmo BWD registrou a menor variação de tempo entre sua melhor e pior AUC. A variação entre as AUCs foi de 0,00093 e o custo de tempo foi de 960,49 segundos, ou seja, uma taxa de 16 minutos entre a pior e melhor AUC. O tipo Rp contabilizou a maior AUC, e sua variação entre seu maior e menor valor foi de 0,00113, sendo assim a segunda maior variação. O custo de tempo para isso foi de 3777,81 segundos.

4.1.2 Análise da MLP com a Variação do Número de Neurônios

Considerando que nos testes realizados na seção anterior, a variação entre a pior e a melhor AUC foi de 0,015127 a um custo de tempo de 3.739,31 segundos, optou-se nesta seção por realizar os experimentos com a variação de neurônios utilizando a configuração do treinamento com a menor média de custo de tempo, já que, ainda assim, sua média de AUC não variou muito em comparação com os maiores valores obtidos.

A configuração a ser utilizada é de 50 épocas, 35% de validação e varia de 2 a 10 neurônios, número utilizado nos experimentos anteriores. O objetivo desta seção é aferir se a variação de neurônios tem impacto significativo na AUC. A Figura 9 mostra a variação de tempo e AUC para diferentes números de neurônios para cada tipo de MLP.

A média de AUC entre os diferentes números de neurônios para cada tipo de MLP, com exceção da Qp, se mantiveram acima de 0,998. E o número de neurônios que obteve a maior média de AUC foi o de 10 neurônios, seguido por 5, 3, 7 e 2 neurônios.

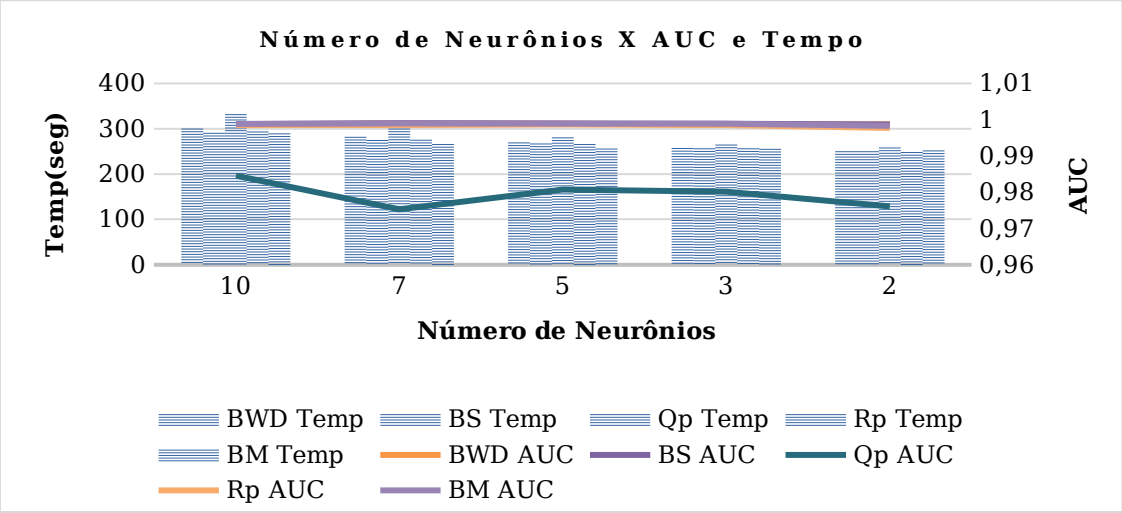

Figura 9. Variação de Neurônios por Tempo e AUC das MLP's

4.2 Análises com SVM

Os experimentos contaram com os parâmetros descritos na Tabela 4. O primeiro teste realizado foi com o kernel *Radial Basis* com os valores de C (Custo de violação de restrições) e *cross* (número *k* do *k*-fold) iguais a 7. Essa experimentação mostrou um

tempo muito elevado, por isso foi estabelecido um tempo limite de 12 horas, que, caso atingido, interromperia o treinamento da rede. Esse primeiro teste excedeu o tempo determinado e, portanto, foi realizado um segundo teste alterando os valores dos parâmetros C e *cross para* 1. Entretanto, esse teste também excedeu o tempo estabelecido.

No terceiro e quarto teste optou-se por alterar o número de observações. Com base nisso, utilizou-se a base B2 e B3, porém, o custo de tempo permaneceu alto e o tempo limite ainda assim foi excedido com as duas variações.

O segundo kernel escolhido para teste foi o Polinomial, com os valores de *C* e *cross* iguais a 1, utilizando a base B1. O teste registrou queda considerável no custo de tempo, e finalizou a execução após 6 horas 58 minutos e 48 segundos. Entretanto, pelos valores estabelecidos para realizar todos os testes com a *SVM*, que são: C igual a 1, 2 e 3; e *cross* igual a 1, 2 e 3, realizou-se o teste com a base B2 e B3. Assim, houve uma queda significativa do tempo para 40 minutos, com a base B2, e 24 minutos, com a base B3. Dessa forma, optou-se por utilizar a base B3 para realizar o restante dos testes.

Na Figura 10, os testes realizados com o kernel Polinomial mostraram que os diferentes valores de *cross* não alteraram o valor da AUC, mas aumentaram consideravelmente o custo de tempo. Já os valores de C, iguais a 1, 2 e 3 mostrados na Figura 11 como: C1, C2 e C3 influenciaram no valor da AUC.

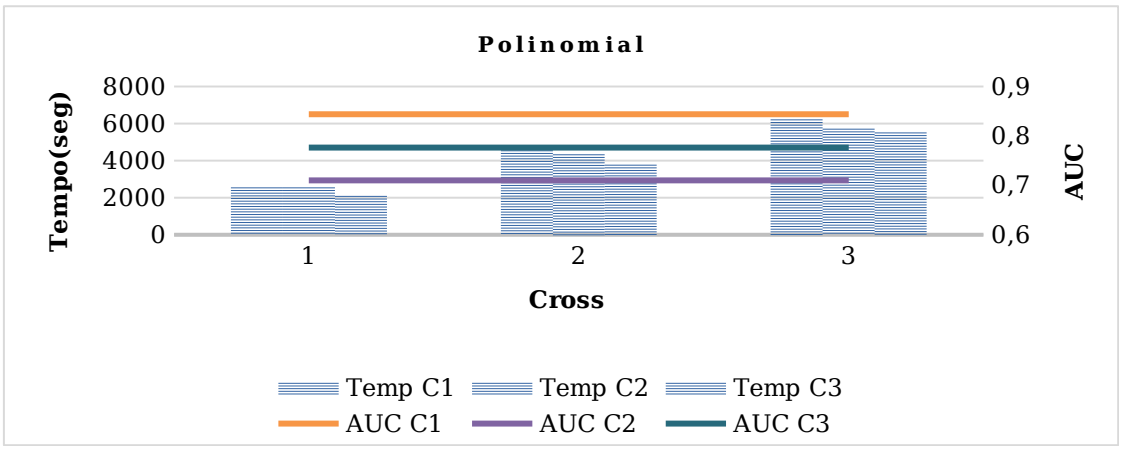

Figura 10. Variação de tempo e AUC para C e *cross* do kernel polinomial

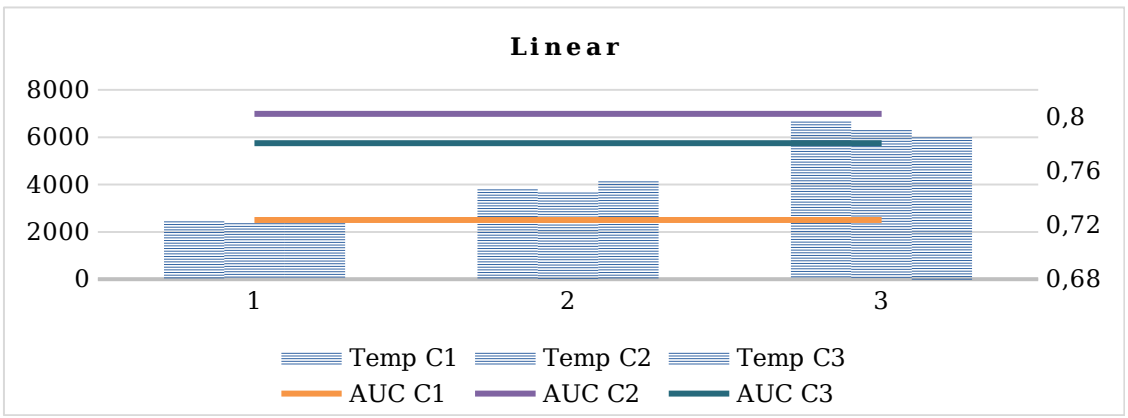

Figura 11. Variação de tempo e AUC para C e *cross* do kernel linear

O valor de C igual a 1 obteve a maior AUC, sendo de 0,84382, seguido por C igual a 3 com um valor de 0,77640 e com o pior valor de AUC correspondendo a C igual a 2, que resultou em 0,70994.

Realizou-se os testes com os mesmos parametros para o kernel Tangente Hiperbólica variando o valor de *cross* de 1 a 3, mantedo por padrão o kernel com o valor de C fixo em 3. Mesmo variando o valor de *cross* a AUC permaneceu constante no valor de 0.77640. A Figura 11 mostra a variação de tempo e AUC com o kernel Linear. Os parâmetros utilizados foram os mesmo dos kernels anteriores. Assim como nos testes realizados com os outros kernels, não houve variação com os diferentes valores de *cross*, de modo que os valores só mudaram de acordo com a variação de C.

A maior AUC foi obtida para o valor de C igual a 2, correspondendo a 0,8023149, seguido por C igual a 3, no valor de 0,78053, e, em seguida, paraC igual 1 com uma AUC de 0,72389.

4.3. Análise Comparativa

A fim de realizar uma análise comparativa entre as técnicas MLP e SVM, foi utilizado a base B3 e divididos a mesma quantidade de conjunto de treinamento e validação. As configurações utilizadas, foram aquelas que obtiveram os maiores valores de AUC das duas estruturas.

Para a MLP foi utilizado o tipo *Resilient propagation* (com 2.000 iterações e 10 neurônios). Já para a SVM foi utilizada o kernel Polinomial (com C igual a 1 e *cross validation* igual a 1).

A maior AUC foi obtida com a *MLP*, chegando a 0,99898, com um custo de 1.507 segundos. Já a *SVM* obteve uma AUC de 0,84382 a um custo de 2.656 segundos.

Para breve avalição comparativa, ressalta-se que os testes aqui realizados foram executados em uma máquina de baixo desempenho. Além da baixa quantidade de memória RAM, o processador também possui uma pontuação baixa segundo a métrica utilizada pela *PassMark Software*.

Assim, foi realizada uma comparação com referência da *PassMark (2019)*. O processador está localizado na posição 1.780. Segundo a classificação de referência, tal processador está 14.226 pontos atrás do primeiro colocado. Se comparado a um dos processadores utilizados na literatura apresentada na seção 2.3, que ocupa a 601º posição na classificação, o processador possui 5.020 pontos de diferença.

5. Conclusão

Após analisar os resultados obtidos utilizando técnicas de reconhecimento de padrões, pode-se concluir que a sua utilização para realizar tal tarefa com uma MLP possui resultado satisfatório, considerando que a menor AUC obtida entre todos os 95 testes e os 5 tipos de MLP, foi de 0,97523 com a MLP QP. Em contrapartida, o maior valor de AUC da SVM foi de 0,84382, que corresponde a um valor bem abaixo do obtido na MLP.

O tempo total de todos os testes foi de aproximadamente 100 horas. Com isso, observou-se que nesse experimento uma quantidade maior de épocas tem um custo de tempo consideravelmente maior para os testes em MLP, como também o número de

cross e observações para os testes em SVM. Entretanto, essas variações não garantem, para essa base, uma melhor AUC. Dificilmente se encontra na literatura demonstração de ajuste e de configuração das técnicas de reconhecimento de padrões utilizando essa base de dados, apresentando no caso da MLP, o número de neurônios, épocas e taxa de aprendizado. Há ferramentas que são voltadas para abstração e identificação dos melhores valores de parâmetros, porém o processo de utilização não fica no domínio do desenvolvedor.

Por fim, é notória a diferença de porte entre a estrutura utilizada no presente artigo e as que são apresentadas pela literatura, no que diz respeito ao ajuste dos parâmetros das ferramentas selecionadas, o que representa uma relevante análise para a extensão de soluções, para os diversos tipos de ambientes e limitações, que uma solução de algoritmo de reconhecimento de padrões pode enfrentar. Estudos futuros podem ser realizados, como a variação da configuração do hardware de teste com as mesmas metodologias, além de explorações com outras ferramentas, como o KNN, Naive Bayes e Regressão logística.

Referências Bibliográficas

Afzal, M., Park, B. J., Hussain, M., Lee, S. (2020). “Deep Learning Based Biomedical Literature Classification Using Criteria of Scientific Rigor”. *Electronics*, v. 9, n. 8, p. 1253.

Ahmim, A.; Maglaras, L.; Ferrag, M. A.; Derdour, M.; Janicke, H. (2019) “A novel hierarchical intrusion detection system based on decision tree and rules-based models”. In: *15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE. p. 228-233.

Almseidin, M.; Alzubi, M.; Kovacs, S.; Alkasassbeh, M. (2017) “Evaluation of machine learning algorithms for intrusion detection system”. In: *IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY)*. IEEE. p. 000277-000282.

Alrowaily, M.; Alenezi, F.; Li, Z. (2019) “Effectiveness of machine learning based intrusion detection systems”. In: *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*. Springer, Cham. p. 277-288.

Ardad, M.V.V.; Dhar, M.A.; Nema, S. (2019) “Machine Learning Processing for Intrusion Detection”, In *International Research Journal of Engineering and Technology*. p. 4707-4712. v. 6. e-ISSN: 2395-0056.

Aksu, D.; Ustebay, S.; Aydin, M. A.; Atmaca, T. (2018)“Intrusion detection with comparative analysis of supervised learning techniques and fisher score feature selection algorithm”. In: *International Symposium on Computer and Information Sciences*. Springer, Cham. p. 141-149.

Chaves S, A. C. F. (2006) “Extração de regras fuzzy para máquinas de vetores suporte (SVM) para classificação em múltiplas classes”. Tese (Doutorado em Engenharia Elétrica) - Pontifícia Universidade Católica do Rio de Janeiro, Rio de Janeiro.

Guezzaz, A.; Asimi, A.; Asimi, Y.; Tbatou, Z.; Sadqi, Y. (2019) “A Global Intrusion Detection System using PcapSockS Sniffer and Multilayer Perceptron Classifier” In *IJ Network Security*, v. 21, n. 3, p. 438-450.

Kumar, E. V.; Reddy, B. I. (2019) “A Review on Application of Data Mining Techniques for Intrusion Detection”, In *International Research Journal of Engineering and Technology*. Páginas 1457-1460. v. 6. e-ISSN: 2395-0056.

Lashkari, A. H.; Draper-Gil, G.; Mamun, M. S. I.; Ghorbani, A. A.(2017) “Characterization of Tor Traffic using Time based Features”. In: ICISSP. p. 253-262.

Lisboa, A. C.; De Souza, F. H. B.; Ribeiro, C. M.; Maia, C. A.; Saldanha, R. R.; Castro, F. L.; Vieira, D. A. (2019) “On Modelling and Simulating Open Pit Mine Through Stochastic Timed Petri Nets”. In *IEEE Access*, v. 7, p. 112821-112835, 2019.

Liu, Hongyu; Lang, Bo. Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey. *Applied Sciences*, v. 9, n. 20, p. 4396, 2019.

Lopes, B., Ramos, I. C. D. O., Ribeiro, G.; Correa, R., Valbon, B. F., Luz A. C., Salomão, M., Lyra, J. M., Junior, R. A.(2014) “Bioestatísticas: conceitos fundamentais e aplicações práticas”. *Revista brasileira oftalmologia*. vol.73, n.1, pp.16-22. ISSN 0034-7280.

Lopez, A. D.; Mohan, A. P.; Nair, S. (2019) “Network Traffic Behavioral Analytics for Detection of DDoS Attacks”. In *SMU Data Science Review*, v. 2, n. 1, p. 14.

Müller, K. R.; Mika, S.; Ratsch, G.; Tsuda, K.; Scholkopf, B (2001) “An introduction to kernel-based learning algorithms”. In *IEEE Transactions on Neural Networks*, v. 12, n.2, pp. 181–201.

Netto, R. S. (2006) “Detecção de Intrusão Utilizando Redes Neurais Artificiais no Reconhecimento de Padrões de Ataque”. Universidade Federal de Itajubá, Itajubá.

Osowski, S., Siwek, K., Markiewicz, T. (2004). “MLP and SVM networks-a comparative study”. In *Proceedings of the 6th Nordic Signal Processing Symposium* NORSIG 2004. pp. 37-40. IEEE.

Passmark Software. (s.d) “Intel Pentium E5700 @ 3.00GHz”. Disponível em: <https://www.cpubenchmark.net/cpu.php?cpu=Intel+Pentium+E5700+ %40+3.00GHz&id=1101>. Acesso em: 20 de Maio de 2019.

Pereira, J. M., Dominguez, M. Á. C., Ocejo, J. L. S. (2007) “Modelos de previsão do fracasso empresarial: aspectos a considerar”. Tékhne-Revista de Estudos Politécnicos, n. 7, p. 111-148.

Prati, R. C.; Batista, G. E. A. P. A.; Monard, M. C.(2008) “Curvas ROC para avaliação de classificadores”. *IEEE Latin America Transactions*.

Piccirillo, E.; Amaral, A. T. D. (2018) “Busca virtual de compostos bioativos: conceitos e aplicações”. *Química Nova*, v. 41, n. 6, p. 662-677.

Quero Educação. Quero Bolsa (2020). “Um terço dos estudantes de ensino superior não têm a estrutura mínima para o ensino a distância”. Disponível em: < https://querobolsa.com.br/revista/um-terco-dos-estudantes-de-ensino-superior-nao-tem-a-estrutura-minima-para-o-ensino-a-distancia>. Acesso em: 10 de Agosto de 2020.

Sharafaldin, I.; Lashkari, A. H.; Ghorbani, A. A. (2018) “Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization”. In: ICISSP. 2018. p. 108-116.

Silva, L. M. (2005) “Uma aplicação de Árvores de Decisão, Redes Neurais e KNN para a Identificação de Modelos ARMA não Sazonais e Sazonais”. Tese de Doutorado. Pontifícia Universidade Católica do Rio de Janeiro. Rio de Janeiro.

Silva, R. M.; Leal, M.R.R.; Lima, F.M. (2019) “Predico do Câncer de Mama com Aplicação de Modelos de Inteligência Computacional” TEMA (São Carlos), São Carlos , v. 20, n. 2, p. 229-240.