UNIVERSIDADE FEDERAL DO AMAZONAS
INSTITUTO DE COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

CÉSAR HENRIQUE GOERSCH ANDRADE

AUTENTICAÇÃO CONTÍNUA DE USUÁRIOS UTILIZANDO CONTADORES DE DESEMPENHO DO SISTEMA OPERACIONAL

Manaus
2021

CÉSAR HENRIQUE GOERSCH ANDRADE

AUTENTICAÇÃO CONTÍNUA DE USUÁRIOS UTILIZANDO CONTADORES DE DESEMPENHO DO SISTEMA OPERACIONAL

Dissertação apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas como requisito para a obtenção do grau de Mestre em informática.

Orientador: Prof. Dr. Eduardo James Pereira Souto

Manaus

2021

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

A553a

Andrade, César Henrique Goersch
Autenticação contínua de usuários utilizando contadores de desempenho do sistema operacional / César Henrique Goersch Andrade . 2021
88 f.: il. color; 31 cm.

Orientador: Eduardo James Pereira Souto
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Autenticação Contínua. 2. Biometria Comportamental. 3. Contadores de Desempenho. 4. Nível de Confiança. I. Souto, Eduardo James Pereira. II. Universidade Federal do Amazonas III. Título

FOLHA DE APROVAÇÃO

"AUTENTICAÇÃO CONTÍNUA DE USUÁRIOS UTILIZANDO CONTADORES DE DESEMPENHO DO SISTEMA OPERACIONAL"

CÉSAR HENRIQUE GOERSCH ANDRADE

Dissertação de Mestrado defendida e aprovada pela banca examinadora constituída pelos Professores:

Prof. Eduardo James Pereira Souto - PRESIDENTE

Prof. Eduardo Luzeiro Feitosa - MEMBRO INTERNO

Dr. Thiago de Souza Rocha - MEMBRO EXTERNO

Manaus, 18 de Março de 2021

A minha mãe Elza,

dedico.

AGRADECIMENTOS

Agradeço à minha família pela compreensão pelos muitos períodos de ausência, mesmo que fisicamente estivessem ao seu lado; pela paciência que tiveram em revisar textos, mesmo que não os tenham compreendido na sua plenitude; pelo incentivo nos momentos de desânimos, pois me ajudaram a ganhar energia para continuar; pelas críticas e reclamações pelo tempo não partilhado, pois me fizeram perceber o quanto a presença de vocês em minha vida é valiosa e que não há projeto de vida sem a família. Sem o apoio de vocês eu não teria chegado até aqui. E agradeço especialmente minha esposa Márcia, por me apoiar nos momentos mais difíceis e acreditar na força da nossa união, do nosso amor.

Agradeço ao meu orientador professor Dr. Eduardo Souto pelo incentivo e apoio que foram primordiais para a conclusão dessa jornada, seu companheirismo incansável que mesmo entre reuniões sem fim arranjava um tempo para ouvir e aconselhar. Obrigado professor Souto pelos elogios que me levaram a ganhar confiança e me fazer acreditar no meu potencial e pelas críticas que me reconduziram ao caminho certo e assim me trouxeram até aqui.

Agradeço ao professor Dr. Juan Colonna por sua contribuição em um momento chave deste projeto e aos amigos dos laboratórios da UFAM que me fizeram conhecer o verdadeiro significado da palavra companheirismo e suas contribuições durante o projeto, e por compartilharem a vida acadêmica da forma mais agradável. Agradeço aos amigos da Secretaria de Estado da Fazenda pelo apoio em todas as fases de meu projeto e que foi essencial para que pudesse chegar ao final.

"Para tudo há um tempo, para cada coisa há um momento debaixo do céu: tempo de nascer, e tempo de morrer; tempo de plantar, e tempo de arrancar o que se plantou; Tempo de matar, e tempo de curar; tempo de derrubar, e tempo de edificar,..."

(Eclesiastes, 3)

RESUMO

Os computadores pessoais e coorporativos predominantemente utilizam credenciais de contas (e.g. login e senha) como método de autenticação, também conhecidos como métodos estáticos. Um problema com esta abordagem é que o usuário pode deixar o computador sem sair da sessão ou bloquear seu acesso, possibilitando a um intruso acessar os recursos disponíveis. Por essa razão, pesquisas recentes têm direcionado seus esforços em soluções de autenticação contínua baseada em modelos comportamentais dos usuários. A maioria das abordagens emprega modelos de autenticação construídos a partir de informações extraídas das interações dos usuários com os dispositivos como, por exemplo, a partir de movimentos do mouse, dinâmica na digitação de textos ou reconhecimento de fala. Diferentemente das abordagens existentes, este trabalho propõe a utilização de informações estatísticas relacionadas ao uso de hardware e software obtidos a partir dos contadores de desempenho dos sistemas operacionais para gerar modelos de autenticação. A ideia é usar as informações relacionadas ao uso dos recursos de um computador pelo usuário ao longo do tempo como o uso de memória, processador, rede, armazenamento e aplicações, para criar um perfil que possa ser usado para autenticar o usuário. A vantagem do uso destes atributos é que eles podem ser coletados de forma transparente, sem interferir na atividade do usuário. Além disso, os principais sistemas operacionais (e.g. Linux e Windows) já disponibilizam coletores nativos, não requerendo o desenvolvimento de softwares de coleta específicos. Para gerar os modelos de autenticação, nós empregamos uma arquitetura de rede profunda híbrida, composta por camadas de convolução e por camadas de recorrência. As camadas de convolução realizam a extração automática de características (neste caso, correlações entre dados dos contadores de desempenho) e as camadas de recorrência são utilizadas para capturar características temporais dos dados processado pelas camadas convolucionais. Além disso, este trabalho emprega um modelo de confiança que evita o bloqueio de usuários genuínos e impede que um impostor fique muito tempo agindo sem ser detectado. Os resultados obtidos em três cenários de avaliação mostram que o método proposto consegue detectar 100% dos usuários impostores em até 15 segundos. Os resultados comprovam a viabilidade do uso de contadores de desempenho na definição de modelos de autenticação contínua.

Keywords: Autenticação Contínua, Biometria Comportamental, Contadores de Desempenho, Nível de Confiança.

ABSTRACT

The personal and corporative computers predominantly use accounts credentials (for example: login and password) as an authentication method, also known as statistic methods. A problem with this approach is that the user can leave the computer without logout or lock the access and allowing an intruder to access the available resources. Therefore, recently searches are directing its efforts into continuing authentication solutions, based on the user's behavioral models. Most of the approaches uses authentication models built from information that are extracted from user's interactions with the devices, such as, mouse movements or dynamic text typing or speech recognition. Different form the existing approaches, this work has the propose to make use of static information related to the use of the hardware and software, which are obtained from the performance's counter from the operational systems to generate authentication models. The idea is to take the information related to the usage of computers sources by the user over time as the use of memory, processor, network, storage and application, to create a profile that can be used to authenticate the user. The advantage to use those attributes is that they can be collected in a transparent way without interfering on the user's activity. Be sides, the main operational systems (for example, Linux and Windows) have already made available native collectors, not requiring the development of specific collection software. To generate de automation models, we used a hybrid deep network architecture, composed by convolution layers and by recurrence layer. The convolution layers perform the automatic extraction of the characteristics (in this case, the correlation between the data from the performance counter) and the recurrence layer are used to computed temporal characteristic from the data processed by the convolution layers. Furthermore, this work employs a trust model that avoids blocking genuine users and prevents an imposter from spending too much time undetected. The results obtained in three evaluation scenarios show that the proposed method can detect 100% of imposter users in up to 15 seconds. These results prove the feasibility of using performance counters in the definition of continuous authentication models.

Keywords: Continuos Authentication, Behavioral Biometrics, Performance Counters, Trust Level.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Sistemas computacionais empregam uma abordagem de autenticação estática (*static authentication* - SA) baseada em credenciais de contas (e.g., logins e senhas) como o único meio de verificação da autenticidade do usuário. Em geral, esse processo de autenticação ocorre somente na entrada do sistema pelo usuário. Um problema com esta abordagem é que o usuário pode deixar o computador sem sair da sessão ou bloquear seu acesso, possibilitando a um intruso acessar os recursos disponíveis [Ayeswarya et al. 2019].

Esse acesso não autorizado pode ocorrer de diferentes maneiras. Por exemplo, a usuária Alice faz login em seu computador usando suas credenciais (usuário e senha). Por alguma razão, Alice sai de perto do seu computador sem bloquear o sistema. Outro usuário, Bob, este denominado de impostor, passa a utilizar os recursos liberados pelo login de Alice de forma indevida. Alice também pode ter compartilhado a senha com Bob, ou Bob pode ter obtido a senha de Alice por meio de um ataque.

Neste cenário, os métodos de autenticação contínua (*Continuous Authentication* – CA) que recorrentemente avaliam a autenticidade do usuário podem ser utilizados para mitigar as limitações apresentadas pelos métodos estáticos de autenticação. Na literatura já existem diferentes mecanismos de autenticação que fornecem autenticação contínua para o usuário usando biometria fisiológica e comportamental [Oak et al. 2017] [Ayeswarya et al. 2019] [Ouch et al. 2017]. Os métodos que empregam biométrica fisiológica autenticam o usuário usando atributos pessoais tais como impressão digital, íris, retina e reconhecimento facial [Akash, S e Arya 2017]. Por outro lado, os métodos de autenticação baseado no comportamento avaliam as interações dos usuários com os dispositivos para extrair padrões comportamentais como, por exemplo, a partir de movimentos do mouse, dinâmica na digitação de textos ou reconhecimento de fala [Neja et al. 2018].

Uma desvantagem do uso da biometria fisiológica é a necessidade de hardware

para executar a coleta de dados biométricos, acrescentando custo e outra camada de complexidade para o processo de login do usuário [Bailey, Okolica e Peterson 2014]. Além disso, na autenticação contínua, o usuário precisa interromper suas atividades constantemente para realizar o processo de autenticação. Por esta razão, muitos métodos de autenticação têm adotado a biometria comportamental, pois os atributos coletados podem ser obtidos silenciosamente, de forma transparente, sem atrapalhar o usuário genuíno e sem alertar o impostor que está sob avaliação [Mondal e Bours 2016]. Nos estudos de biometria comportamental, identificamos na literatura o emprego de atributos que são relacionadas às ações voluntárias dos usuários, tais como sequências de cliques de mouse [Mondal e Bours 2015], dinâmica de digitação [Bours e Barghouthi 2009] ou um sistema de múltiplos atributos que empregam movimentos de mouse e dinâmica de digitação [Fridman et al. 2015]. Entretanto, estas abordagens necessitam de softwares coletores especializados.

Outras abordagens avaliam atributos que podem não estar associados diretamente às ações diretas e voluntárias dos usuários, mas sofrem influências destas ações e estão relacionadas a atividades dos sistemas operacionais, tais como contadores de desempenho do sistema operacional [Malatras, A. et al., 2017], *system calls* [Song, et al., 2013], e uso de memória, processador, tráfego de rede e logs [Chen et al. 2016]. Estes atributos também podem ser coletados de forma transparente, sem interferir na atividade do usuário, e além disso, os principais sistemas operacionais (e.g. Linux e Windows) já disponibilizam coletores nativos, não requerendo o desenvolvimento de softwares de coleta específicos.

Por outro lado, o emprego de contadores de desempenho pode acarretar crescimento da quantidade de registros coletados e na quantidade de atributos que compõem a base de dados. Uma possibilidade para mitigar este problema é selecionar os atributos mais relevantes e que mais contribuem para o processo de classificação. Na literatura, trabalhos como os de Mondal e Bours [2015] e Chen et al. [2016] se concentram na extração manual de características dos dados de origem (do inglês, *handcraft features* - HF) . As desvantagens das abordagens manuais de extração de características são que os recursos criados ou selecionados manualmente consomem tempo, são específicos do domínio e exigem conhecimento especializado [Ronao e Cho 2016]. Este problema pode ser enfrentado utilizando rede neurais que possibilitem a extração automática de características como as redes convolucionais (*Convolutional Neural network* - CNNs) [Yang et al. 2015].

Outra questão importante é que trabalhos, como os de Fridman et al. [2015] e Bailey et al. [2014], avaliam seus resultados usando taxa de falso positivo (FPR), falso negativo (FNR), taxa de erro (EER) e acurácia (ACC). Entretanto, estes métodos falham por não considerar o fato que um usuário não consegue manter um padrão comportamental constante ao longo do tempo. Por exemplo, um mesmo usuário poderia digitar um texto de forma lenta ou acelerada, dependendo do tempo disponível para essa atividade ou passar a utilizar o clique do mouse mais lentamente devido a uma lesão existente na mão. Estes desvios de comportamento podem levar sistema de autenticação a erros como falsas rejeições ou falsas aceitações. Para minimizar estes erros, alguns estudos de biometria comportamental para fins de CA propõem que a avaliação final seja feita a partir da utilização de modelos de confiança ao invés das métricas de avaliação comumente adotadas como FPR, FNR e EER [Mondal, S. e Bours P. 2017] [Mondal, S. e Bours, P. 2014] [Deutschmann e Lindholm 2013]. O método de nível de confiança baseia-se num modelo de pontuação, onde comportamentos classificados como genuínos geram pontuações positivas (recompensa) e comportamentos classificados como impostores geram pontuações negativas (penalidade). Penalidades sucessivas podem levar o nível de confiança a ultrapassar um limite mínimo estabelecido, o que acarreta o bloqueio do sistema até que nova autenticação por senha seja efetuado.

1.1 Objetivos

Para lidar com os problemas mencionados acima, esta pesquisa tem como objetivo desenvolver um método de autenticação contínua baseado em dados extraídos a partir dos contadores de desempenho do sistema operacional e demonstrar a eficácia do método proposto na identificação de usuários impostores identificando padrões comportamentais dos usuários a partir dos efeitos que os atos ou ações voluntárias geram para os inúmeros componentes de um sistema operacional.

Para atingir esse objetivo, pretende-se alcançar os seguintes objetivos específicos:

- Definir um mecanismo de tratamento dos dados coletados pelo contadores de desempenho do sistema operacional que servirá de entrada para a criação do modelo de autenticação;
- Definir e implementar uma arquitetura de rede profunda baseada em redes neu-

rais convolucionais (CNN) com camadas recorrentes *Long Short-Term Memory* (LSTM) nas etapas de aprendizado e classificação. As camadas de convolução são usadas no processo de extração automática de características (neste caso, correlações entre dados dos contadores de desempenho) e as camadas de recorrência são utilizadas para capturar características temporais dos dados processado pelas camadas convolucionais;

- Empregar no modelo proposto a metodologia de nível de confiaça aplicado em [Barghouthi e Bours, 2009], de modo que efetue uma avaliação continuada das atividades do usuário com o objetivo de evitar o bloqueio de usuários genuínos e impedir que um impostor fique muito tempo agindo sem ser detectado.

1.2 Estrutura do Documento

O restante desta dissertação está organizado como segue:

O Capítulo 2 introduz os conceitos fundamentais para a compreensão do método proposto, como a definição e os tipos de autenticação encontrados na literatura. Além disso, o capítulo apresenta uma breve descrição os métodos de avaliação e os métodos de classificação utilizados neste trabalho.

O Capítulo 3 apresenta os trabalhos da literatura para autenticação contínua baseado em biometria comportamental, os quais estão organizados conforme os tipos de dados empregados para esta finalidade.

O Capítulo 4 apresenta uma descrição sobre os contadores de desempenho do sistema operacional, detalha a fase de tratamento dos dados como limpeza, segmentação, normalização e remoção de dados correlacionados. Por fim, apresenta o modelo de autenticação, que utiliza uma rede neural profunda na tarefa de classificação e o modelo de confiança empregado como critério de avaliação.

O Capítulo 5 apresenta o protocolo experimental, as bases de dados utilizadas nesta pesquisa, a metodologia de separação dos dados de treino e teste, as métricas de avaliação utilizada, apresenta os resultados obtidos e avalia as diversos aspectos dos experimentos.

O Capítulo 6 apresenta uma discussão sobre os pontos positivos e negativos encontrados no decorrer da pesquisa, mostrando as conclusões acerca dos resultados obtidos

pelo método proposto e das tecnologias empregadas. Por fim, são apontadas futuras direções.

2 FUNDAMENTAÇÃO TEÓRICA

Este Capítulo apresenta os principais conceitos para o entendimento e correta compreensão deste trabalho. São apresentados os conceitos de autenticação, os tipos de autenticação (estática e contínua), alguns dos principais métodos de autenticação encontrados na literatura e os tipos de dados empregados. O Capítulo também explica as abordagens de avaliação empregadas e os principais métodos de classificação utilizados.

2.1 Biometria e Autenticação

Segundo o dicionário da língua portuguesa Michaelis, autenticar significa "autorizar ou certificar como legítimo ou autêntico, segundo as fórmulas legais". No contexto deste trabalho, equivale a responder se o usuário é quem ele alega ser. A literatura apresenta duas abordagens de autenticação: os métodos de autenticação estáticos e contínuos [Mondal e Bours 2015].

Métodos de autenticação estáticos, também conhecidos por *one-time authetication*, são executados uma vez no início de uma sessão do usuário durante o procedimento de login. Este tipo de autenticação possibilita, por exemplo, que um usuário não autorizado tenha acesso a informações quando o usuário legítimo deixar seu computador sem bloquear a sessão. Os métodos de autenticação estáticos são baseados no conhecimento prévio de informações do usuário (por exemplo, senha de acesso, código PIN), em algo que o usuário possui (por exemplo, tokens de acesso ou cartões de identificação) ou baseiam-se em características biométricas intrínsecas ao usuário como impressão digital ou leitura da íris.

Como alternativa, os métodos de autenticação contínuos implementam verificações continuadas de modo a garantir que o usuário que faz uso do sistema continue sendo o usuário autenticado no início da sessão. Havendo dúvida sobre a autenticidade

do usuário, o sistema pode bloquear a sessão forçando o usuário a autenticar se novamente através do método de autenticação estática disponível. A autenticação contínua não é uma solução de segurança alternativa para o login inicial e deve ser vista como medida de segurança complementar [Mondal e Bours 2017].

Métodos contínuos de autenticação baseiam-se principalmente em características biométricas comportamentais do usuário como dinâmicas de pressionamento de tecla, dinâmica de uso de mouse, reconhecimento de voz e verificação de assinatura. Os métodos de autenticação biométricos serão explanados a seguir, iniciando com uma conceituação de biometria.

Segundo Michaelis, biometria, em um significado amplo, é a "ciência da aplicação de métodos de estatística quantitativa a fatos biológicos". Já no sentido estrito, pode significar o uso de características físicas em mecanismos de identificação como a impressão digital. No contexto deste trabalho, os dados biológicos que identificam unicamente um indivíduo possibilitam que os sistemas dotados de mecanismos biométricos de verificação de identidade consigam descobrir quem é o indivíduo que está utilizando o sistema.

Os fatos ou fatores biológicos mencionados e aplicados no contexto de autenticação podem ser de origens fisiológicas ou comportamentais. Segundo Oak [2017], fatores fisiológicos estão relacionados às propriedades anatômicas e biológicas de um indivíduo, com destaque para a impressão digital, reconhecimento facial e varredura de íris. Estas propriedades são relativamente estáveis e praticamente não se alteram ao longo da vida, a menos que algum acidente ou trauma no indivíduo cause a alteração da característica. Além disso, o emprego de propriedades anatômicas requer hardware específico para coleta das informações.

Por outro lado, a autenticação biométrica comportamental corresponde ao processo de mensuração das tendências comportamentais de um usuário, resultante de diferenças psicológicas e fisiológicas existentes entre indivíduos diferentes [Bailey et al. 2014]. Em geral, os modelos comportamentais são gerados a partir de dados extraídos da dinâmica de pressionamento de tecla, dinâmica de uso de mouse, reconhecimento de voz ou verificação de assinatura.

Diferente das características fisiológicas, as características comportamentais são sujeitas a maior grau de variabilidade. Por exemplo, uma simples gripe pode alterar o padrão de voz de uma pessoa. Em contrapartida, os dados utilizados para autenticação

podem ser obtidos silenciosamente, de forma transparente, sem atrapalhar o usuário genuíno e sem alertar o impostor que está sob avaliação [Mondal e Bours 2016].

2.2 Métodos de Autenticação Contínua

Na literatura, os métodos de autenticação comportamental contínua são criados a partir de características biométricas extraídas da dinâmica de digitação (*keyboard dynamics* – KD) e do movimento de mouse (*mouse dynamic* – MD).

2.2.1 Dinâmica de digitação (KD)

Um método de autenticação baseado em características extraídas da dinâmica da digitação tem como hipótese que a velocidade de digitação varia de indivíduo para indivíduo. Consequentemente, se o ritmo de cada indivíduo é único, é possível identificá-lo através destas características [Cai et al., 2014]. No estudo da dinâmica da digitação de uma pessoa existem várias características que podem ser medidas enquanto o usuário digita textos no teclado, como exemplos:

• Latência entre digitações consecutivas que pode ser medida através da tempo entre pressionamento de duas teclas, entre duas teclas serem soltas ou entre o pressionar de uma tecla e soltar de outra. Latência de pressionamento de duas teclas é denominada digraph, sendo que esta expressão pode ser generalizada para n-graph, onde n é o número de teclas consecutivas pressionadas.

• Duração do tempo em que a tecla é mantida pressionada.

• Velocidade total de digitação.

• Frequência de erros e correções de erros de digitação.

• O hábito de utilizar teclas em determinadas posições do teclado como, por exemplo, utilizar os números do “keypad” ou os números do próprio teclado.

• Correlação entre as teclas pressionadas, principalmente quando se digita letras maiúsculas e acentuação.

2.2.2 Dinâmica de Uso do Mouse (MD)

Dinâmica de movimentação do mouse está baseada na hipótese de que cada indivíduo interage com o sistema através do mouse de forma única. Para a autenticação biométrica baseada na dinâmica do uso do mouse é necessário capturar a trajetória do mouse e os dados do clique do mouse enquanto os usuários interagem com seu sistema.

Como exemplos de características que podem ser extraídas temos o tipo de ação (por exemplo, 1 - movimento do mouse; 2 - silêncio; 3 - point e click; 4 - drag and drop), a distância percorrida (em pixels), o tempo decorrido do movimento (unidade em segundos) e direção do movimento.

2.3 Métricas de Avaliação

Considerando os critérios de avaliação empregados nos estudos de CA, a maioria dos estudos avalia suas abordagens usando métricas como taxas de falso positivo, falso negativo, e de erro, além da acurácia [Fridman et al. 2015] [Bailey, Okolica, e Peterson 2014]. Outra abordagem possível é definir um modelo de confiança como o proposto inicialmente por Bours e Barghouthi [2009]. O modelo de confiança fundamenta-se na premissa que mesmo um usuário genuíno pode agir com um padrão diferente do habitual gerando, deste modo, falsos negativos no processo de classificação.

Abordagens que expressam os resultados atravez de taxas de erros e taxas de acertos, utilizam uma matriz de confusão para avaliar esses resultados. Segundo Duda e Stork [2001], no contexto de aprendizado de máquina, uma matriz de confusão é uma tabela que permite a visualização do desempenho de um algoritmo de classificação. Esta matriz busca entender a relação entre acertos e erros que o modelo apresenta, conforme Tabela 2.1, onde são quantificados basicamente o número de casos falsos positivos, falsos negativos, verdadeiros positivos e verdadeiros negativos.

Tabela 2.1: Matriz de confusão.

	Condição positiva	**Condição negativa**
Condição positiva prevista	Verdadeiro positivo	Falso positivo
Condição negativa prevista	Falso positivo	Verdadeiro negativo

Da Tabela 2.1, derivamos outras opções de visualização de resultados, entre os quais: acurária, taxa de verdadeiro positivo (TPR), taxa verdadeiro negativo (TNR),

taxa de falso positivo (FPR) e taxa de falso negativo (FNR).

$$Acuracia(ACC) = \frac{\sum VerdadeiroPositivo + \sum VerdadeiroNegativo}{\sum PopulacaoTotal} \tag{2.1}$$

$$TaxaVerdadeiroPositivo(TPR) = \frac{\sum VerdadeiroPositivo}{\sum CondicaoPositivo} \tag{2.2}$$

$$TaxaVerdadeiroNegativo(TNR) = \frac{\sum VerdadeiroNegativo}{\sum CondicaoNegativo} \tag{2.3}$$

$$TaxaFalsoPositivo(FPR) = \frac{\sum FalsoPositivo}{\sum CondicaoPositivo} \tag{2.4}$$

$$TaxaFalsoNegativo(FNR) = \frac{\sum FalsoNegativo}{\sum CondicaoNegativo} \tag{2.5}$$

Onde:

- Acurácia (ACC): medida de desempenho global que avalia a proporção de classificações corretas, sejam tanto os casos positivos quanto negativos;
- Taxa Verdadeiro Positivo (TPR): quantifica os elementos que foram classificados como classe positiva e que pertencem a classe positiva;
- Taxa Verdadeiro Negativo (TNR): quantifica os elementos que foram classificados como classe negativa e que pertencem a classe negativa;
- Taxa Falso Positivo (FPR): quantifica os elementos que foram classificados erroneamente como classe negativa mas que pertencem a classe positiva;
- Taxa Falso Negativo (FNR): quantifica os elementos que foram classificados erroneamente como classe positiva mas que pertencem a classe negativa.

2.3.1 Modelo de Confiança

No modelo de confiança, as ações realizadas por um usuário em avaliação são comparadas continuamente com um modelo matemático que qualifica as ações realizadas pelo usuário genuíno. Assim, se uma ação específica for executada de acordo

com a forma como o usuário genuíno executaria a tarefa (modelo do usuário genuíno), a confiança do sistema nesse usuário aumentará. Tal procedimento é chamado de recompensa.

Por outro lado, se houver um desvio entre o comportamento do usuário genuíno e o usuário em avaliação, a confiança do sistema nesse usuário diminuirá, ocasionando uma penalidade na confiança. Penalidades sucessivas podem levar o nível de confiança a ultrapassar um limite mínimo de confiança estabelecido, o que ocasiona o bloqueio do sistema até que nova autenticação (por exemplo, autenticação por senha) seja efetuada, como mostrado na Figura 2.1. Espera-se que o usuário genuíno gere mais recompensas sucessivas ao longo de um período de avaliação se comparado a um usuário impostor.

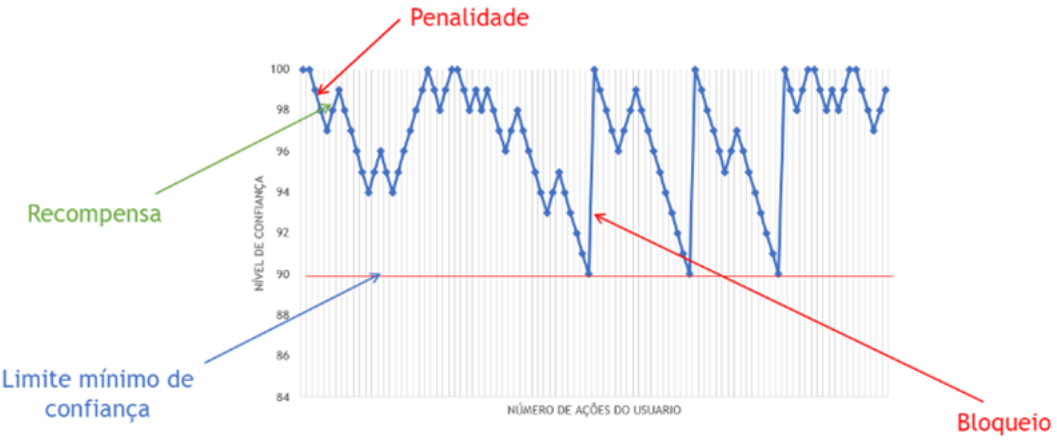

Figura 2.1: Visualização gráfica do modelo de confiança.

O objetivo do modelo de confiança é detectar um usuário impostor num menor tempo possível e evitar bloquear indevidamente um usuário legítimo. Para alcançar este objetivo é necessário medir o desempenho em termos de número médio de ações de um impostor (*Average Number of Imposter Actions* - ANIA) e do número médio de ações genuínas (*Average Number of Genuine Actions* - ANGA).

A função de cálculo de ANGA é dada por:

$$ANGA = \frac{1}{n}\sum_{i=1}^{n}\frac{ag}{bg} \tag{2.6}$$

onde n é o número usuários, ag é o número de ações genuínas de cada usuário e bg

é o número de vezes que o usuário genuíno é bloqueado indevidamente (bloqueio genuíno).

A função de cálculo de ANIA é dada por:

$$ANIA = \frac{1}{n}\sum_{i=1}^{n}\frac{ai}{bi} \tag{2.7}$$

onde n é o número usuários, ai é o número de ações impostoras de cada usuário e bi é o número de bloqueios impostores. O desejável é que haja um número reduzido de bloqueios genuínos (bg) e um elevado número de bloqueio impostor (bi). Quando não há bloqueios genuínos, bg é zero e ANGA tende ao infinito.

2.3.1.1 Variações do Modelo de Confiança

O modelo de confiança, inicialmente proposto por Bours e Barghouthi [2009], tem sido continuamente melhorado. Bours [2012] introduz a proposta de adoção de dois diferentes limiares de punição e recompensa ao modelo de confiança. Este conceito ficou conhecido depois por nível de confiança estático. Mondal e Bours [2015] contribuem com o desenvolvimento do conceito de nível de confiança dinâmico, onde pretende superar limitações existentes no nível de confiança estático, em especial sua incapacidade de trabalhar com limiares de punição e recompensa que se adequem aos padrões de comportamento de cada usuário avaliado.

Algorithm 1: Algoritmo do nível de confiança estático com duas faixas.

Dados:
$x_i \rightarrow$ Estimativa gerada pelo classificador para a amostra i
$T_c \rightarrow$ Limiar recompensa penalidade
$NC_{i-1} \rightarrow$ Nível de confianca antes da i-ésima ação
Função de recompensa: $f_{recompensa}(x_i) = x_i$
Função de penalidade: $f_{penalidade}(x_i) = -(1 - x_i)$
$\Delta NC \rightarrow$ Recompensa/Penalidade
Resultado: $NC_i \rightarrow$ Nivel de confiança da i-ésima ação
BEGIN:
if $x_i > T_c$ **then**
    $\Delta NC = f_{recompensa}(x_i)$
else
    $\Delta NC = f_{penalidade}(x_i)$
$NC_i = minmaxNC_{(i-1} + \Delta NC, 0, 100$
END:

O algoritmo de cálculo do nível de confiança estático pode ser construído usando diferentes limiares de penalidades e recompensas. Em sua versão mais simples, com apenas dois limiares (Algoritmo 1), sendo um limiar de penalidade e outro de recompensa. Primeiro é verificado se a estimativa gerada pelo classificador para uma amostra é superior ou inferior ao limiar de recompensa/penalidade. Caso seja superior (linha 12), aplica-se uma recompensa com cálculo baseado na função de recompensa. Caso contrário, uma penalidade é atribuída pela função de penalidade (linha 14).

Outras variações do modelo de confiança como com com quatro limiares, sendo dois de penalidades e dois de recompensas foram propostas em [Mondal e Bours 2015].

Para superar algumas das limitações do modelo de confiança estático, Mondal e Bours [2015] também propõem o modelo de confiança dinâmico, descrito no Algoritmo 2. Esse algoritmo utiliza vários parâmetros e devolve a confiança do sistema da autenticidade do usuário após a ação atual realizada pelo usuário. Os parâmetros para este algoritmo podem variar para diferentes usuários. O parâmetro A corresponde ao valor limiar para penalidade ou recompensa do modelo de confiança. Se a pontuação de classificação (x_i) da ação atual é superior a esse limiar, então significa que é uma recompensa, caso contrário, significa que é uma penalidade. O parâmetro B é a largura do sigmoide para esta função. Os parâmetros C e D são o limite superior de recompensa e penalidade .

Algorithm 2: Algoritmo do nível de confiança dinâmico.

1 **Dados:**
2 $x_i \rightarrow$ Estimativa gerada pelo classificador para a amostra i
3 $A \rightarrow$ Limiar recompensa/penalidade
4 $B \rightarrow$ Largura do sigmoid
5 $C \rightarrow$ Limite superior para recompensa
6 $D \rightarrow$ Limite superior para penalidade
7 $\Delta NC(i-1) \rightarrow$ Nivel de confianca antes da i-ésima acao
8 $\Delta NC \rightarrow$ Recompensa/Penalidade
9 **Resultado:**
10 $NC_i \rightarrow$ Nível de confianca da i-ésima ação
11 **BEGIN:**
12 $\Delta NC = \min(-D + D(\frac{1+\frac{1}{C}}{\frac{1}{C}+\exp(-\frac{x_i-A}{B})}), C)$
13 $\Delta NC_i = \min(\max(NC(i-1) + \Delta NC, 0), 100)$
14 **END:**

O modelo de confiança dinâmico é bem parecido com o modelo de confiança estático. A principal diferença é que no modelo estático o limiar de recompensa/penalidade

é fixo enquanto para o modelo de confiança dinâmico este limiar é diferente para cada usuário, conforme a Equação abaixo:

$$ANIA = \frac{1}{n}\sum_{i=1}^{n} E_i \tag{2.8}$$

onde E_i é a estimativa gerada pelo classificador para a amostra i durante processo de validação de treinamento.

2.4 Modelos de Classificação

A análise de séries temporais é um importante instrumento no desenvolvimento de modelos comportamentais para estudo de autenticação contínua. Os modelos estatísticos para séries temporais utilizam o passado histórico das variáveis para projetar observações futuras ou fazer predições que estejam relacionadas ao comportamento histórico. Redes neurais recorrentes (RNN) têm sido empregadas com o objetivo de tratar problemas que envolvem séries temporais.

Estudos de autenticação contínua, em sua maioria, têm utilizado técnicas de classificação baseados em aprendizado raso, seguido de classificadores baseados em distância e redes neurais. Sendo necessário efetuar a extração manual de características dos dados de origem (do inglês, *handcraft features* - HF) como, por exemplo, Mondal e Bours [2015] e Chen et al. [2016].

Segundo Ronao e Cho [2016], abordagens manuais de extração de características e de transformação de dados manualmente consomem tempo, são específicos do domínio e exigem conhecimento especializado. Este problema será enfrentado neste trabalho utilizando redes convolucionais (*Convolutional Neural network* - CNNs) com o objetivo de extrair as características mais relevantes e que mais contribuem para o processo de classificação de forma automática.

2.4.1 Redes Neurais Recorrentes (RNN)

Um componente importante no desenvolvimento de métodos de classificação que empregam redes neurais são as redes neurais recorrentes (RNN). Para entender o conceito de RNN no contexto de aprendizado de máquina, pode-se fazer uma analogia com o processo de aprendizado humano. Por exemplo, ao se ler um livro, a compreensão

do texto ocorre a partir do entendimento dos textos anteriores. Para que isso ocorra, é necessário que os aprendizados iniciais persistam em memória para que então o texto completo possa ser compreendido. As redes neurais tradicionais não armazenam informações de memória anterior, o que é um grande limitador do uso de redes neurais. Por sua vez, as redes neurais recorrentes conseguem persistir as informações anteriores por meio de loops (Figura 2.2:) que operam como memória dos eventos recentes.

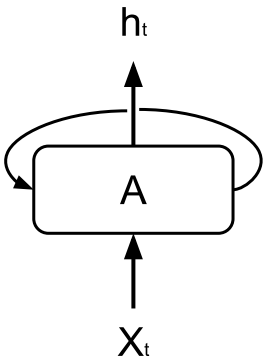

Figura 2.2: Rede neural recorrente com loop.
Fonte: https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Uma rede neural recorrente pode ser vista como várias cópias da mesma rede, cada uma passando uma mensagem para um sucessor. O desenho da rede recorrente (Figura 2.2) pode ser expandido de modo que cada rede se conecte ao próximo. Estas ligações permitem que as informações sejam passadas de uma etapa da rede para a próxima de forma encadeada (Figura 2.3:).

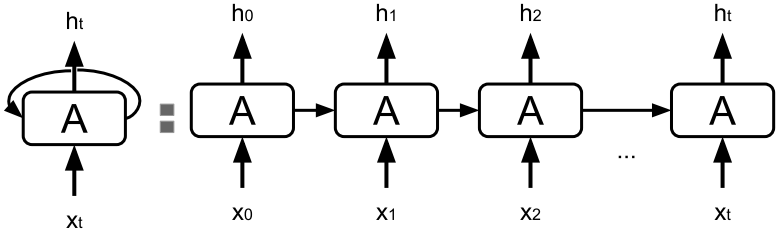

Figura 2.3: Encadeamento de redes neurais
Fonte: https://colah.github.io/posts/2015-08-Understanding-LSTMs//

Se pensarmos cada entrada da rede [X0, X1, X2, ..., Xt] como séries temporais, sequências ou listas, percebemos que temos uma arquitetura natural da rede neural para usar esses tipos de dados. Este encadeamento revela como as redes neurais recorrentes

interagem com as redes próximas. Apesar de conseguirem persistir a memória para as redes próximas, redes RNN não são efetivas quando se trata de problemas onde se exige memória de longo prazo.

2.4.2 Long Short Term Memory (LSTM)

Um tipo especial de redes neurais recorrentes são as LSTMs (*Long Short Term Memory*). Estas redes foram especialmente criadas para evitar o problema de memória de longo prazo comum em redes neurais recorrentes e dispõem de mecanismos internos chamados gates que podem regular o fluxo de informações. A Figura 2.4 fornece uma ilustração gráfica dos componentes internos de uma célula LSTM.

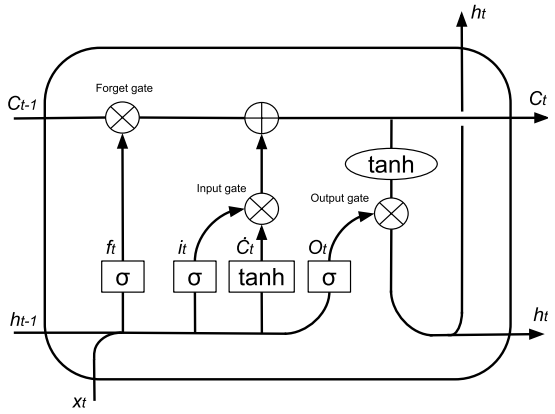

Figura 2.4: Estrutura interna de uma célula LSTM.
Fonte: https://colah.github.io/posts/2015-08-Understanding-LSTMs//

Os gates podem aprender quais dados em uma sequência são importantes para manter ou jogar fora. Ao fazer isso, eles podem transmitir informações relevantes a longa cadeia de seqüências para fazer previsões. Quase todos os resultados de última geração baseados em redes neurais recorrentes são alcançados com redes LSTM e suas variações. As redes LSTMs podem ser encontradas em aplicações como reconhecimento de voz, síntese de fala e geração de texto e tratamento de séries temporais.

Uma rede LSTM tem três destes gates para proteger e controlar o estado da célula: forget gate, input gate e output gate.

• Forget Gate: As informações que não são mais úteis no estado da célula são removidas com o forget gate. Na equação 2.9, temos duas entradas: x_t (entrada no momento específico) e h_{t-1} (saída de célula anterior) que são alimentadas ao gate e

multiplicadas por matrizes de peso W_f, seguidas pela adição do bias (b_f). O resultante é passado por uma função de ativação (e.g sigmoid) que fornece uma saída binária. Se para um determinado estado de célula a saída for 0, a informação é esquecida e para a saída 1, a informação é retida para uso futuro.

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \tag{2.9}$$

• Input Gate: A adição de informações úteis ao estado da célula é feita pelo input gate. Primeiro, a informação é regulada usando a função sigmoide que filtra os valores a serem lembrados de forma similar ao forget gate usando as entradas x_t e h_{t-1}, equação 2-10. Então, um vetor é criado usando a função tanh (equação 2-11) que dá saída de -1 a +1, que contém todos os valores possíveis de x_t e h_{t-1}. Os valores do vetor e os valores regulados são multiplicados para obter as informações úteis aplicando a equação 2-12.

$$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i) \tag{2.10}$$

$$\dot{C}_t = \tanh(W_{\dot{C}}.[h_{t-1}, x_t] + b_{\dot{C}}) \tag{2.11}$$

$$C_t = f_t.C_{t-1} + i_t.\dot{C}_t \tag{2.12}$$

• Output Gate: A tarefa de extrair informações úteis do estado da célula atual para ser apresentadas como uma saída é feita pelo output gate. Primeiro, um vetor é gerado aplicando a função tanh na célula. Então, a informação é regulada usando a função sigmóide que filtra os valores a serem lembrados usando as entradas x_t e h_{t-1} (equação 2-13). Os valores do vetor e os valores regulados são multiplicados para serem enviados como uma saída e entrada para a próxima célula (equação 2-14).

$$O_t = \sigma(W_O[h_{t-1}, x_t] + b_O) \tag{2.13}$$

$$h_t = O_t.\tanh(C_t) \tag{2.14}$$

Um componente importante para a rede LSTM é o barramento que representa a célula de memória (C_t), a linha horizontal que atravessa a parte superior da Figura 2.4. Este barramento interliga todas as células, e permite que informações sejam transferidas para as próximas células através do encadeamento conforme Figura 2.5.

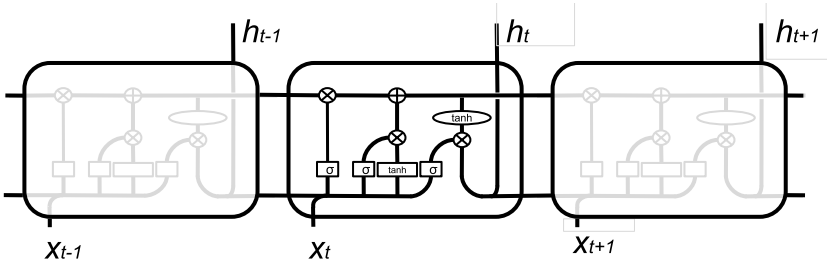

Figura 2.5: Células LSTM encadeadas em um barramento.
Fonte: https://colah.github.io/posts/2015-08-Understanding-LSTMs//

2.4.3 Redes Neurais Convolucionais (CNN)

Na arquitetura utilizada neste estudo para classificação, outros componentes importantes são as redes neurais convolucionais (CNN), entretanto para compreender sua funcionalidade é necessário revisar os conceitos matemáticos de convolução e agrupamento (ou pooling):

• Convolução: No ponto de vista da matemática, a convolução é uma função linear que se baseia na multiplicação da matriz de entrada pelo kernel de convolução, também conhecido por detector de características, e como resultado desta operação obtemos o mapa de características (Figura 2.6). O kernel atua como uma janela deslizante que irá percorrer os dados de entrada da esquerda para a direita e de cima para baixo de acordo com o valor do stride length[1] definido, realizando a multiplicação entre a sua matriz e o contexto atual em que a janela se encontra, somando os valores da multiplicação resultante. O tamanho do kernel varia de acordo com o valor do parâmetro escolhido. Uma escolha comum na literatura é uma janela 3x3 e tem seus valores inicializados de forma randômica sendo ajustados de acordo com o treinamento da rede. Outro parâmetro que influencia diretamente no mapa de característica resultante é o padding que é uma borda adicionada a entrada para evitar que haja redução no tamanho do mapa de características quando não desejável;

[1] O stride length é o tamanho do passo do kernel ao percorrer a imagem.

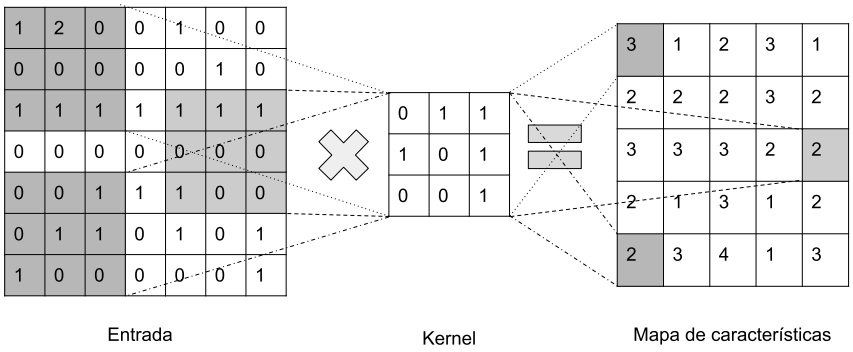

Figura 2.6: Processo de convolução.

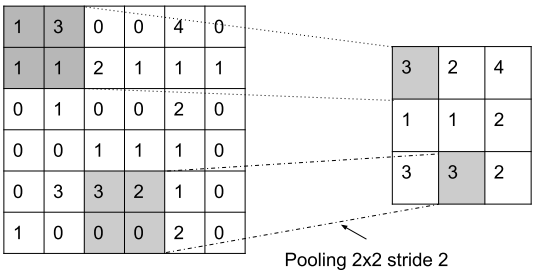

Figura 2.7: Função Pooling 2 x 2.

• Pooling: Operações de pooling tem um papel fundamental em redes neurais convolucionais (Convolucional Neural Network - CNN). Tais operações reduzem o tamanho dos mapas de características utilizando alguma função que sumariza uma subregião (DUMOULIN; VISIN, 2018). Existem diversas funções de pooling, entretanto a mais aceita e utilizada na literatura atualmente é o pool máximo (ou max pooling) que propaga para as próximas camadas o maior valor em uma janela NxN do mapa de característica. Como o próprio nome diz, o max pooling escolhe o maior valor em uma janela NxN do mapa de característica e ignora todo o resto (Figura 2.7). Mesmo reduzindo o tamanho do mapa de característica, as características ainda são preservadas, pois se sabe que o maior valor é exatamente onde foi encontrado a maior similaridade com um detector de característica. Tal técnica reduz o número de parâmetros que vão para as próximas camadas da rede, prevenindo o overfitting (sobre-ajuste, termo utilizado em estatística para descrever quando um modelo se ajusta bem um conjunto de dados específico, e mostra-se ineficaz para prever novos resultados) por justamente propagar as características mais relevantes e otimizando o tempo de processamento.

Continuando o entendimento de redes neurais convolucionais (CNN), a arquitetura de uma CNN está organizada em camadas (Figura 2.8), sendo a primeira composta por

uma camada de convolução com o objetivo de extrair características dos dados de entrada, a segunda camada é responsável por efetuar redução no mapa de características por meio de operações de pooling e a última é composta de uma camada flatten que tem como objetivo facilitar a integração com outras camadas de classificação.

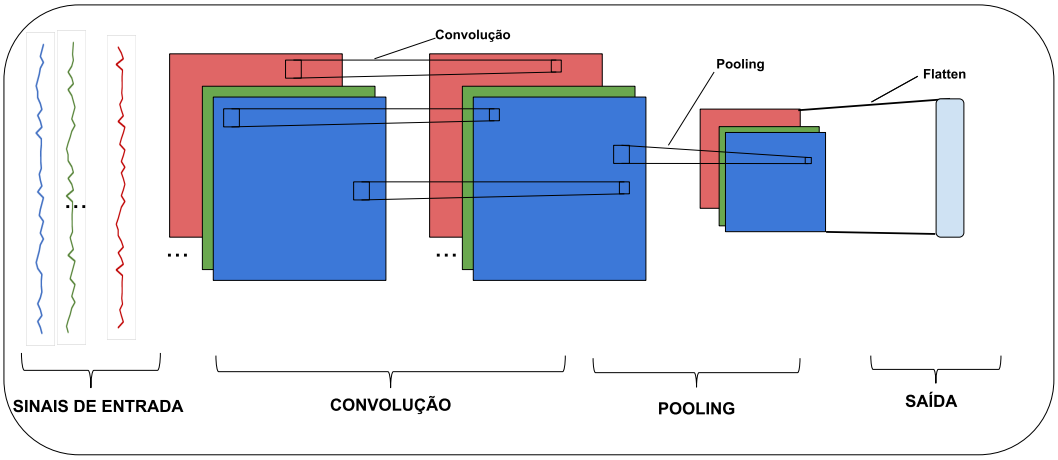

Figura 2.8: Arquitetura de uma CNN.

Após uma camada de convolução e pooling ocorre perda de informação. A perda de informação é inevitável e até desejável pelo fato que o objetivo do detector de característica ser justamente preservar aquelas mais discriminativas e relevantes ao contexto da aplicação e descartar o restante. Na maioria das vezes, os detectores possuem seus valores iniciados aleatoriamente o que resulta em mapas desconexos para a visão humana. Contudo, à medida que a rede vai aprendendo, os valores do detector são ajustados e a rede decide o que deve ser mantido ou alterado.

2.5 Considerações Finais

Neste capítulo foram apresentados os principais fundamentos teóricos que ajudarão a compreender os trabalhos relacionados, o detalhamento da abordagem de autenticação contínua que será empregada neste estudo e o modelo experimental desenvolvido. Foram detalhadas as tecnologias aplicadas nos processos de classificação e as razões que motivaram suas escolhas.

3 TRABALHOS RELACIONADOS

O controle de acesso ao sistema utilizando login inicial é geralmente utilizado como uma prova única de identidade e confia-se que a legitimidade de um indivíduo seja a mesma no meio da sessão. Uma vez que a identidade do usuário é verificada, todos os recursos estão acessíveis a ele durante uma sessão inteira, e qualquer indivíduo não autorizado que tenha acesso ao sistema desbloqueado pode ter acesso a informações críticas, o que demonstra a vulnerabilidade de sistemas baseados somente em login e senha.

Como opção, alguns autores estudam sistemas de autenticação contínua baseada em biometria comportamental onde a autenticidade de um indivíduo é verificada com base na atividade de um indivíduo que atualmente trabalha em um computador. No momento em que surge a dúvida sobre a identidade do usuário, a estrutura pode travar e um indivíduo precisa retornar à autenticação comum para chegar ao sistema de controle para continuar funcionando.

As pesquisas de autenticação contínua baseada em biometria comportamental utilizam como dados de estudo as sequências de cliques e movimentos de mouse (MD) com [Nakkabi, Y. et al., 2010], [Shen et al. 2012], [Feher et al., 2012] e [Mondal et al., 2015], dados obtidos pela dinâmica de digitação em teclados (KD) com [Bours e Barghouthi, 2009], [Revett, 2009], [Vural et al. 2014], [Sun et al. 2017] e [Locklear et al. 2014], uma combinação multimodal de MD e KD com [Jagadeesan e Hsiao, 2009], [Friedman et al. 2015],[Bailey et al. 2014], [Mondal e Bours, 2016].

3.1 Autenticação Contínua baseado em Sequências de cliques e movimentos de mouse (MD)

Nakkabi et al. [2010] já estavam preocupados com a forte variabilidade oriunda da utilização de dados de dinâmica do mouse e que tende a afetar a sua precisão. Para

reduzir a variância do sistema de autenticação proposto, os autores utilizam técnicas de fusão para mesclar pontuações biométricas obtidas pelo classificador *Learning Algorithm for Multivariate Data Analysis* LAMBDA [Piera e Aguilar, 1991]. Os resultados experimentais utilizando uma base de dados de 48 usuários mostram que o método proposto é capaz de obter taxas de falso positivo de 0% e taxa de falso negativo de 0,36%.

Feher et al. [2012] apresentam um amplo estudo sobre os atributos relacionados a dinâmica do mouse encontrados na literatura e propõem expandir a quantidade de atributos a serem utilizados através de uma proposta de organização hierárquica dos dados em 4 níveis. No nível mais baixo, as atividades são decompostas em 5 eventos básicos, que ao final são expandidos para um total de 354 possíveis eventos e que podem ser utilizados como atributos em estudos de autenticação contínua. Eles coletaram dados de 25 usuários usando um classificador Random Forest e alcançaram EER de 8,53% (em 30 ações) com tempo de autenticação de menos de 2 minutos.

Shen et al. [2012] desenvolveram o estudo em uma base com dados de 28 usuários, com foco em diferentes eventos do mouse e onde não há necessidade de dados de treinamento de impostores, pois utilizam One Class SVM como classificador. O melhor resultado foi uma taxa de falso positivo de 0,37% e uma taxa de falso negativo de 1,12%.

Mondal et al. [2015] investigam o desempenho de um sistema de autenticação biométrica contínua considerando um amplo espectro de análises, dentre as quais podemos listar a implementação de testes em diferentes combinações de técnicas de fusão para classificação, a realização de ajustes de threshold para definir os parâmetros do nível de confiança ideal para o modelo, o desenvolvimento de técnicas de incremento de pontuação (boosting) de recompensa e penalidade para casos onde haja alta probabilidade de classificação do evento como genuíno ou impostor e a fizeram um extenso estudo comparando os modelos de confiança estática versus modelos de confiança dinâmico. A base de dados utilizada foi a de Nakkabi et al. [2010], e os autores mostram que o desempenho alcançado com as técnicas implementadas melhorou significativamente os resultados obtidos em outros estudos utilizando a mesma base de dados, com taxas de falso positivo 0% e taxa de falso negativo de 0%..

3.2 Autenticação Contínua baseado na Dinâmica de Digitação(KD)

Revett [2009] coletou dados de 20 participantes aos quais foram solicitados realizar 100 auto-logins e 100 ataques a outras contas e buscaram encontrar a latência máxima entre digitações de duas e três teclas. Esses atributos são usados para construir um modelo de como um usuário digita e foi usado um algoritmo de correspondência de sequencias de texto para determinar se são capazes de autenticar e identificar os usuários participantes e obtiveram resultados assertivos em 100% dos usuários.

Bours e Barghouthi [2009] apresentam pela primeira vez o modelo de nível de confiança como métrica de avaliação alternativa a maioria dos autores que utilizam FNR, FPR, EER e ACC. Em seus experimentos utilizaram dados com a latência entre pressionamento e liberação de teclas e aplicaram métodos de classificação Euclidianos e Bayesianos. E foram capazes de identificar e bloquear um usuário impostor entre 79 e 348 teclas digitadas.

Vural et al. [2014] desenvolveram um estudo com o objetivo de ser utilizado como baseline para outros experimentos e desta forma disponibilizam as bases de dados. O novo conjunto de dados inclui dados de teclas de frases curtas, texto fixo com transcrição de prosas longas e texto livre. Além dos dataset, disponibilizam alguns algoritmos utilizados e imagens das coletas geradas. Utilizaram os algoritmos de Gunetti e Picardi [2005] e Leggett at al. [1991], onde obtiveram taxas de falso positivo de 0,75% e taxas de falso negativo de 3,93%.

Sun et al. [2017] investigaram o desempenho de sistemas biométricos utilizando dinâmica de digitação de teclado de 75 usuários utilizando o mesmo teclado e também compartilharam a base de dados. Utilizaram métodos estatísticos gaussianos em suas análises e validaram os resultados obtidos comparando aos de Vural et al. [2014] com taxas de erro de 0,08% para autenticação de usuários.

Locklear et al. [2014], diferente de outros trabalho que utilizam somente dados de dinâmica de digitação, propõem um método para extrair informações cognitivas durante a digitação do texto, entre os quais estão a sofisticação do vocabulário, fluência da língua e compreensão de estruturas gramaticais, para fins de autenticação contínua. Como classificador utilizaram o método baseado em distância Manhattan Distance (MD). Usando somente dados cognitivos obtiveram percentual de usuários au-

tenticados de 98,7% e aplicando métodos de fusão de classificadores com dados de dinâmica de digitação os resultados foram de 99,96%.

3.3 Combinação de Dinâmica de Digitação(KD) e Dinâmica do Mouse (MD):

Jagadeesan e Hsiao [2009] desenvolvem experimentos com 20 usuários tendo coletado 62 atributos (10 de MD e 52 de KD). Neste trabalho de autenticação contínua empregam classificadores com ANN, KNN e métodos estatísticos e obtém resultados com acurácia de 96.4%, taxa de falso positivo de 3,6% e taxa de falso negativo de 0%.

Friedman et al. [2015] buscam identificar quais os atributos que mais contribuem para a classificação final, usando método de fusão de classificadores individuais. Atributos com base nas características da distância da trajetória do mouse, curvatura da trajetória do mouse, duração do pressionamento de uma tecla e intervalo entre pressionamento tiveram maior contribuição entre os atributos utilizados. Os resultados indicam taxas de falso positivo de 0,004% e taxas de falso negativo de 0,01% após 30 segundos de interação do usuário.

Bailey et al. [2014] partem da premissa que avaliar MD e KD sem considerar o contexto da aplicação que está em uso pode comprometer os resultados obtidos, para sustentar essa ideia, os autores exemplificam que os padrões de uso do mouse e teclado são diferentes, quando se utiliza o aplicativo Word da Microsoft para escrever um documento e de quando se navega Internet utilizando um navegador (browser). Para classificação dos resultados usaram a fusão de rede bayesiana, SVM e árvores de decisão e obtiveram a taxa de falso positivo de 2.10% e taxa de falso negativo de 2.24%.

Mondal et al. [2016] ampliam os estudos de Mondal et al. [2015] com MD e nível de confiança, acrescentando dados de KD. Para isso, coletam dados de 53 voluntários entre 5 a 7 dias. Efetuam testes com combinações de técnicas de fusão de diferentes métodos de classificação entre os quais Artificial Neural Network (ANN), Counter-Propagation Artificial Neural Network (CPANN) e SVM. O melhor resultado obtido nesta pesquisa é que 50 dos 53 usuários genuínos nunca são inadvertidamente bloqueados pelo sistema, enquanto os 3 usuários genuínos restantes (ou seja, 5,7%) às vezes são bloqueados, em média, após 2.265 ações. Além disso, existem apenas 3

de 2.756 impostores não foram detectados, ou seja, apenas 0,1% dos impostores não foram detectados. Impostores são detectados em média após 252 ações.

3.4 Dados de sistema operacional

Referente aos trabalhos que abordam dados de sistema operacional, Song et al. [2013] e Malatras et al. [2017] contribuem ao apresentar abordagens inovadoras, utilizando somente dados de camadas de sistemas operacional.

Song et al. [2013] apresentam uma proposta de autenticação contínua utilizando dados coletados por aplicativo próprio que entre outros atributos coleta informações sobre a janela ativa, o arquivo ativo, a conexão de rede ativa, monitor de processos, log de sistema. Os autores estadam quais algoritimos são mais adequados para modelar o comportamento dos usuários, e para isso usam *Gaussian mixture model* (GMM) combinado com o algorítimo *Fisher features* comparando com SVM, e o método de Parzen, obtendo resultado 17,6% superior para GMM.

Malatras et al. [2017] trazem um estudo de identificação contínua com dados exclusivamente de indicadores de desempenho do sistema operacional, baseado em arquitetura de algoritmos de aprendizagem de máquina supervisionada para distinguir os usuários e informá-los sobre sua exposição ao anonimato. Entre os atributos utilizados estão o percentual de CPU utilizada, a memória livre e utilizada do sistema, número de conexões de rede ativas, o número total segmentos de dados enviados e recebidos, e outros. Apresentam resultados de 92,53% para o algoritmo 5-NN quando todos os atributos são considerados, entretanto quando utiliza somente CPU, memória, TCP MIB e estatísticas de rede, e aplica o algoritimo J48, obtem o seu melhor resultado de 94,748%.

3.5 Discussão

Estudos em autenticação contínua podem ser organizados quanto aos critérios de avaliação, métodos de classificação e tipos de dados utilizados, Tabela 3.1. Nesta seção, é apresentada de forma resumida uma discussão de alguns trabalhos encontrados em revisão bibliográfica.

Considerando os critérios de avaliação empregados nos estudos de CA, a maioria

dos trabalhos avalia suas abordagens usando métricas como taxas de falso positivo, falso negativo, erro e acurácia. [Nakkabi, Y. et al., 2010] [Friedman, et al., 2015] [Bailey, et al., 2014]. Entretanto, estes trabalhos não consideram o fato que um usuário não consegue manter um padrão de comportamento ao longo do tempo. Para um sistema de autenticação contínua, na verdade, não é apenas importante saber se um impostor é detectado, mas quando ele é detectado, ou seja, determinar a quantidade de ações ele foi capaz de realizar antes de ser detectado [Bours e Barghouthi, 2009], nesse trabalho é apresentado pela primeira vez o modelo de nível de confiança como métrica de avaliação alternativa as métricas comumente usadas como FNR, FPR, EER e ACC.

Outros estudos trouxeram contribuições e melhorias ao modelo de nível de confiança proposto por Bours e Barghuthi, apresentando propostas de adoção de diferentes limiares de punição e recompensa ao modelo de confiança [Mondal, S. e Bours, P., 2015], [Mondal, S. e Bours, P., 2016], [Mondal, S. e Bours, P., 2017].

Considerando os métodos de classificação empregados, os trabalhos apresentados podem ser agrupados em: i) métodos que usam classificadores baseados em distância (entre estas, R-distance, A-distance, distância euclidiana ou distância em escala de Manhattan); ii) métodos que usam classificadores baseados em aprendizado raso (Support Vector Machine - SVM, k-Nearest Neighbors - KNN, Naives Bayes - NB, Decision Tree - DT, Random Forest - RF, J-48, K-MEANS, entre outros menos utilizados); e, iii) métodos que utilizam técnicas diversas (Fuzzy, Gunetti & Picardi's Algoritm, Gaussian mixture model - GMM).

Diferente dos outros trabalhos, o método proposto neste estudo utiliza contadores de desempenho do sistema operacional como atributos empregados para fins de autenticação contínua de usuários. Contadores de desempenho já haviam sido utilizados por Malatras, A. et al.[2017] em um estudo de identificação contínua de usuários. Este trabalho também se diferencia dos outros trabalhos por fazer uso de redes neurais profundas enquanto outros trabalham usam predominantemente os classificadores baseados em distância para classificação, seguido de aprendizado raso e redes neurais artificiais (ANN).

Tabela 3.1: Resumo dos trabalhos relacionados.

Tipo de dados	Autores	Usuários	Classificador	Resultados
MD	()[Nakkabi, Y. et al., 2010]	48	LAMBDA	FP 0%, FN 0,36%.
	[Shen et al. 2012]	28	One Class SVM	FP 0,37%, FN 1,12%.
	[Feher et al., 2012]	25	Random Forest	EER de 8,53%
	[Mondal et al., 2015]	48	ANN e SVM	FP 0%, FN 0%.
KD	[Bours e Barghouthi, 2009]	25	Euclidianos e Bayesianos	impostor identificado entre 79 e 348 teclas digitadas.
	[Revett, 2009]	20	Variados	acurácia 100%
	[Vural et al. 2014]	39	Gunetti & Picardi e Leggett	FP 0,75%, FN 3,93%
	[Sun et al. 2017]	75	métodos estatísticos gaussianos	Taxas de erro de 0,08%
	[Locklear et al. 2014]	486	distância Manhattan Distance	Acurácia de 99,96%
KD e MD	[Jagadeesan e Hsiao, 2009]	20	ANN, KNN e métodos estatísticos	acurácia de 96.4%, FP 3,6% e FN 0%.
	[Friedman et al. 2015]	67	NV e SVM	FP 0,004%, FN 0,01%
	[Bailey et al. 2014]	31	NB, SVM e J48	FP 2.10%, FN 2.24%.
	[Mondal e Bours, 2016].	53	ANN, CPANN e SVM	FP 5,7%, FN 0,1%
Contadores de desempenho	Método proposto	26 e 37	CNN e LSTM	FP 0%, FN 0%.

4 AUTENTICAÇÃO CONTÍNUA BASEADA EM CONTADORES DE DESEMPENHO DO SISTEMA OPERACIONAL

Este Capítulo descreve a abordagem proposta e empregada neste trabalho, iniciando na seção 4.1 onde detalha os processos de coleta dos dados. A seção 4.2 explana a fase de tratamento dos dados, incluindo a limpeza, normalização e segmentação. A seção 4.3 descreve o processo de separação de dados e apresentação das abordagens de verificação que foram empregadas. A seção 4.4 detalha as fases de treino e teste com destaque para os componentes de aprendizado de máquina que utilizam uma rede neural profunda na tarefa de classificação e o nível de confiança que é empregado como critério de avaliação do resultado.

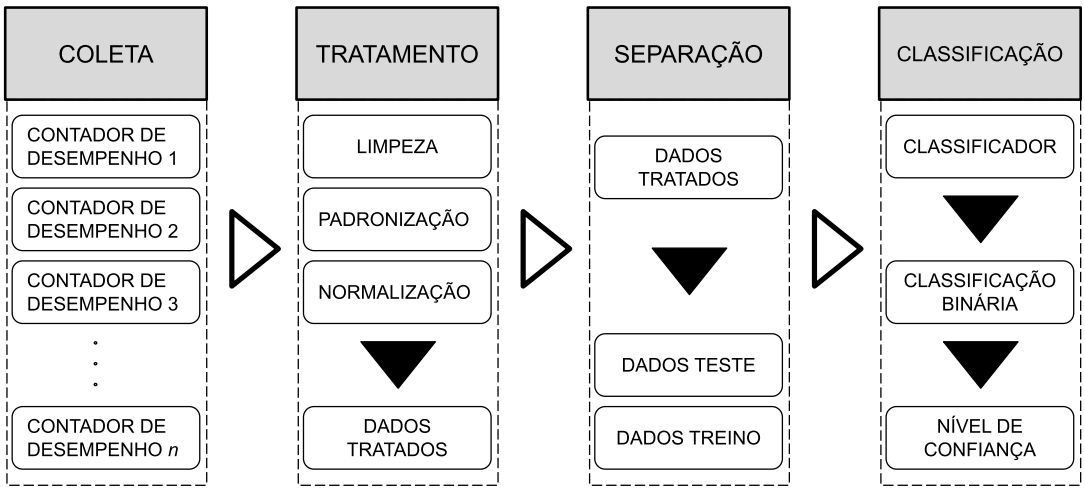

Figura 4.1: Visão geral da abordagem proposta para autenticação contínua.

4.1 Abordagem proposta

A abordagem proposta neste trabalho é apresentada na Figura 4.1 e é composta de módulos que são executados em etapas sequenciais, de forma estanque, onde os produtos finais são repassados as camadas subsequentes. Nessa lógica, os dados de contadores de desempenho do sistema operacional são coletados em computadores desktop e armazenados em arquivos identificando os computadores de origem.

Na sequência, inicia a fase de tratamento dos dados que passam por processos de limpeza, padronização e normalização dos dados, preparando-os para as etapas subsequentes, e entregando num formato adequado a ser apresentado aos classificadores da rede neural, seja na fase de treino como de teste. Entretanto os dados tratados devem ser separados de modo a atender as necessidades formais exigidos pelos métodos de treinamento e teste que irão servir de base para todos os experimentos subsequentes.

Na sequência das atividade, na etapa de classificação, primeiro é executado o treinamentos dos dados com o objetivo de gerar o modelo computacional que é o identificador único de cada usuário. Em seguida aplicamos o identificador único gerado para cada usuário para classificar os dados separados para teste. Com base nos resultados dos dados de teste classificados, aplica-se a metodologia de avaliação pelo método de nível de confiança e por fim encontra-se os resultados para avaliação final dos objetivos desta pesquisa. Os detalhamento dos módulos serão apresentados a seguir.

4.2 Contadores de Desempenho do Sistema Operacional

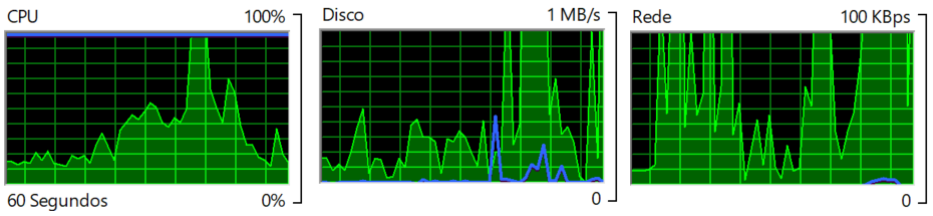

Figura 4.2: Visualização gráfica dos contadores de desempenho.

Neste trabalho, os registros estatísticos dos contadores de desempenho (*performance counters* – PCs) de sistema operacional (SO) são utilizados como características para gerar modelos de autenticação contínua de usuários. Estes contadores têm

como objetivo quantificar diferentes tipos de eventos do SO. Os eventos disponíveis e o número de contadores dependem do tipo e versão do sistema operacional. A Tabela 4.1 fornece uma lista com exemplos de alguns contadores de desempenhos que podem ser medidos.

Tabela 4.1: Exemplos de características (contadores de desempenho) coletadas.

Contadores de desempenho
Interface de rede Bytes enviados por segundo
Interface de rede Bytes recebidos por segundo
Interface de rede Pacotes enviados por segundo
Interface de rede Pacotes por segundo
LogicalDisk (Total) Média de disco por gravação
LogicalDisk (Total) Média de disco por leitura
Memória Bytes de cache
Memória Páginas por segundo
Processo (Total) Bytes de gravação de E por segundo por segundo
Processo (Total) Bytes de leitura de E por segundo por segundo
Processo (Total) Número de Identificadores
Processo (Total) Operações de dados de ES por segundo
Processador (Total) Percentual tempo de processador
Processador (Total) Percentual tempo de usuário

A Figura 4.2 fornece uma visualização gráfica dos contadores CPU, Disco e Rede de uma máquina com o sistema operacional Windows. Diversos softwares podem ser utilizados com objetivo de proceder com a coleta automatizada. No sistema operacional Linux, pode-se utilizar o programa "PERF"[1], enquanto nos sistemas operacionais Windows, o programa "PERFMON"[2] pode ser utilizado para a mesma finalidade. Além disso, mesmo em ambientes virtuais como o Vmware já existem contadores de desempenho nativos nas máquinas virtuais como o programa "vRealize Hyperic"[3].

Para os estudos conduzidos neste trabalho foram utilizados dois conjunto de dados (Dataset1 e Dataset2) de contadores de desempenho do sistema operacional Windows extraídos a partir do programa PERFMON.

[1]http://www.brendangregg.com/perf.html

[2]https://docs.microsoft.com/pt-br/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc749154(v=ws.11)

[3]https://www.vmware.com/products/vrealize-hyperic.html

O primeiro dataset corresponde a dados de 26 computadores (um usuário por computador) com diferentes configurações de hardware e versões de sistemas operacionais. No total foram coletados dados de 159 contadores de desempenho. O segundo dataset corresponde a dados de 37 computadores (um usuário por computador) com configurações idênticas de hardware, programas instalados (softwares) e sistema operacional. No total foram coletados dados de 218 contadores de desempenho. A Tabela 4.1 apresenta exemplos de características coletadas em ambos datasets. A relação completa dos contadores de desempenho utilizados neste trabalho pode ser obtida no ANEXO 1.

4.3 Tratamento dos Dados

Ao longo do processo de coleta de dados, alguns problemas na qualidade dos dados podem ocorrer, como a falta de padronização das características a serem extraídas, a diferença entre as escalas dos dados e atributos altamente correlacionados. Embora muitos algoritmos de aprendizagem de máquina tenham sido projetados para manipular dados em tais situações, pode-se esperar que esses algoritmos produzam resultados mais precisos caso a maioria desses problemas presentes nos dados tenham sido resolvidos. Para tratar estes problemas, a etapa de tratamento utiliza um conjunto de técnicas para realizar a limpeza, padronização e normalização dos dados. Na etapa de limpeza dos dados são extraídos os acentos e caracteres especiais constantes nos nomes dos atributos, que podem gerar erros nas fases seguintes. Além disso, são excluídos todos os atributos que não variam ao longo da coleta, i.e., desvio padrão igual a zero, considerando todos os usuários coletados. Por fim, transformação de valores NULL em “0”;

Como parte dos dados são coletados em computadores diferentes, para um mesmo padrão de sistema operacional, configurações de linguagem distintas podem gerar nomes de atributos diferentes para o mesmo atributo. Por essa razão, um script de padronização é utilizado para evitar diferenças na representação de nomes de atributos.

Os atributos também passam por um processo de normalização, visto que alguns atributos utilizam escalas diferentes. Assim, atributos numéricos são normalizados dentro de uma escala de valores, atributos simbólicos precisam ser codificados em valores numéricos e valores desconhecidos precisam ser preenchidos usando de métodos como médias dos valores dos atributos. Os dados foram normalizados pela normaliza-

ção z-scores que permite a conversão dos valores de um atributo A baseado na média e desvio padrão deste atributo, segundo a fórmula abaixo.

$$v\prime = \frac{v - \bar{A}}{\sigma_A} \tag{4.1}$$

Onde $v\prime$ é o valor normalizado, v é um valor do atributo A, $\bar{A}$ é a média de A e $sigma_A$ é o desvio padrão de A.

4.4 Separação dos Dados

O objetivo desta etapa é separar os dados que serão utilizados na geração do modelo de autenticação (treino) e no teste. Para a fase de treino é necessário que os dados estejam preparados para serem apresentados a um classificador binário. Portanto, os dados devem estar rotulados como "genuíno"ou “impostor”.

Para a base de treino, a quantidade de dados do usuário genuíno é de 50% do total de dados de um determinado usuário. Os dados de treinamento dos usuários impostores são separados dos outros usuários, de modo que a quantidade total de dados de todos os impostores juntos seja igual à quantidade de dados de treinamento do usuário genuíno. Isso é feito para evitar que haja viés em relação à classe genuína ou à classe impostora. Para a base de teste, de forma geral, utiliza-se os dados que não foram utilizados para treinamento.

4.5 Classificação: Rede Neural DEEPCONVLSTM

Nesta abordagem, o modelo de classificação utiliza uma arquitetura de rede neural profunda denominada DEEPCONVLSTM, proposta por Ordóñez [2016]. A arquitetura combina camadas convolucionais, as quais atuam como extratoras de características (neste caso, correlações entre dados dos contadores de desempenho) e fornecem representações abstratas dos dados de entrada. Em seguida, camadas de recorrência são empregadas para capturar características temporais dos dados processados pelas camadas convolucionais.

Segundo Palaz et al [2015], aplicar aos dados brutos técnicas de extração de características, na maioria das vezes, leva a um desempenho superior do classificador. Deste modo, foi empregado as redes neurais recorrentes podem receber como entrada

os dados brutos coletados pelos contadores de desempenho. Alem disso, a descoberta manual de características requer conhecimento especializado, e a escala dos dados brutos produzidos pelos contadores de desempenho é um fator limitador. Por essa razão, as redes convolucionais (Convolutional Neural network - CNNs) podem ser utilizadas para tratar estes desafios [Yang et al. 2015].

Outro aspecto importante é que as dependências internas entre os dados dos contadores de desempenho podem trazer informações de contexto significativas ou padrões desconhecidos que podem ser úteis para identificar comportamentos. Por exemplo, uma correlação entre o uso de navegador, interface de rede e processador. Para tirar proveito destas correlações, este projeto adiciona duas camadas recorrentes LSTM (Long Short-Term Memory) [Hochreiter e Schmidhuber, 1997]. Segundo Ordóñez [2016], o uso de duas camadas recorrentes em profundidade é suficiente para capturar as relações temporais das características.

A entrada para a rede neural profunda consiste em segmentos da série temporal para vários canais, onde cada atributo (contador de desempenho) corresponde a um canal. Na Figura 4.6, o número de canais é representado por D e S é o tamanho do segmento por canal (número de amostras por canal). Assim, os dados de entrada são transformados através de três camadas convolucionais, que são subtidos a processos de convolução e pooling, e a qual é aplicado a um kernel (K1, K2 e K3), representados pelos retângulos vermelhos na Figura 4.6. Neste caso, o kernel é representado por um vetor de pesos de tamanho 4. O kernel se desloca sobre os dados de entrada, executando uma multiplicação elementar com a parte da entrada em que está atualmente, e então somando os resultados em um único ponto de saída. Essas camadas convolucionais empregam como função de ativação unidades lineares retificadas (ReLUs) para calcular os mapas de características.

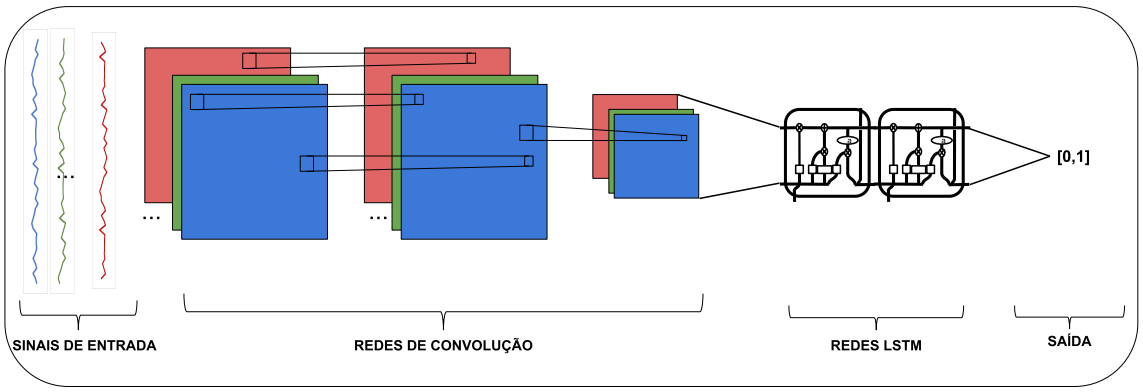

Figura 4.3: Rede DEEPCONVLSTM.

Os dados resultantes das camadas de convolução são passados para as camadas densas recorrentes. As unidades de um LSTM são usadas como unidades de construção e, neste caso, cada camada recorrente é composta por 128 unidades, com saída de tamanho 20. A saída da rede é obtida através de uma camada densa de 1 unidade com a função de ativação sigmoid, que contém a probabilidade da amostra pertencer ao usuário genuíno ou impostor.

4.5.1 Nível de Confiança: Obtenção de ANGA e ANIA

O processo de avaliação do nível de confiança e da quantificação de ANGA e ANIA é realizado durante a etapa de teste em momentos distintos. Para obtenção de ANGA de um usuário, a parcela genuína dos dados de teste deste usuário é apresentada ao seu modelo previamente treinado na rede DEEPCONVLSTM, e para cada instância da referida base de dados, é obtida uma probabilidade de esta instância pertencer ao usuário em análise.

Para a obtenção de ANIA, a parcela impostora dos dados de teste deste usuário também é apresentada ao seu modelo previamente treinado na rede DEEPCONVLSTM, sendo que cada usuário que compõe a parte impostora dos dados de teste é avaliado separadamente, e para cada instância, é obtida uma probabilidade desta instância pertencer ao usuário em análise.

O índice de nível de confiança é obtido da avaliação da sequência destas probabilidades ao longo do tempo usando o algoritmo de nível de confiança.

Entretanto, o conceito de ANGA e ANIA está relacionado ao número de ações

executadas pelo usuário genuíno e impostor, respectivamente. Essas ações nos estudos de Mondal e Bours [2015] são geradas a partir de atos voluntários do usuário na interação com o mouse. Entretanto no estudo de contadores de performance o processo de coleta é periódico, divergindo portanto da ideia de ação voluntária. O que não invalida a metodologia, visto que podemos abstrair o conceito de "ações" como sendo "observações" obtidas em intervalos de tempo.

4.5.2 Considerações do capítulo

Este capítulo apresentou uma visão geral da abordagem proposta e seu processo de classificação, detalhou o tratamento a que os dados de origem são submetidos, e apresentou a metodologia geral do projeto e a arquitetura da rede profunda DEEPCONVLSTM empregada para treinamento e classificação dos dados. No capitulo seguinte são apresentados maiores detalhes do modelo experimental e dos resultados obtidos.

5 EXPERIMENTOS E RESULTADOS

Este Capítulo descreve os experimentos efetuados e os resultados obtidos. A seção 5.1 detalha o protocolo experimental empregado e as peculiaridades de cada experimento executado. A seção 5.2 explana as diferenças entre as diferentes bases de dados empregadas. A seção 5.3 apresenta as considerações gerais sobre a separação dos dados e parametrizações empregadas. A seção 5.4 explora os resultados obtidos em cada experimento. A seção 5.5 avalia os vetores probabilidade obtidos para ampliar a análise dos resultados obtidos. Por fim, a seção 5.6 apresenta as considerações finais sobre os resultados alcançados.

5.1 Protocolo Experimental

A arquitetura proposta é avaliada considerando três conjuntos de experimentos e três bases de dados, conforme detalhado a seguir.

5.1.1 Experimentos

5.1.1.1 Experimento 1

Este estudo teve como objetivo validar o modelo proposto no contexto de autenticação contínua utilizando dados de contadores de desempenho do sistema operacional. Para enfrentar este problema e estabelecer o primeiro parâmetro comparativo, utilizamos a base de dados, o protocolo experimental e as métricas de avaliação empregadas por Mondal e Bours [2015].

O trabalho de Mondal e Bours [2015] utilizou técnicas de aprendizado computacional raso, especificamente máquinas de vetores de suporte (SVM) e redes neurais artificias (ANN) cujos resultados foram combinados usando o algoritmo MCF (Multi Classifier Fusion). Este trabalho, por outro lado, emprega uma arquitetura de rede com-

posta por camadas convolucionais e de recorrência, DEEPCONVLSTM. Estas redes já haviam sido estudadas em processamento de sinais, mas não haviam sido empregadas para estudo de autenticação contínua.

Quanto às análises dos resultados obtidos, embora o trabalho de Mondal e Bours [2015] seja muito abrangente e avalie múltiplos algoritmos de níveis de confiança. Não era pretensão deste estudo repetir todas as análises por eles efetuadas. Para efeitos de comparação e obtenção de um baseline, somente foi utilizado o algoritmo de nível de confiança que apresentou melhor resultado.

5.1.1.2 Experimento 2

O experimento 2, por sua vez, teve como objetivo avaliar se os indicadores de desempenho do sistema operacional são capazes de gerar modelos de redes neurais profundas eficazes para autenticação contínua de usuários. O experimento também ajudou a estabelecer os parâmetros gerais que foram empregados neste estudo.

Como não encontramos trabalhos que empregassem contadores de desempenho do sistema operacional como base de dados em estudos de autenticação contínua, os resultados deste experimentos serão comparados com os resultados obtidos no experimento 1.

5.1.1.3 Experimento 3

Após a avaliação do experimento 2 e análise dos resultados foi identificado a necessidade de estudos adicionais devido à possibilidade que os resultados apresentados indicarem a capacidade dos métodos de classificação de diferenciar o hardware do computador de origem, quando na realidade deveria autenticar o usuário do computador.

Para dirimir esta dúvida, nova coleta foi necessária, e desta vez, os usuários voluntários fizeram uso do mesmo padrão de computador e do mesmo conjunto de softwares instalados. Nesta nova coleta foi ampliado o número de participantes, o número médio de instâncias coletadas para cada usuário, bem como o número de atributos.

5.1.2 Base de dados

Os experimentos realizados para avaliar o método proposto utilizam três bases de dados, sendo uma para cada experimento. Serão utilizadas os dados de movimentos de mouse obtidas em Nakkabi et al. [2010] e duas outras base de dados com contadores de desempenho do sistema operacional conforme detalhamento:

5.1.2.1 Nakkabi Dataset

O estudo desenvolvido por Mondal e Bours [2015] utilizou a base de dados coletada no estudo de Nakkabi et al. [2010]. Esta base foi gentilmente cedida pelo Prof Dr. Issa Traore, Universidade de Victoria - Canadá, o que permitiu implementar um baseline de avaliação da rede DeepConvLSTM para dados biométricos comportamentais, nesse caso movimentos de mouse.

O Nakkabi Dataset é composto pela coleta de movimento de mouse de 49 usuários voluntários, aos quais foi solicitado que usassem o computador de maneira normal, sem quaisquer restrições às tarefas que deveriam executar. Para cada ação do mouse de um voluntário, o software de coleta de dados armazenou os seguintes atributos: i) tipo de ação (1: movimento do mouse; 2: silêncio; 3: point e click; ou 4: drag and drop; ii) distância percorrida em pixels; iii) tempo decorrido do movimento, unidade em segundo (com um intervalo de amostragem de 0,25 segundos); iv) direção do movimento.

Para compatibilizar com o estudo de Mondal e Bours [2015], as seguintes adaptações nos dados foram feitas:

1. Tipo de ação: Foram removidos os eventos de "silêncio" dos dados brutos uma vez que o objetivo é obter o comportamento do usuário. Assim, somente são utilizados: 1: movimento do mouse; 3: point e click; ou 4: drag and drop.

2. Direção: Tomado diretamente dos dados brutos.

3. Velocidade da ação do mouse: Isso equivale à distância percorrida em pixels/o tempo decorrido.

4. Aceleração recíproca da ação do mouse: Igual ao tempo/velocidade decorrido da ação.

5. Distância percorrida (quantizado): A distância percorrida foi quantizada em agrupamentos referente ao alcance de distância percorrido em pixels.Os agrupamentos crescem segundo a escala:

    - De 1 a 1000 pixels: o tamanho do agrupamento é de 50 pixels, portanto, há 20 agrupamentos no total.
    - De 1001 a 2000 pixels: o tamanho do agrupamento é de 100 pixels, então há 10 agrupamentos no total.
    - De 2001 a 3000 pixels: o tamanho do agrupamento é de 200 pixels, então há 5 agrupamentos no total.
    - De 3001 a 4000 pixels: o tamanho do agrupamento é de 500 pixels, então há 2 agrupamentos no total.
    - Mais de 4001 pixels: Tratado como um agrupamento separado

Além das transformações citadas, os dados foram normalizados e segmentados em séries temporais pelo método de janelas deslizantes.

A análise estatística da distribuição do número de amostra para cada usuário (Tabela 5.1) permite observar que os dados desta base de dados estão desbalanceados, com um alto desvio padrão e a diferença entre mínimo e máximo na ordem de 100 vezes.

Tabela 5.1: Distribuição estatística do número de amostras do Nakkabi Dataset.

Quantidade de usuários	49
Média de amostras por usuário	47775,61
Desvio padrão	60013,39
Quantidade mínima de amostras	2908
25%	8979
50%	19985
75%	59280
Quantidade máxima de amostras	288450

5.1.2.2 Contadores de Desempenho - Dataset1

159O Dataset1 é composto de dados extraídos de contadores de desempenho do sistema operacional Windows referentes a 26 computadores (26 usuários). Cada amostra corresponde a um vetor de característica composto por 159 atributos. O Anexo I apresenta a relação de atributos coletados nos computadores do Departamento de Tecnologia da Informação de uma organização pública e teve como usuários participantes

um grupo de analistas de sistemas e de programadores, todos voluntários e para os quais não foram apresentadas quaisquer restrições às tarefas que deveriam executar. O ciclo de coleta de cada amostra foi de 5 segundos e teve uma duração média de aproximadamente de 26 (vinte e seis) horas para cada um dos 26 usuários.

A coleta foi realizada em computadores com características e configurações distintas quanto ao hardware, sistemas operacionais e softwares instalados. Essa variação nas configurações teve influência direta na seleção final de quais contadores de desempenho seriam utilizados, pois haviam atributos que estavam presentes em algumas coletas, mas não estavam em outras. Ao final, foram utilizados somente os contadores que fossem comuns a todos os usuários.

A análise estatística da distribuição do número de amostra para cada usuário (Tabela 5.2) permite observar que os dados desta base de dados estão balanceados, com um desvio padrão de 5.188,50, menos que um terço do valor médio de amostras.

Tabela 5.2: Distribuição estatística do número de amostras do Dataset1.

Quantidade de usuários	26
Média de amostras por usuário	18587.23
Desvio padrão	5188.50
Quantidade mínima de amostras	15375
25%	16623
50%	17303
75%	17536
Quantidade máxima de amostras	40523

5.1.2.3 Contadores de Desempenho - Dataset2

O Dataset2 é composto de dados extraídos de contadores de desempenho do sistema operacional Windows de 37 computadores. Cada amostra corresponde a um vetor de característica composto por 218 atributos. O Anexo I lista o conjunto de contadores de desempenhos coletados em 37 computadores de uma organização pública. Os 37 usuários executam diferentes atividades administrativas, de diversas unidades organizacionais, todos voluntários e para os quais não foram apresentadas quaisquer restrições às tarefas que deveriam executar. O ciclo de coleta de cada amostra foi de 5 segundos e teve uma duração média de coleta aproximadamente de 48 (quarenta e oito) horas. Todos os computadores possuíam características e configurações idênticas quanto ao hardware, sistemas operacionais e softwares instalados.

A Tabela 5.3 descreve estatísticas da distribuição do número de amostra para cada usuário.

Tabela 5.3: Distribuição estatística do número de amostras para o Dataset2.

Quantidade de usuários	37
Média de amostras por usuário	34518.81
Desvio padrão	17709.63
Quantidade mínima de amostras	4897
25%	23157
50%	33799
75%	43970
Quantidade máxima de amostras	85228

5.1.3 Separação dos Dados

Três cenários de avaliação foram considerados na avaliação do desempenho do método de autenticação proposto. Estes cenários são denominados de “interno”, “externo” e “híbrido”. No caso do cenário “interno”, assumimos que o sistema é usado dentro de uma organização onde os dados de todos os participantes estão disponíveis e conhecidos. Pode-se supor que, como os dados de todos os impostores são usados durante o treinamento, isso influenciará o desempenho do sistema de uma maneira positiva. Por essa razão são projetados os cenários “externo” e "híbrido". Para o cenário “externo”, assumimos que o sistema pode ser atacado apenas por pessoas para as quais não há dados disponíveis ao treinar o classificador. O cenário "híbrido"é uma combinação dos cenários “interno” e “externo”.

5.1.3.1 Cenário Interno (CI)

Este cenário de verificação simula os usuários de dentro da empresa e os dados de todos os usuários são usados para treino e para teste. Se assumirmos os N usuários da organização, cada classificador é treinado com os dados de um usuário genuíno e de N-1 usuários impostores, ou seja, todos os impostores foram considerados neste processo de verificação.

Para treino, a parte genuína é obtida de 50% da base de dados de cada usuário e a parte impostora é tomada de todos os usuários restantes (usuários impostores), conforme mostrado na Figura 5.1. Todos os usuários contribuem aproximadamente com a mesma quantidade de dados para o treinamento do classificador, treinamento

$(T_i/(N-1))$, onde T_i é o tamanho da base de treino do usuário i e N é número de usuários. A base de teste é feita com todos os dados que não foram usados para treinamento, sendo um conjunto de dados genuíno e (N-1) conjunto de dados impostores.

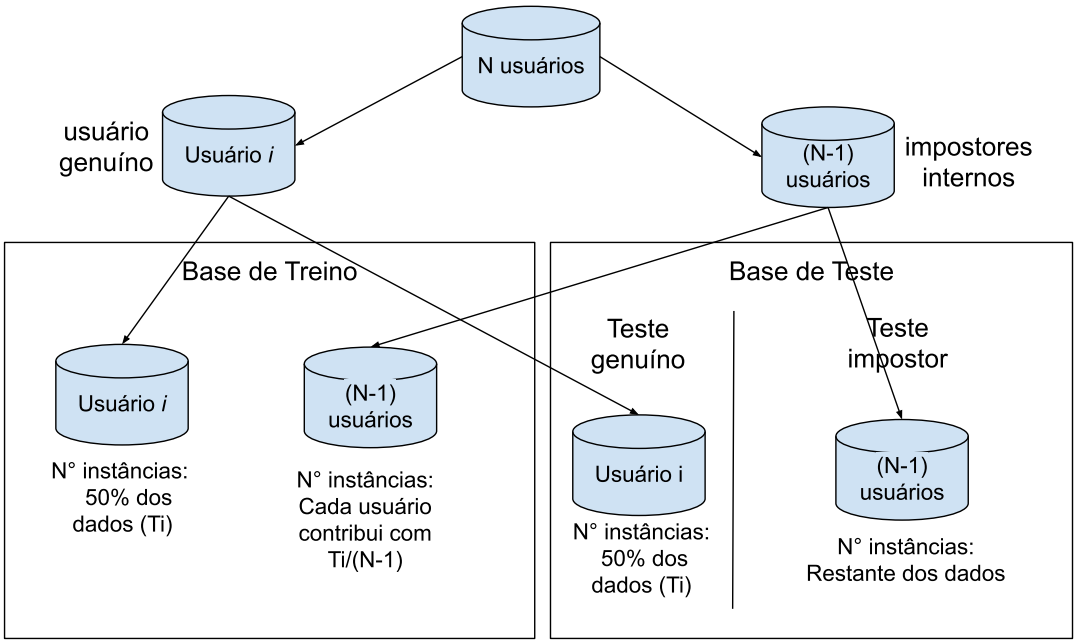

Figura 5.1: Separação dos dados de cenário interno.

5.1.3.2 Cenário Externo (CE)

A verificação do cenário externo é formada por conjuntos previamente separados de usuários impostores utilizados para treino e para teste, simulando os treinos com usuários internos e os testes somente com usuários externos. Para cada usuário genuíno, é necessário separar os impostores que farão parte do conjunto exclusivo de treino (50%) e os que farão parte do conjunto exclusivo de teste (50%). Nesta verificação, para preparação da base de treino de cada usuário, 50% vem dos dados de treino do usuário que estamos treinando, e os outros 50% vêm dos usuários que foram selecionados para o conjunto de treino (Figura 5.2). Neste cenário, os dados dos usuários selecionados contribuem com quantidade de dados igual para o treinamento ($\frac{T_i}{\frac{N-1}{2}}$), onde N é número de usuários e Ti é o tamanho da base de treino do usuário i.

Para o teste, o usuário genuíno utiliza os 50% de dados separados para teste, e a parte impostora é tomada de 50% dos impostores do conjunto de teste, utilizando todos os dados destes impostores.

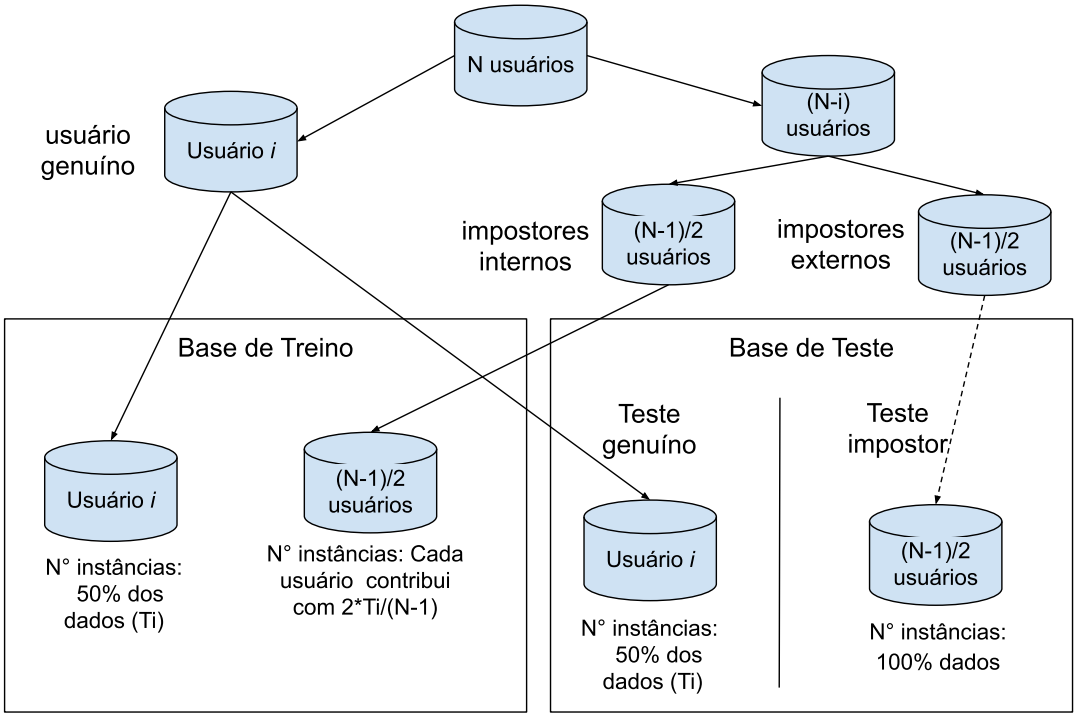

Figura 5.2: Separação dos dados de cenário externo.

5.1.3.3 Cenário Híbrido (CH)

Neste cenário (Figura 5.3), o sistema de autenticação deve levar em consideração a existência de dados de usuários internos e externos à organização. Neste método de verificação, para cada usuário será necessário separar, dentre os outros, quais usuários serão utilizados para treino e teste (50% do total de usuários e que representarão os usuários internos) e quais os usuários serão utilizados exclusivamente para teste (50% restantes e que representarão os usuários externos). Para treino, do usuário i, 50% de seus dados são separados,e cada um dos usuários internos contribui com uma quantidade de dados igual para o treinamento ($\frac{T_i}{\frac{N-1}{2}}$), onde N é número de usuários e T_i é o tamanho da base de treino do usuário i. Os dados dos usuários externos não são utilizados para treino.

Para teste, o usuário genuíno utiliza o restante de dados genuínos separados (50%) para esta finalidade. Quanto à parte impostora, dois conjuntos diferentes serão utilizados: dos usuários internos, será utilizado a parte dos dados que não foi utilizada no treino; dos usuários externos utilizamos 100% dos dados.

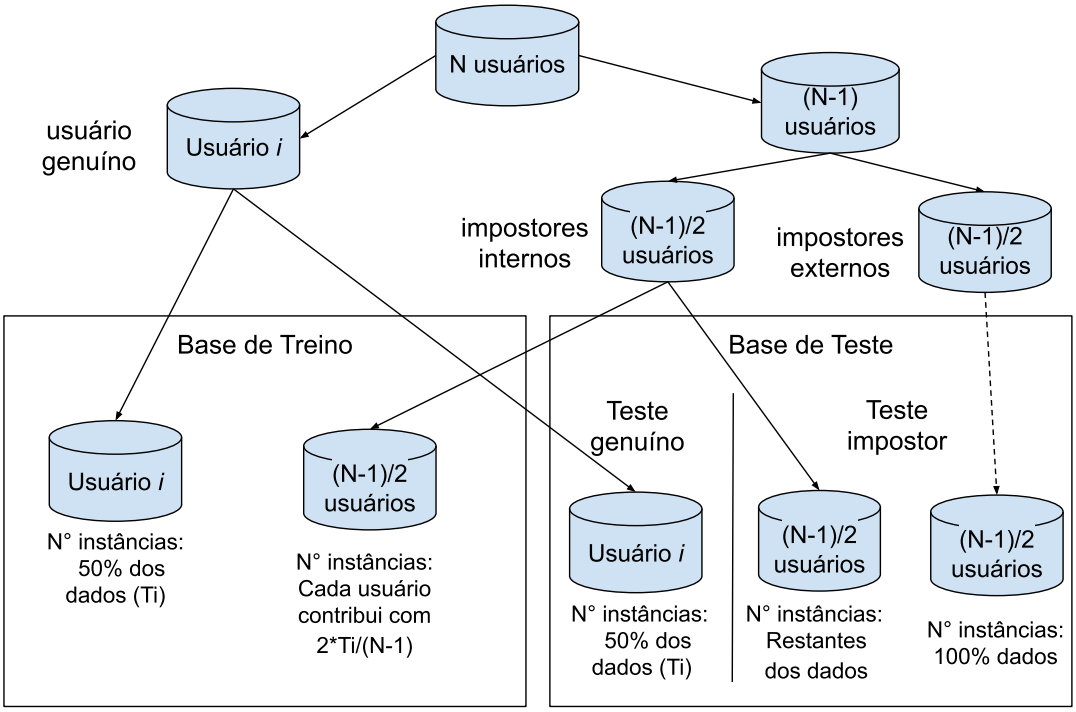

Figura 5.3: Separação dos dados de cenário híbrido.

Para minimizar o desbalanceamento de um único usuário com um número maior de dados coletados, limitou-se em 20.000 o tamanho máximo da base de treino. E por questão de organização dos dados e economia de espaço, a etapa de separação de dados, fase de treino, fase de teste são efetuadas em um fluxo contínuo para cada metodologia de verificação.

5.1.4 Treino, Teste e Validação do Modelo de Autenticação

Em aprendizagem de máquina, os classificadores são capazes de examinar os dados de itens para determinar a qual classe cada item pertence. Frequentemente, os algoritmos de classificação produzem um vetor de probabilidades que representam as probabilidades da amostra pertencer a cada classe. No contexto deste estudo, podemos simplesmente definir duas classes: usuário legítimo e impostor. A etapa de classificação é composta essencialmente de dois componentes: Treino e Teste.

Na etapa de treino, os dados separados com esta finalidade, devidamente rotulados, são apresentados à rede neural. No início do treino os pesos da rede neural estão preenchidos aleatoriamente e ao longo do treino ou "aprendizado", estes pesos são

ajustados a cada ciclo que treino de modo a aproximar o resultado obtido ao resultado esperado.

A fase de treino tem como produto os modelos personalizados, gerados pelo framework Keras/Tensorflow, e que serão utilizados na etapa de teste. Para cada usuário que se deseja autenticar, deve-se gerar um modelo que pode ser interpretado como uma "assinatura única" capaz de autenticar o usuário treinado. Os modelos são armazenados em arquivos que contém a arquitetura e os pesos da rede neural, que será utilizada depois para fazer predição.

Para a fase de teste, os dados do modelo de um determinado usuário são carregados para a rede neural. Na sequência, apresenta-se um subconjunto de dados de teste para avaliação da rede neural e como resultado obtém-se o vetor de probabilidade utilizado para predizer se o subconjunto de dados avaliado do usuário corresponde a um usuário legítimo ou impostor.

Para verificação do nível de confiança e obtenção de ANIA e ANGA, é empregado o algoritmo de nível de confiança estático com quatro faixas. O nível de confiança dinâmico não será utilizado na seção de resultados, pois avaliações preliminares indicaram não haver melhoria nos resultados. A Tabela 5.4 apresenta os parâmetros utilizados nesse modelo.

Tabela 5.4: Parâmetros empregados no modelo de nível de confiança.

Limiar de penalidade intermediária	$T_{pi} = 0.4$
Função de recompensa	$f_{recompensa}x_i = 1Xx_i$
Função de penalidade	$f^1_{penalidade}x_i = 1 - x_i, f^2_{penalidade}x_i = 1$

Com o objetivo de avaliar os resultados e padronizar sua visualização, foi adaptado uma tabela (Tabela 5.5) que mostra um extrato da classificação de n usuários genuínos e N_v agrupamentos de dados que foram rotulados como impostores, num formato adaptado de uma matriz de confusão, onde expressa os resultados gerais obtidos na classificação dos usuários genuínos e impostores comparando com seus rótulos originais.

A Tabela 5.5 mostra os valores de número médio de ações impostoras - ANIA, calculados para os dados rotulados como impostores, e número médio de ações genuínas – ANGA que são verificados para os rótulos genuínos. Além disso, também é apresentada uma análise da variação do limite mínimo do nível de confiança. Inicialmente este limite é fixado em 90 e tem seus valores ajustados a maior com o objetivo de

Tabela 5.5: Exemplo de exibição dos resultados para teste Cenário genérico.

Base de Dados		**Cenário**	
Limite Mínimo	Classificação N. de teste	Usuário n	Usuário N_v
Variável	Genuíno	n_g/n ANGA: anga	n_i/N_v ANIA: ania
	Impostor	$1 - n_g/n$ ANGA: anga	$1 - n_i/N_v$ ANIA: ania
90	Genuíno	n_g/n ANGA: anga	n_i/N_v ANIA: ania
	Impostor	$1 - n_g/n$ ANGA: anga	$1 - n_i/N_v$ ANIA: ania

maximizar o número de usuários genuínos corretamente classificados. Segue abaixo o significado dos índices utilizados na Tabela 5.5:

- n: número de usuários total do estudo;
- n_g: número de usuários que não sofreram bloqueios durante análise preditiva dos dados genuínos;
- anga: valor do número médio de ações genuínas – ANGA, obtidas para determinado usuário. Não havendo bloqueios, anga tende a infinito;
- N_v: número de agrupamentos de teste gerados pelo modelo de separação empregado;
- n_i: número de agrupamentos de teste que sofreram bloqueios durante a análise preditiva dos dados impostores;
- ania: valor do número médio de ações impostoras – ANIA, obtidas para determinado usuário;

5.1.5 Estabelecimento do Limiar Mínimo do Nível de Confiança

No modelo de confiança, penalidades sucessivas podem levar o nível de confiabilidade a ultrapassar um limite mínimo estabelecido, o que ocasiona o bloqueio do sistema até que nova autenticação (por exemplo, autenticação por senha) seja efetuada. Num cenário ideal é desejável que usuários impostores sejam rapidamente bloqueados

e usuários genuínos não sejam bloqueados indevidamente. Neste contexto, foi avaliado qual o valor ideal para o limite mínimo de confiança de modo a reduzir o número ações indesejadas de um impostor sem que venha a gerar bloqueio indevido de usuários genuínos.

O valor inicial de análise do limite mínimo do nível de confiança foi inicialmente fixado em 90 variando ao valor máximo de 100. Este estudo teve como objetivo identificar o valor de ANGA ideal onde, há o máximo de bloqueios de usuários impostores e não há bloqueio indevido de usuários genuínos.

Para o estudo de possíveis bloqueios indevidos de usuários genuínos, pode ser necessário ajustar o limite mínimo do nível de confiança incrementando o valor de 90 ou decrementando o valor de 90. Em ambos os casos, se faz necessário avaliar qual o impacto que o ajuste teve em ANIA para se determinar qual o valor ideal. Da análise dos resultados obtidos, o valor ideal para o limite mínimo do nível de confiança e que é utilizado para avaliação dos resultados é de 97.

5.2 Resultados

5.2.1 Experimento 1

Os resultados obtidos neste experimento e apresentados nas Tabelas 5.6, 5.6 e 5.8, para as testes dos três cenários de verificação, considerando o algoritmo de nível de confiança estático de quatro faixas e avaliando o comportamento para o limite mínimo de confiança fixo em 90 e variável, indicam que a rede DeepConvLSTM foi capaz de, em todas as verificações e variações do limite mínimo de confiança, identificar 100% dos usuários genuínos. Também é possível observar uma redução dos valores de ANIA em todas as avaliações, quando comparado aos resultados de compilados do trabalho de Mondal e Bours [2015] (Tabelas 5.9, 5.10 e 5.11), indicando que o modelo proposto foi capaz de reconhecer um usuário impostor.

Entretanto, quando avaliado o número de verificações de usuários impostores que erroneamente não foram devidamente bloqueados, observa-se que nos teste de CI, a rede DEEPCONVLSTM, errou em 0,26% das verificações enquanto o modelo de Mondal e Bours falhou em 1,15% das verificações, para o limite mínimo de confiança fixado em 90. Para os testes de CH, os resultados obtidos pela rede DEEPCONVLSTM indicam erro em identificar impostores em 1,02% dos casos, versus 0,47% dos casos

Tabela 5.6: Resultado obtidos com Nakkabi Dataset e para cenário CI.

Limite Mínimo	N. de verificações	49	2352
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.09% ANIA: 37.14
	Impostor	ANGA:	9.91% ANIA: 20.39
90	Genuíno	100% ANGA: ∞	0.26% ANIA: 54.01
	Impostor	ANGA:	99.74% ANIA: 12.46

Tabela 5.7: Resultado obtidos com Nakkabi Dataset e para cenário CH.

Limite Mínimo	N. de verificações	49	2352
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.94% ANIA: 155.42
	Impostor	ANGA:	96.06% ANIA: 15.81
90	Genuíno	100% ANGA: ∞	1.02% ANIA: 54.01
	Impostor	ANGA:	98.98% ANIA: 33.42

Tabela 5.8: Resultado obtidos com Nakkabi Dataset e para cenário CE.

Limite Mínimo	N. de verificações	49	1176
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	1.79% ANIA: 313.37
	Impostor	ANGA:	98.21% ANIA: 30.26
90	Genuíno	100% ANGA: ∞	1.96% ANIA: 364.26
	Impostor	ANGA:	98.04% ANIA: 55.23

obtidos por Mondal e Bours para o mesmo cenário. Em CI os resultados são 1,95% versus 1,53%. Estes resultados podem estar relacionados a distribuição da quantidade média de amostras por usuário, com alguns usuários com pouca volume de dados o que pode ter sido insuficiente para o treino da rede neural profunda.

Tabela 5.9: Resultado obtidos por Mondal e Bours com Nakkabi Dataset e para cenário CI.

Limite Mínimo	N. de verificações	49	2352
	Classificação	Genuíno	Impostor
Variável	Genuíno	93.88% ANGA: ∞	1.91% ANIA: 1967
	Impostor	6.12% ANGA: 2066	98.09% ANIA: 68
90	Genuíno	83.67% ANGA: ∞	1.15% ANIA: 926.3
	Impostor	16.33% ANGA: 3729	98.85% ANIA: 93.83

Tabela 5.10: Resultado obtidos por Mondal e Bours com Nakkabi Dataset e para cenário CH.

Limite Mínimo	N. de verificações	49	2352
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.04% ANIA: 366
	Impostor	 ANGA:	99.96% ANIA: 109
90	Genuíno	83.67% ANGA: ∞	0.47% ANIA: 781
	Impostor	16.33% ANGA: 3729	99.53% ANIA: 164.9

Tabela 5.11: Resultado obtidos por Mondal e Bours com Nakkabi Dataset e para cenário CE.

Limite Mínimo	N. de verificações	49	1176
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.17% ANIA: 1026
	Impostor	 ANGA:	99.83% ANIA: 130
90	Genuíno	85.71% ANGA: ∞	1.53% ANIA: 696
	Impostor	14.29% ANGA: 45608	98.47% ANIA: 162

Os resultados acima indicam que o modelo de Mondal foi capaz de generalizar me-

lhor quando testando dados de usuários desconhecidos, e foram obtidos considerando a decisão de limitar o número máximo de amostras em 20.000. Para a rede DEEP-CONVLSTM seria mais efetivo remover da análise os usuários como poucas amostras a limitar o número de amostras, entretanto o limite foi necessário para manter compatibilidade com os experimentos.

5.2.2 Experimento 2

As Tabelas 5.12, 5.13 e 5.14 apresentam os resultados obtidos para Dataset1 – Contadores de desempenho do sistema operacional, aplicando o algoritmo do modelo de confiança estático com quatro faixas e avaliando os três cenários.

Tabela 5.12: Resultado obtidos com Dataset1 e para cenário CI.

Limite Mínimo	N. de verificações	26	650
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 2.02
90	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 10.08

Tabela 5.13: Resultado obtidos com Dataset1 e para cenário CH.

Limite Mínimo	N. de verificações	26	650
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.46% ANIA: 140.67
	Impostor	ANGA:	99.54% ANIA: 64.18
90	Genuíno	100% ANGA: ∞	0.77% ANIA: 194.90
	Impostor	ANGA:	99.23% ANIA: 67.23

Avaliando a capacidade de reconhecer os usuários genuínos, neste experimento, 100% dos usuários genuínos não sofreram bloqueios indevidos em todas as verificações.

Nos testes das amostras impostoras, considerando o teste do cenário CI, 100%

Tabela 5.14: Resultado obtidos com Dataset1 e para cenário CE.

Limite Mínimo	N. de verificações	26	312
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	0.96% ANIA: 290.89
	Impostor		99.04% ANGA: ANIA: 132.19
90	Genuíno	100% ANGA: ∞	1.60% ANIA: 395.11
	Impostor		98.40% ANGA: ANIA: 130.14

dos impostores foram bloqueados. Apesar do bom resultado para CI, ocorreram casos de usuários impostores que não foram detectados para os testes dos cenários CE e CH, com percentuais de acerto de 99,23% e 98,40%, respectivamente. Quanto aos números médios de ações impostoras - ANIA foram 10,8, 67,23 e 130,14. Esses resultados obtidos estão coerentes com o que se esperava a priori, pois se esperava que à medida que os dados fossem verificados com usuários que não foram utilizados na base de treino, maior o número de ações para que o usuário impostor seja bloqueado.

Os resultados indicam que a rede DEEPCONVLSTM foi capaz de reconhecer 100% dos casos de usuários genuínos, sem bloqueios indevidos. Por outro lado, essa rede não teve a mesma assertividade para detectar os usuários impostores nos testes de CE e CH. Indicando dificuldade da rede neural profunda em generalizar para amostras que não tenham sido treinadas a priori.

Outro ponto de problema é o valor de ANIA de 130,14 para o teste do cenário CE. Se considerar que o intervalo entre as coletas é de 5 segundos, o período para coletar 130 ações seria: 130 (ações) X 5 (segundos), o que daria 650 segundos ou quase 11 minutos. Este tempo elevado é um impeditivo para aplicação prática do modelo, pois num cenário real um usuário impostor teria em média 11 minutos para agir sem que fosse efetivamente bloqueado.

Uma proposição para melhorar a generalização é aumentar a quantidade de dados do experimento, seja em quantidade de usuários ou em número médio de amostras por usuário. O aumento na quantidade de dados do experimento também pode ter impacto na capacidade geral da rede neural, resultando na redução dos valores de ANIA.

Baseado no fato que os dados utilizados neste experimento foram coletados em computadores com configurações de hardware, sistema operacional e conjunto de

softwares instalados distintas. Existe a possibilidade do modelo está identificando o computador quando deveria identificar o usuário. Um terceiro conjunto de experimentos foi realizado para esclarecer estas dúvidas.

5.2.3 Experimento 3

As Tabelas 5.15, 5.16 e 5.17 apresentam os resultados obtidos para Dataset2 – Contadores de desempenho do sistema operacional, aplicando o algoritmo do modelo de confiança estático com quatro faixas e usando os três cenários.

Tabela 5.15: Resultado obtidos com Dataset2 e para cenário CI.

Limite Mínimo	N. de verificações	37	1332
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 2.04
90	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 10.22

Em todas as verificações, 100% dos usuários genuínos não sofreram bloqueios indevidos e 100% dos impostores foram bloqueados. Resta então analisar os resultados dos números médios de ações que os usuários impostores foram capazes de efetuar até que fossem bloqueados pelo sistema (ANIA). Para o limite mínimo de nível de confiança de 90 os números médios de ANIA foram 10,22, 13,48 e 17,12, e se considerarmos o limite mínimo variável, os números médios de ANIA foram 2,04, 2,7 e

Tabela 5.16: Resultado obtidos com Dataset2 e para cenário CH.

Limite Mínimo	N. de verificações	37	1332
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 2.7
90	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 13.48

Tabela 5.17: Resultado obtido com o Dataset2 e para cenário CE.

Limite Mínimo	N. de verificações	37	666
	Classificação	Genuíno	Impostor
Variável	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 3.44
90	Genuíno	100% ANGA: ∞	ANIA:
	Impostor	ANGA:	100% ANIA: 17.12

3,4.

No cenário CE, onde todos os testes são efetuados com usuários desconhecidos pela rede neural, se considerarmos o intervalo entre as coletas de 5 segundos, o período para coletar 4 (valor arredondado de 3,44) amostras seriam: 4 (ações) X 5 (segundos), o que daria 20 segundos. Com um limite mínimo de nível de confiança de 90, o período para coletar 18 (valor arredondado de 17,12) seria de 90 segundos. Estes tempos de detecção e bloqueio já não são um impeditivo para aplicação prática do modelo.

Os resultados obtidos para ANIA estão coerentes com o esperado para um sistema de autenticação contínua. Para o mesmo limite mínimo do nível de confiança, variando o método de verificação, esperava-se que à medida que os dados fossem verificados com usuários que não foram utilizados na base de treino, maiores os números de ações para que o usuário impostor seja bloqueado. Para a variação no limite mínimo do nível de confiança era esperado que elevação desse valor causasse a redução do ANIA.

No geral, os resultados obtidos indicam que os objetivos estabelecidos para o experimento foram atendidos. O aumento do volume de amostras coletadas para cada usuário e o aumento do número de usuários envolvidos no estudo realmente propiciaram uma melhor eficácia do modelo proposto.

Quanto ao fato de os dados terem sido coletados em computadores seguindo um mesmo padrão de configuração, ficou evidente que o modelo foi capaz de diferenciar o modelo comportamental de cada usuário, sendo viável o uso de contadores de desempenho do sistema operacional para representar o perfil biométrico comportamental de um usuário.

5.2.4 Avaliação dos vetores probabilidade obtidos durante fase de teste

Tabela 5.18: Matriz de confusão do Experimento 2 para os três cenários.

	CI		**CH**		**CE**	
	Genuíno	Impostor	Genuíno	Impostor	Genuíno	Impostor
Genuíno	99.88% 241113	0.12% 267	99.86% 241044	0.14% 336	99.86% 241044	0.14% 336
Impostor	0.28% 32917	99.72% 11842983	4.90% 434612	95.09% 8428074	7.51% 424681	92.49% 5231836

Tabela 5.19: Matriz de confusão do Experimento 3 para os três cenários.

	CI		**CH**		**CE**	
	Genuíno	Impostor	Genuíno	Impostor	Genuíno	Impostor
Genuíno	99.61% 551002	0.39% 2115	99.51% 550424	0.49% 2693	99.51% 550424	0.49% 2693
Impostor	0.91% 140573	99.09% 15346555	4.4% 1290135	95.60% 28038603	6.29% 1184091	93.71% 17649200

A melhora geral dos resultados obtidos entre os Experimentos 2 e 3, levam a pensar haver uma melhor generalização dos usuários genuínos e impostores. Entretanto o aumento nas taxas de falso positivo (Tabelas 5.18 e 5.19), indicam que a rede DEEPCONVLSTM teve mais dificuldades de reconhecer os usuários genuínos no Experimento 3.

Quando é feita a mesma análise referente a generalização de usuários impostores, para as separações CE e CH, o aumento das taxas de verdadeiro negativo e a diminuição nas taxas de falso negativo (Tabelas 5.18 e 5.19), indicam haver melhor reconhecimento do usuário impostor no Experimento 3. Esta melhora pode estar relacionadas a maior quantidade de usuários disponíveis para treino e para teste. Entretanto, a piora para os usuários genuinos torna essa análise inconclusiva.

Comparando os resultados de taxa de falso positivo e taxa de falso negativo obtidos nas matrizes de confusão do Experimento 3 (Tabela 5.19) com os resultados aplicando a avaliação do nível de confiança, onde em todos as verificações obtivemos taxas de falso positivo de 0% e taxa de falso negativo de 0% (Tabelas 5.15, 5.16 e 5.17). Apesar de haver eventuais avaliações de falso positivo e falso negativo, na Tabela 5.18, quando aplicadas a avaliação do nível de confiança não necessariamente geram bloqueio indevido do usuário genuíno ou deixam de bloquear um usuário impostor. Conclui-se que

a avaliação do nível de confiança realmente é a metodologia mais adequada para tratar de autenticação contínua.

5.2.5 Considerações Finais

Os resultados apresentados mostram a viabilidade da abordagem para autenticação contínua de usuários baseadas em dados de contadores de desempenho de sistemas operacionais. As análises dos resultados mostram que nenhum genuíno foi bloqueado indevidamente e todos os impostores foram detectados com no mínimo de 3 ciclos de coleta realizadas quando do teste do cenário CI.

Apesar dos bons resultados obtidos, as etapas de treino e teste consumiram aproximadamente 220 horas, o que pode ser considerado um tempo elevado. Para o processamento do experimento foi utilizado um equipamento Core i7 com 32 GB de RAM e com uma GPU Nvidia MX150.

O uso de redes neurais profundas demanda uma quantidade maior de dados para processamento de treino e teste. No experimento 3 foi utilizado 164,28% mais amostras que no experimento 2. Este incremento teve impacto significativo no tempo de processamento, mas considerando os resultados obtidos nos experimentos 2 e 3, leva a crer que o incremento dos dados no Dataset2 pode ter contribuído para melhorar os resultados gerais obtidos.

Quanto aos dados de contadores de desempenho dos sistema operacional utilizados nos Dataset1 e Dataset2, no escopo deste trabalho não houve intenção de identificar quais atributos mais contribuem para o processo de classificação, essa tarefa de seleção de característica é realizada de forma automática pela rede CNN utilizada. Considerando o tradeoff acurácia X escalabilidade, um desafio desse estudo será reduzir o conjunto de atributos sem comprometer substancialmente o desempenho do sistema. Métodos de otimização de modelos profundos como quantização podem ser empregados neste sentido.

6 CONCLUSÕES E TRABALHOS FUTUROS

Este trabalho demonstrou a viabilidade de uma nova abordagem para autenticação contínua de usuários utilizando dados de contadores de desempenho de sistemas operacionais. Nos resultados apresentados, nenhum genuíno foi bloqueado indevidamente e todos os impostores foram detectados com no mínimo de três ciclos de coleta realizadas quando aplicado teste do cenário interno. Entretanto, pensar em possíveis avanços em estudos com autenticação contínua usando contadores de desempenho do sistema operacional requer algumas considerações:

1. Extração Automática de Características

   Dado o grande número de contadores de desempenho existentes nos sistemas operacionais modernos, identificar os mais relevantes para classificação dos usuários genuínos e impostores é uma tarefa desafiadora. Embora este trabalho delegue a tarefa de seleção de característica a uma rede CNN, que comprovamente foi eficiente, conforme resultados obtidos, a identificação das caracteristicas mais relevantes possibilitaria minimizar os custos de processamento, memória e tempo necessários para gerar o modelo de autenticação. Entretanto, não estava no escopo deste trabalho a seleção dos atributos mais significativos para classificação.

2. Avaliação do Uso de um Modelo de Confiança

   Refletindo sobre o processo de avaliação empregado, o nível de confiança, e comparando com outros estudos que avaliam a taxa de falso positivo e de falso negativo, observa-se que utilizar a métrica de nível de confiança agrega ao estudo uma maior robustez e estabilidade. Uma vez que um mesmo usuário pode, eventualmente, fugir de seu padrão de comportamento, caso a avaliação ocorresse somente com os métodos tradicionais, certamente estes desvios implicariam em

situação de falso negativo.

Mesmo que as taxas de falso positivo e taxas de falso negativo não sejam estatisticamente relevantes, para autenticação contínua, pode propiciar bloqueios desnecessários para um usuário genuíno e permitir a atuação de atividades de um usuário impostor.

3. Contadores de Desempenho dos Sistemas Operacionais

   Os dados de movimento de mouse e dinâmica de uso de teclado são coletados após a ação direta e voluntária de um usuário que está interagindo com o sistema. Já os contadores de desempenho do sistema operacional, são coletados periodicamente independente da ação direta e voluntária do usuário, e foi objetivo deste estudo avaliar se através dos contadores de desempenho, pode-se identificar padrões comportamentais dos usuários a partir dos efeitos que os atos ou ações voluntárias geram para os inúmeros componentes de um sistema operacional.

   Com o experimento 2 levantou-se um questionamento se o que estava sendo reconhecido era o ecossistema computacional ou o seu usuário, o que levou ao experimento 3. A padronização do que chamamos de ecossistema no aspecto de hardware, sistema operacional e conjunto de softwares instalados, leva a crer que cada usuário utiliza o ecossistema de forma a permitir criar uma assinatura única, seja através de ações e atos voluntários interativos ou pela forma o utiliza e escolhe os programas, números de janelas abertas, uso de programas que exijam mais ou menos atividades de rede e disco, entre outros inúmeros aspectos.

4. Modelo de Autenticação Contínua

   O modelo de rede neural profunda DEEPCONVLSTM, proposta por Ordóñez [2016], já havia demonstrado bons resultados para tratamento de sinais. Os resultados obtidos neste estudo apontam que a escolha do classificador foi assertiva do ponto de vista da capacidade da rede de efetuar a autenticação contínua reconhecendo a biometria comportamental dos usuários.

   Considerações são necessárias quanto a escalabilidade do modelo de aprendizado usando redes neurais profundas. Uma alternativa apresentada é a redução do universo de características apresentadas à rede. Outra alternativa é melhorar o hardware utilizado nos experimentos, em especial quanto a qualidade da GPU empregada.

6.1 Contribuições

As principais contribuições deste trabalho são:

1. Desenvolvimento de duas bases de dados (datasets) extraídas a partir de contadores de desempenho do sistema operacional. O primeiro dataset contém x amostras referente a 24 horas de coleta de dados de 26 usuários. Os computadores utilizados por esses usuários possuíam características de hardware e software distintos. O segundo dataset contém x amostras referente a horas de coletas para cada usuário (37 usuários no total). Neste dataset, os 37 computadores possuíam características de hardware e software semelhantes.

2. Implementação de uma arquitetura de rede profunda baseada em camadas de convolução e de recorrência. Os dados dos contadores de desempenho são coletados em períodos sucessivos de tempo e, portanto, são caracterizados como uma série temporal.

3. Adaptação do modelo de confiança proposto por Mondal e Bours [2015], que havia sido desenvolvido para estudo autenticação contínua baseada na dinâmica de mouse, para o estudo de contadores de desempenho do sistema operacional.

4. As pesquisas deste estudo geraram publicação do artigo "Autenticação contínua de usuários utilizando contadores de desempenho do sistema operacional"que foi apresentado no XIX Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2019).

6.2 Direções futuras

Como trabalhos futuros ou variações deste trabalho podem ser elencados:

1. Estudos para reduzir a quantidade de atributos utilizados, assim é necessário identificar quais dentre os contadores de desempenho coletados são mais efetivos para propiciar uma melhor classificação.

2. Desenvolver um experimento onde os dados sejam coletados em um mesmo computador e o processo de teste de autenticação contínua possa ser efetuado em tempo real.

3. Alguns trabalhos de detecção de malware a partir da análise comportamental de contadores de desempenho já foram desenvolvidos, entretanto cabe estudar o emprego da rede DEEPCONVLSTM para esta finalidade;

REFERÊNCIAS BIBLIOGRÁFICAS

Akash Sanghi, Y.D.S. Arya (2017) “Survey, Applications and Security of Keystroke Dynamics for User Authentication“. International Journal of Computer & Mathematical Sciences (IJCMS), ISSN 2347 – 8527, Volume 6, Issue 9 September 2017.

Kyle O. Bailey, James S. Okolica, Gilbert L. Peterson (2014) "User identification and authentication using multi-modal behavioral biometrics", Computers & Security, Volume 43, Páginas 77-89, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2014.03.005.

Bours, P. e Barghouthi, H. (2009). “Continuous Authentication using Biometric Keystroke Dynamics”. The Norwegian Information Security Conf. (NISK), Páginas 1–12.

Bours, P. (2012). “Continuous keystroke dynamics: A different perspective towards biometric evaluation”. Information Security Technical Report, Volume 17, Issues 1–2, Páginas 36-43, ISSN 1363-4127, https://doi.org/10.1016/j.istr.2012.02.001.

Cai Z., Shen, C. e Guan, X. (2014) “Mitigating Behavioral Variability for Mouse Dynamics: A Dimensionality-Reduction-Based.”, IEEE Transactions on Human-Machine Systems, Volume 44, Páginas 244 –255.

Chen, A., Brahma, P., Wu, D. O., Ebner, N., Matthews, B., Crandall, J., Xuetao, W., Faloustsos, M. e Oliveira, D. (2016) “Cross-layer personalization as a first-class citizen for situation awareness and computer infrastructure security”, Proceedings of the 2016 New Security Paradigms Workshop (NSPW), Páginas 23-35, https://doi.org/10.1145/3011883.3011888.

Deutschmann, I. e Lindholm, J. (2013) “Behavioral biometrics for DARPA’s Active Authentication program”. International Conference of the BIOSIG Special Interest Group (BIOSIG), Darmstadt, Germany, Páginas 1-8.

Duda, Richard O. e Stork, David G. (2001) "Pattern classification 2nd edition". New York, USA: John Wiley&Sons.

Feher, C., Elovici, Y., Moskovitch, R., Rokach, L. e Schclar, A.. (2012). “User identity verification via mouse dynamics”. Information Sciences, Volume 201, Páginas 19-36, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2012.02.066.

Fridman, L., Stolerman, A., Acharya, S., Brennan, P., Juola, P., Greenstadt, R., e Kam, M. (2015) “Multi-modal decision fusion for continuous authentication”, Computers & Electrical Engineering, Volume 41, 2015, Páginas 142-156, ISSN 0045-7906, https://doi.org/10.1016/j.compeleceng.2014.10.018.

Gunetti, D. e Picardi, C..(2005) “Keystroke analysis of free text”. Association for Computing Machinery (ACM) Trans. Inf. Syst. Secur, Volume 8, Número 3, Páginas 312–347, ISSN 1094-9224, https://doi.org/10.1145/1085126.1085129.

Hochreiter, S., e Schmidhuber, J. (1997) “Long Short-Term Memory”, Neural Computation, Volume 9, Número 8, Páginas 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735

Jagadeesan, H. e Hsiao, Michael S.. (2009). “A novel approach to design of user re-authentication systems”. IEEE 3rd International Conference on Biometrics: Theory, Applications and Systems (BTAS), Páginas 1-6, DOI 10.1109/BTAS.2009.5339075

Leggett, J., Williams, G., Usnick, M. e Longnecker M.. (1991). “Dynamic identity verification via keystroke characteristics”. International Journal of Man-Machine Studies, Volume 35, Número 6, Páginas 859-870, ISSN 0020-7373, https://doi.org/10.1016/S0020-7373(05)80165-8.

Locklear, Hilbert e Brizan, David Guy. (2014) "Continuous authentication with cognition-centric text production and revision features". IEEE International Joint Conference on Biometrics, Páginas 1-8, DOI: 10.1109/BTAS.2014.6996227

Malatras, A., Geneiatakis, D., e Vakalis, I. (2016) “On the efficiency of user identification: a system-based approach”, International Journal of Information Security, Volume 16, Número 6, Páginas 653–671,https://doi.org/10.1007/s10207-016-0340-2.

MICHAELIS. Moderno Dicionário da Língua Portuguesa. Disponível em: <http://michaelis.uol.com.br>. Acesso em: 10 abr. 2020.

Mondal, S., e Bours, P. (2015) “A computational approach to the continuous authentication biometric system”, Information Sciences, Volume 304, Páginas 28-53, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2014.12.045.

Mondal, S., e Bours, P. (2016) “Combining keystroke and mouse dynamics for continuous user authentication and identification”, IEEE International Conference on Identity, Security and Behavior Analysis (ISBA), Páginas 1-8, DOI: 10.1109/ISBA.2016.7477228.

Mondal, S., e Bours, P. (2017) “A study on continuous authentication using a combination of keystroke and mouse biometrics”,Neurocomputing, Volume 230, Páginas 1-22, ISSN 0925-2312, https://doi.org/10.1016/j.neucom.2016.11.031.230.

Nakkabi, Y., Traoré, I. e Ahmed, A., (2010). “Improving mouse dynamics biometric performance using variance reduction via extractors with separate features”, IEEE Trans. Syst. Man Cybern. – Part A: Systems and Humans, Volume 40, Número 6, Páginas 1345-1353, DOI: 10.1109/TSMCA.2010.2052602.

Ordóñez, F. J., e Roggen, D. (2016) "Deep Convolutional and LSTM Recurrent Activity Recognition". Sensors, Páginas 1-25.

Ouch, R., Garcia-zapirain, B. e Yampolskiy, R. (2017). “Multimodal Biometric Systems: a systematic review”, IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Páginas 439-444, DOI: 10.1109/ISSPIT.2017.8388683.

Palaz, D., Magimai.-Doss, M. e Collobert, R. "Analysis of CNN-based Speech Recognition System using Raw Speech as Input". In Proceedings of the 16th Annual Conference of International Speech Communication Association (Interspeech), Páginas 11–15.

Piera, N. e Aguilar, J. (1991). “Controlling selectivity in nonstandard pattern recognition algorithms” IEEE Transactions on Systems, Man, and Cybernetics, Volume 21, Número 1, Páginas 71-82, DOI: 10.1109/21.101138.

Revett, Kenneth. (2009). “A Bioinformatics Based Approach to User Authentication via Keystroke Dynamics”. International Journal of Control, Automation and Systems, Volume 7, Páginas 7-15, https://doi.org/10.1007/s12555-009-0102-2.

Ronao, C. A. e Cho, S.. (2016). “Human activity recognition with smartphone sensors using deep learning neural networks”. Expert Systems with Applications. Volume 57, Páginas 235-244.

Shen, C., Cai, Z. and Guan, X. (2012) "Continuous authentication for mouse dynamics: a pattern-growth approach". EEEIFIP International Conference on Dependable

Systems and Networks (DSN 2012), Páginas 1-12, DOI: 10.1109/DSN.2012.6263955.

Song, Y., Salem, B., Hershkop, S., e Stolfo, S. J. (2013) “System level user behavior biometrics using Fisher features and Gaussian mixture models”. EEE Security and Privacy Workshops, Páginas 52-59, DOI: 10.1109/SPW.2013.33

Sun, Y., Ceker, H. e Upadhyaya, S.. (2017). “Shared keystroke dataset for continuous authentication”. EEE International Workshop on Information Forensics and Security (WIFS), Páginas 1-6, DOI: 10.1109/WIFS.2016.7823894.

Vural, E., Huang, J., Hou, D. e Schuckers, S.. (2014). “Shared Research Dataset to Support Development of Keystroke Authentication”. IEEE International Joint Conference on Biometrics, Páginas 1-8, DOI: 10.1109/BTAS.2014.6996259.

ANEXO I

Relação de atributos que compõem Dataset 1 e Dataset2, referente a contadores de desempenho do sistema operacional.

Contador de Desempenho	DataSet1	DataSet2
Adaptador de Rede Bytes enviados por segundo		Sim
Adaptador de Rede Bytes recebidos por segundo		Sim
Adaptador de Rede Pacotes enviados de difusão ponto-a-ponto por segundo		Sim
Adaptador de Rede Pacotes enviados por segundo		Sim
Adaptador de Rede Pacotes por segundo		Sim
Adaptador de Rede Pacotes recebidos de difusão não-ponto-a-ponto por segundo		Sim
Adaptador de Rede Pacotes recebidos de difusão ponto-a-ponto por segundo		Sim
Adaptador de Rede Pacotes Recebidos por segundo		Sim
Adaptador de Rede Total de bytes por segundo		Sim
Arquivo de paginação (Total) Percentual uso		Sim
Atividade da Placa de Interface de Rede Física (Intel Ethernet Connection (7) I219-LM) Transições de Baixo Consumo de Energia Tempo de Vida		Sim
Atividade da Placa de Interface de Rede Física (Intel Ethernet Connection (7) I219-LM) Estado de Energia do Dispositivo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Chamadas de Solicitação de Envio por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Chamadas de Conclusão de Envio por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Chamadas de Lista de Agrupamento de Criação de Dispersão por segundo		Sim

Contador de Desempenho	Dataset1	Dataset2
Atividade da Placa de Interface de Rede por Processador (Total) DPCs Adiados por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) DPCs enfileirados por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Interrupções por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Pacotes de Conclusão de Envio por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Pacotes Enviados por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Pacotes Recebidos por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Pacotes Retornados por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Retornar Chamadas de Pacote por segundo		Sim
Atividade da Placa de Interface de Rede por Processador (Total) Receber Indicações por segundo		Sim
Atividade de Disco FileSystem (Total) Bytes Gravados pelo FileSystem		Sim
Atividade de Disco FileSystem (Total) Bytes Lidos pelo FileSystem		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Compilação de Scatter por Gather por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Envio Concluído de NDIS por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Envio Concluído de Pilha por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Envio de Miniporta por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Envio de NDIS por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Indicação de Recepção de Pilha por segundo		Sim

Contador de Desempenho	Dataset1	Dataset2
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Indicação de Recepção de NDIS por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Interrupção por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Latência DPC de Interrupção por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Pacotes de Retorno de Miniporta por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos de Pacotes de Retorno de NDIS por segundo		Sim
Ciclos de Atividade de Rede por Processador (Total) Ciclos DPC de Interrupção por segundo		Sim
Informações do Processador (Total) DPCs Enfileirados por segundo	Sim	Sim
Informações do Processador (Total) Eventos de Interrupção Ociosa por segundo		Sim
Informações do Processador (Total) Interrupções do Relógio por segundo		Sim
Informações do Processador (Total) Interrupções por segundo	Sim	Sim
Informações do Processador (Total) Media de Tempo Ocioso		Sim
Informações do Processador (Total) Percentual de Desempenho do Processador		Sim
Informações do Processador (Total) Percentual de Frequência Máxima		Sim
Informações do Processador (Total) Percentual de Tempo Ocioso	Sim	Sim
Informações do Processador (Total) Percentual de Utilitário do Processador		Sim
Informações do Processador (Total) Percentual de Utilitário Privilegiado		Sim
Informações do Processador (Total) Percentual Tempo C1	Sim	Sim
Informações do Processador (Total) Percentual Tempo C2		Sim
Informações do Processador (Total) Percentual Tempo C3		Sim
Informações do Processador (Total) Percentual Tempo de DPC	Sim	Si

Contador de Desempenho	Dataset1	Dataset2
Informações do Processador (Total) Percentual Tempo de Interrupção	Sim	Sim
Informações do Processador (Total) Percentual tempo de prioridade	Sim	Sim
Informações do Processador (Total) Percentual Tempo do Processador	Sim	Sim
Informações do Processador (Total) Percentual Tempo do Usuário	Sim	Sim
Informações do Processador (Total) Percentual Tempo Privilegiado	Sim	Sim
Informações do Processador (Total) Taxa de DPC	Sim	Sim
Informações do Processador (Total) Transições C1 por segundo	Sim	Sim
Informações do Processador (Total) Transições C2 por segundo		Sim
Informações do Processador (Total) Transições C3 por segundo		Sim
Interface de rede Bytes enviados por segundo	Sim	Sim
Interface de rede Bytes recebidos por segundo	Sim	Sim
Interface de rede Largura de banda atual		Sim
Interface de rede Pacotes enviados de difusão não-ponto-a-ponto por segundo	Sim	Sim
Interface de rede Pacotes enviados de difusão ponto-a-ponto por segundo	Sim	Sim
Interface de rede Pacotes enviados por segundo	Sim	Sim
Interface de rede Pacotes por segundo	Sim	Sim
Interface de rede Pacotes enviados de difusão não-ponto-a-ponto por segundo	Sim	Sim
Interface de rede Pacotes recebidos de difusão ponto-a-ponto por segundo	Sim	Sim
Interface de rede Pacotes Recebidos por segundo	Sim	Sim
Interface de rede Total de bytes por segundo	Sim	Sim
LogicalDisk (Total) Bytes de disco por segundo	Sim	Sim
LogicalDisk (Total) Bytes de gravação de disco por segundo	Sim	Sim
LogicalDisk (Total) Bytes de leitura de disco por segundo	Sim	Sim
LogicalDisk (Total) Comprimento da fila de disco atual	Sim	Sim
LogicalDisk (Total) Comprimento médio da fila de disco	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
LogicalDisk (Total) Comprimento médio da fila de gravação de disco	Sim	Sim
LogicalDisk (Total) Comprimento médio da fila de leitura de disco	Sim	Sim
LogicalDisk (Total) E por segundo dividida por segundo	Sim	Sim
LogicalDisk (Total) Gravações de disco por segundo	Sim	Sim
LogicalDisk (Total) Leituras de disco por segundo	Sim	Sim
LogicalDisk (Total) Media de bytes de disco por gravação	Sim	Sim
LogicalDisk (Total) Media de bytes de disco por leitura	Sim	Sim
LogicalDisk (Total) Media de bytes de disco por transferência	Sim	Sim
LogicalDisk (Total) Media de disco por gravação	Sim	Sim
LogicalDisk (Total) Media de disco por leitura	Sim	Sim
LogicalDisk (Total) Media de disco por transferência	Sim	Sim
LogicalDisk (Total) Megabytes livres	Sim	Sim
LogicalDisk (Total) Percentual de espaço livre	Sim	Sim
LogicalDisk (Total) Percentual tempo de disco	Sim	Sim
LogicalDisk (Total) Percentual tempo de gravação de disco	Sim	Sim
LogicalDisk (Total) Percentual tempo de leitura de disco	Sim	Sim
LogicalDisk (Total) Percentual tempo ocioso	Sim	Sim
LogicalDisk (Total) Transferências de disco por segundo	Sim	Sim
Memória de alocação de pool não paginável	Sim	Sim
Memória de alocação de poll paginável	Sim	Sim
Memoria Bytes confirmados	Sim	Sim
Memoria Bytes da Lista de Páginas Livre e Zero	Sim	Sim
Memoria Bytes da Lista de Páginas Modificadas	Sim	Sim
Memoria Bytes de cache	Sim	Sim
Memoria Bytes de pool não-paginável	Sim	Sim
Memoria Bytes de pool paginável	Sim	Sim
Memoria Bytes de Prioridade Normal de Cache em Espera	Sim	Sim
Memoria Bytes de Reserva de Cache em Espera	Sim	Sim
Memoria Bytes disponíveis		Sim
Memoria Bytes Principais de Cache em Espera		Sim
Memoria Bytes residentes de cache do sistema	Sim	Sim
Memoria Bytes residentes de driver do sistema	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
Memoria Bytes residentes de pool paginável	Sim	Sim
Memoria Copias de gravação por segundo	Sim	Sim
Memoria de No NUMA (Total) MBytes de Lista de Espera		Sim
Memoria de No NUMA (Total) MBytes de Lista de Paginas Livre e Zero		Sim
Memoria de No NUMA (Total) MBytes Disponíveis		Sim
Memoria Entrada de paginas por segundo	Sim	Sim
Memoria Entradas livres de tabela de paginação do sistema	Sim	Sim
Memoria Falhas de cache por segundo	Sim	Sim
Memoria Falhas de demanda zero por segundo	Sim	Sim
Memoria Falhas de paginas por segundo	Sim	Sim
Memoria Falhas de transição por segundo	Sim	Sim
Memoria Kbytes disponíveis	Sim	Sim
Memoria Gravações de pagina por segundo		Sim
Memoria Leituras de pagina por segundo	Sim	Sim
Memoria MBytes disponíveis	Sim	Sim
Memoria Páginas de transição realocadas por segundo		Sim
Memoria Páginas por segundo	Sim	Sim
Memoria Percentual bytes confirmados em uso	Sim	Sim
Memoria Saída de páginas por segundo		Sim
Memoria Total de bytes de driver do sistema	Sim	Sim
PhysicalDisk (Total) Bytes de disco por segundo	Sim	Sim
PhysicalDisk (Total) Bytes de gravação de disco por segundo	Sim	Sim
PhysicalDisk (Total) Bytes de leitura de disco por segundo	Sim	Sim
PhysicalDisk (Total) Comprimento da fila de disco atual	Sim	Sim
PhysicalDisk (Total) Comprimento médio da fila de disco	Sim	Sim
PhysicalDisk (Total) Comprimento médio da fila de gravação de disco	Sim	Sim
PhysicalDisk (Total) Comprimento médio da fila de leitura de disco	Sim	Sim
PhysicalDisk (Total) E por segundo dividida por segundo	Sim	Sim
PhysicalDisk (Total) Gravações de disco por segundo	Sim	Sim
PhysicalDisk (Total) Leituras de disco por segundo	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
PhysicalDisk (Total) Media de bytes de disco por gravação	Sim	Sim
PhysicalDisk (Total) Media de bytes de disco por leitura	Sim	Sim
PhysicalDisk (Total) Media de bytes de disco por transferência	Sim	Sim
PhysicalDisk (Total) Media de disco por gravação	Sim	Sim
PhysicalDisk (Total) Media de disco por leitura	Sim	Sim
PhysicalDisk (Total) Media de disco por transferência	Sim	Sim
PhysicalDisk (Total) Percentual tempo de disco	Sim	Sim
PhysicalDisk (Total) Percentual tempo de gravação de disco	Sim	Sim
PhysicalDisk (Total) Percentual tempo de leitura de disco	Sim	Sim
PhysicalDisk (Total) Percentual tempo ocioso	Sim	Sim
PhysicalDisk (Total) Transferências de disco por segundo	Sim	Sim
Processador (Total) DPCs enfileirados por segundo	Sim	Sim
Processador (Total) Interrupções por segundo	Sim	Sim
Processador (Total) Percentual tempo C1	Sim	Sim
Processador (Total) Percentual tempo C2		Sim
Processador (Total) Percentual tempo C3		Sim
Processador (Total) Percentual tempo de DPC	Sim	Sim
Processador (Total) Percentual tempo de interrupção	Sim	Sim
Processador (Total) Percentual tempo de processador	Sim	Sim
Processador (Total) Percentual tempo de usuário	Sim	Sim
Processador (Total) Percentual tempo ocioso	Sim	Sim
Processador (Total) Percentual tempo privilegiado	Sim	Sim
Processador (Total) Taxa de DPC	Sim	Sim
Processador (Total) Transições C1 por segundo	Sim	Sim
Processador (Total) Transições C2 por segundo		Sim
Processador (Total) Transições C3 por segundo		Sim
Processo (Total) Bytes de arquivo de paginação	Sim	Sim
Processo (Total) Bytes de dados de ES por segundo	Sim	Sim
Processo (Total) Bytes de gravação de E por segundo por segundo	Sim	Sim
Processo (Total) Bytes de leitura de E por segundo por segundo	Sim	Sim
Processo (Total) Bytes de pool não-paginável	Sim	Sim
Processo (Total) Bytes de pool paginável	Sim	Sim
Processo (Total) Bytes virtuais	Sim	Sim
Processo (Total) Conjunto de trabalho	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
Processo (Total) Conjunto de trabalho - particular	Sim	Sim
Processo (Total) Contagem de threads	Sim	Sim
Processo (Total) Falhas de páginas por segundo	Sim	Sim
Processo (Total) Número de Identificadores	Sim	Sim
Processo (Total) Operações de dados de ES por segundo	Sim	Sim
Processo (Total) Operações de gravação de ES por segundo	Sim	Sim
Processo (Total) Operações de leitura de ES por segundo	Sim	Sim
Processo (Total) Outras operações de ES por segundo	Sim	Sim
Processo (Total) Outros bytes de E por segundo por segundo	Sim	Sim
Processo (Total) Percentual tempo de processador	Sim	Sim
Processo (Total) Percentual tempo de usuário	Sim	Sim
Processo (Total) Percentual tempo privilegiado	Sim	Sim
Processo (Total) Pico de bytes de arquivo de paginação	Sim	Sim
Processo (Total) Pico de bytes virtuais	Sim	Sim
Processo (Total) Pico do conjunto de trabalho	Sim	Sim
Sincronização (Total) Aquisições de AcqExclLiteI Exclusivas Recursivas no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqExclLiteI no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqShrdLite Compartilhadas Recursivas no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqShrdLite Exclusivas Recursivas no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqShrdLite no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqShrdStarveExcl Compartilhadas Recursivas no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de AcqShrdStarveExcl no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aquisições de Spinlock por segundo	Sim	Sim
Sincronização (Total) Aquisições Totais de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Aumento de Proprietário Exclusivo do Recurso Execução por segundo	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
Sincronização (Total) Aumento de Proprietários Compartilhados do Recurso Execução por segundo		Sim
Sincronização (Total) Contenção de AcqExclLiteI no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Contenção de AcqShrdLite no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Contenção de AcqShrdStarveExcl no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Contenções de Spinlock por segundo	Sim	Sim
Sincronização (Total) Contenções Totais de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Conversão Total de Recurso Executivo Exclusivo para Compartilhamento por segundo	Sim	Sim
Sincronização (Total) Definição de Proprietário de Ponteiro Compartilhado (Proprietário Existente) do Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Definição de Proprietário de Ponteiro Compartilhado (Novo Proprietário) do Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Definição de Proprietário de Ponteiro Exclusivo do Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Exclusão Total de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Inicialização Total de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Interrupções de Software de Envio IPI por segundo	Sim	Sim
Sincronização (Total) Liberações Compartilhadas Totais de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Liberações Exclusivas Totais de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Pedidos de Rotina de Envio IPI por segundo	Sim	Sim

Contador de Desempenho	Dataset1	Dataset2
Sincronização (Total) Pedidos por Difusão de Envio IPI por segundo	Sim	Sim
Sincronização (Total) Reinicialização Total de Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Rotações de Spinlock por segundo	Sim	Sim
Sincronização (Total) Sem espera de AcqExclLiteI no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Sem espera de AcqShrdLite no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Tentativas de AcqExclLiteI no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Tentativas de AcqShrdLite no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Tentativas de AcqShrdStarveExcl no Recurso Execução por segundo	Sim	Sim
Sincronização (Total) Tentativas de AcqShrdWaitForExcl no Recurso Execução por segundo		Sim
Total de Contadores de Desempenho	159	218