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

AUTENTICAÇÃO DE ALUNOS UTILIZANDO BIOMETRIA COMPORTAMENTAL EM AMBIENTES JUIZ ON-LINE

RONEM MATOS LAVAREDA FILHO

AUTENTICAÇÃO DE ALUNOS UTILIZANDO BIOMETRIA COMPORTAMENTAL EM AMBIENTES JUIZ ON-LINE

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: Juan Gabriel Colonna
Co-orientador: David Fernandes

Manaus - AM

Março de 2022

Ficha Catalográfica

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

L396a

Lavareda Filho, Ronem Matos
Autenticação de alunos utilizando biometria comportamental em ambientes Juiz On-line / Ronem Matos Lavareda Filho . 2022
64 f.: il.; 31 cm.

Orientador: Juan Gabriel Colonna
Coorientador: David Braga Fernandes de Oliveira
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Autenticação. 2. Biometria Comportamental . 3. Juiz On-line. 4. Alunos. 5. Deep Learning. I. Colonna, Juan Gabriel. II. Universidade Federal do Amazonas III. Título

FOLHA DE APROVAÇÃO

"Autenticação de alunos utilizando biometria comportamental em ambiente de Juiz On-line"

RONEM MATOS LAVAREDA FILHO

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

Prof. Juan Gabriel Colonna - PRESIDENTE

Prof. Raimundo da Silva Barreto - MEMBRO INTERNO

Prof. Mário Salvatierra Júnior - MEMBRO EXTERNO

Manaus, 07 de Março de 2022

Agradecimentos

A Deus, que é o dono de tudo. Devo a Ele a oportunidade que tive de chegar aonde cheguei. Deus, tenho poucas palavras para o Senhor, mas são as mais sinceras. Muito obrigado.

Agradeço à minha família, minha querida esposa Jheinne, pelo amor, carinho e principalmente pela paciência e apoio incondicional nos meus momentos de incertezas relacionados a pesquisa. Poder compartilhar a vida com você é uma bênção. Te amo. A minha princesa Analu, filha amada, você é o melhor presente que já recebi na vida. Amo você.

Aos meus pais, por estarem sempre ao meu lado, pelo esforço que tiveram ao longo de todo esse tempo para me manter e ensinar da melhor forma possível. Obrigado por sonharem junto comigo, hoje esse sonho se tornou realidade, amo vocês. Aos meus irmãos Rízia e Renner pelo suporte e apoio, vocês fazem parte dessa conquista.

Agradeço ao meu orientador, o professor Dr. Juan Colonna, pelo incentivo, apoio e por ter guiado-me da melhor forma possível na pesquisa, suas constantes contribuições me trouxeram até aqui, muito obrigado. Também ao meu co-orientador, professor Dr. David Fernandes, pelo suporte que foi primordial para a conclusão dessa jornada.

Aos meus amigos de laboratório da UFAM, pela ajuda e companheirismo, a ajuda de vocês foi essencial para que pudesse chegar ao final.

Por fim, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) pelo financiamento do presente trabalho - Código de Financiamento 001.

"A Deus, que se mostrou Criador, que foi Criativo. Seu fôlego de vida em mim me foi sustento e me deu coragem para questionar realidades e propor sempre um novo mundo de possibilidades"

(Ronem Lavareda)

Resumo

Em ambientes juízes on-line, a autenticação do aluno normalmente é feita apenas no início da sessão de *login* mediante a digitação de uma senha, que apesar de ser o método padrão, apresenta vulnerabilidades, pois, posteriormente, não é verificada nenhuma outra vez a genuinidade do usuário logado. Desta forma, o compartilhamento de informações de *login* e senha ocasiona problemas de autenticidade ou falsificação de identidade. No caso de exercícios e avaliações feitas em ambientes como esse, há a necessidade da verificação não-intrusiva da identidade dos estudantes durante a submissão das atividades, não apenas no *login*.

Nesta pesquisa, está sendo proposto e validado um método de autenticação dos alunos em ambientes Juiz on-line utilizando biometria comportamental, mas especificamente a dinâmica de digitação, essa técnica não exigir nenhuma ação explícita do usuário, além do baixo custo, pois é necessário somente um teclado convencional para a aquisição de informações, tornando o custo relativamente baixo. Para isso, foi projetada uma arquitetura de CNN 1D Siamesa que aprendeu, a partir dos dados brutos do ritmo da digitação dos usuários, a representação necessária para o reconhecimento dos alunos, dispensando, desta forma, a extração manual de *features*, no qual é um processo que exige muito tempo e a participação de especialistas. Para validar esse método, foram utilizados dados da dinâmica da digitação de 42 alunos no ambiente juiz on-line CodeBench. Em seguida, foram realizados vários experimentos onde demonstraram a eficácia do método proposto.

Com a utilização deste método, como um módulo adicional de segurança em ambientes de programação, foi comprovado através de experimentos que é possível diminuir as ocorrências prováveis de fraude por parte de estudantes em exercícios e avaliações de programação. Desta forma, a aplicação do método proposto fornece às instituições e professores uma segurança complementar contra fraude. Além disso, é importante para o aluno, uma vez que testifica sua integridade acadêmica em todas as atividades. De acordo com a nossa revisão bibliográfica, o presente trabalho é o primeiro método de autenticação de alunos em ambientes juiz on-line.

Palavras-chave: Autenticação, Biometria Comportamental, Juiz On-line, Alunos, *Deep Learning*.

Abstract

In online judge environments, student authentication is usually done only at the beginning of the login session by typing a password, which, despite being the standard method, presents vulnerabilities, since, subsequently, the authenticity of the logged in user. In this way, sharing login and password information causes authenticity issues or identity spoofing. In the case of exercises and assessments carried out in environments like this, there is a need for non-intrusive verification of the students' identity during the submission of the activities, not just at login. In this research, a method of authentication of students in online judge environments is being proposed and validated using behavioral biometrics, but specifically typing dynamics, this technique does not require any explicit action from the user, in addition to the low cost, as it is only necessary a conventional keyboard for acquiring information, making the cost relatively low. For this, a Siamese 1D CNN architecture was designed that learned, from the raw data of the users' typing rhythm, the representation necessary for the recognition of the students, thus dispensing with the manual extraction of features, in which it is a process that requires a lot of time and the participation of experts. To validate this method, data from the typing dynamics of 42 students in the CodeBench online judge environment were used. Then, several experiments were carried out demonstrating the effectiveness of the proposed method.

With the use of this method, as an additional module of security in programming environments, it has been proven through experiments that it is possible to reduce the probable occurrences of fraud on the part of students in programming exercises and evaluations. In this way, the application of the proposed method will provide institutions and teachers with additional security against fraud. In addition, it is important to the student as it testifies to their academic integrity in all activities. According to our literature review, the present work is the first method of authenticating students in online judge environments.

Keywords: Authentication, Behavioral Biometrics, Online Judge, Studentus, Deep Learning.

Lista de Figuras

2.1 Interface do juiz on-line Codebench. 9
2.2 Classificação Biométrica. 10
2.3 Tempos de Pressionamento e Latências. 14
2.4 Função *ReLU* - Ilustração Gráfica. 18
2.5 Arquitetura de uma rede neural siamesa clássica 20
2.6 Triplet loss clássica 21
2.7 Curva ROC 22

4.1 Esquema da metodologia proposta. 32
4.2 Janela deslizando. 34
4.3 Arquitetura da rede neural proposta. 36
4.4 Sub-redes CNNs 1D 36
4.5 Pipeline da abordagem proposta. 37
4.6 Abordagem de decomposição um-contra-todos. 38

5.1 Dados brutos da dinâmica de digitação capturados pela IDE do CodeBench representando as *Features* biométricas dos usuários. 41
5.2 *Features* biométricas dos usuários. 41
5.3 Protocolo I. 42
5.4 Protocolo II. 43
5.5 Protocolo III 44
5.6 Protocolo IV 44
5.7 Curva Roc 51
5.8 T-SNE de 10 usuários 51
5.9 T-SNE de 10 usuários (Base de Dados I) 53

Lista de Tabelas

3.1 Sumarização dos trabalhos relacionados. 29

4.1 Estrutura dos dados brutos da dinâmica de digitação 33

5.1 Acurácias e Taxas de Erro (%) alcançadas para diferentes tamanhos de segmentos 46
5.2 Acurácias e Taxas de erro (%) obtidas a partir da quantidade de features . 47
5.3 Acurácias e Taxas de erro (%) obtidas a partir das duas arquiteturas nas duas bases de dados comparadas contra o baseline. 48
5.4 Acurácias e Taxas de Erro (%) alcançadas utilizando diferentes Protocolos de separação de dados 49
5.5 Acurácias e Taxas de Erro (%) alcançadas para diferentes quantidades de atividades utilizadas para treinamento 50
5.6 Acurácias e Taxas de Erro (%) alcançadas para diferentes quantidades de atividades utilizadas para treinamento 52

Lista de Abreviaturas e Siglas

AS - Static Authentication (Autenticação Estática)
AUC - Area Under The Curve (Área sob a curva)
AVAs - Ambientes Virtuais de Aprendizagem
CA - Continuous Authentication (Autenticação Contínua)
CNN - Rede Neural Convolucional
DL - Deep Learning (Aprendizagem Profunda)
EaD - Educação a Distância
EER - Equal Error Rate (Taxa de Erro Igual)
FAR - False Acceptance Rate (Taxa de falsa aceitação)
FRR - False Rejection Rate (Taxa de falsa rejeição)
IDE - Integrated Development Environment (Ambiente de desenvolvimento integrado)
kNN - K nearest neighboors (K vizinhos mais próximos)
LSTM - Long short-term Memory (Memória longa de curto prazo)
MLP - Multi-layer perceptron (Perceptron multicamadas)
RNA - Redes Neurais Artificiais
RNN - Redes Neurais Recorrentes
RNS - Redes Neurais Siamesas
ROC - Receiver Operating Characteristic (Curva de características operacionais)
SBC - Sociedade Brasileira de Computação
SVM - Support Vector Machine (Máquina de vetores de suporte)
TCT - Total de Contas Tentadas
TFR - Total de Falsa Rejeição
UFAM - Universidade Federal do Amazonas

Sumário

Agradecimentos **v**

Resumo **ix**

Abstract **xi**

Lista de Figuras **xiii**

Lista de Tabelas **xv**

Acrónimos e nomenclaturas **xvi**

1 Introdução **1**

1.1 Motivação 3
1.2 Justificativa 4
1.3 Hipótese 5
1.4 Objetivo geral e específicos 5
1.5 Considerações 6
1.6 Organização 6

2 Fundamentos Teóricos **7**

2.1 Juiz On-line 7
2.2 Biometria 8
2.2.1 Autenticação Biométrica 10
2.3 Biometria Comportamental por Dinâmica de Digitação 12
2.4 Aprendizagens de máquina 14
2.4.1 Aprendizagem profunda 16
2.5 Medidas de Desempenho 21
2.6 Considerações finais 23

3 Trabalhos Relacionados **25**

3.1 Dinâmica da digitação em contextos gerais 25
3.1.1 Dinâmica da digitação em cursos on-line 26
3.1.2 Dinâmica da digitação em sistemas Juiz On-line 27
3.2 Dinâmica da digitação com redes siamesas 28
3.3 Discussão 28
3.4 Considerações finais 30

4 Solução Proposta 31
4.1 Visão Geral 31
4.2 Captura de dados 32
4.3 Pré-Processamento 33
4.3.1 Segmentação dos dados 34
4.4 Arquitetura da rede neural convolucional siamesa proposta 34
4.5 Autenticador biométrico 37
4.6 Considerações finais 38

5 Experimentos e Resultados 39
5.1 Base de dados I 39
5.2 Base de dados II 40
5.3 Pré-Processamento 40
5.4 Protocolos de treino e validação 42
5.4.1 Protocolo I 42
5.4.2 Protocolo II 43
5.4.3 Protocolo III 44
5.4.4 Protocolo IV 44
5.5 Configurações dos modelos 45
5.6 Experimentos e Resultados I 46
5.6.1 Experimentos I 46
5.6.2 Experimentos II 47
5.6.3 Experimentos III 47
5.6.4 Experimentos IV 49
5.6.5 Experimentos V 50
5.6.6 Experimentos VI 52
5.7 Considerações finais 53

6 Conclusões 55
6.1 Impactos do trabalho para o âmbito educacional 56
6.2 Limitações do método 56

6.3 Trabalhos futuros 57

Referências Bibliográficas **59**

1

Introdução

Nos últimos anos, cursos de computação atraíram um número significativo de estudantes em todo o mundo [Hao et al., 2019]. Por exemplo, as matrículas em cursos de graduação em Ciência da Computação dobraram no período de 2011 a 2017 em universidades do Brasil [Nunes, 2018]. Dentre as disciplinas ofertadas nesses cursos, as de programação têm se mostrado indispensáveis. O aprendizado dessas disciplinas torna o indivíduo apto a aplicar a lógica de programação na resolução de diversos problemas corriqueiros, fator essencial para prosseguir no curso e na própria carreira profissional [Chaves, 2014].

No cenário de ensino-aprendizagem de programação, a prática e a constante resolução de exercícios são essenciais para a aquisição de conhecimento [Galvão et al., 2016]. Entre as principais tecnologias voltadas para este contexto, destacam-se os sistemas de juízes on-line. Esses são ferramentas de ensino-aprendizagem normalmente adotadas em disciplinas que abordam o ensino introdutório de programação, pois auxiliam na melhoria de habilidades dos alunos iniciantes. Estes sistemas permitem gerar avaliações automáticas para as atividades de programação submetidas pelos alunos. Portanto, esses ambientes são comumente utilizados em concursos de programação [Francisco et al., 2016].

Nesta perspectiva, por serem bastante atrativos, muitos alunos iniciam seus aprendizados de programação nos ambientes juízes on-line [Zhigang et al., 2001]. Dentre as principais características desses sistemas estão o estímulo à autoaprendizagem, as correções automáticas e os *feedbacks* imediatos para aqueles que pretendem aprender a programar. Tanto na modalidade de Educação a Distância (EaD), quanto na modalidade de educação presencial, essas ferramentas normalmente possuem um ambiente de desenvolvimento integrado (*Integrated Development Environment* - IDE) onde os alu-

nos podem codificar suas soluções para os exercícios disponibilizados pelos professores [Gomes et al., 2008].

Entretanto, apesar dos benefícios trazidos pelos sistemas juiz on-line, uma das principais dificuldades acerca de sua adoção é quanto à autenticidade de seus usuários. Segundo Ullah et al. [2019], a maior ameaça ao uso de sistemas que permitem a submissão de qualquer trabalho on-line é a falsificação de identidade (*Impersonation*). Este termo está relacionado ao ato de um usuário fingir ser outra pessoa para fins de fraude, neste caso de identidade.

No caso dos juízes on-line, a autenticação do aluno normalmente é feita apenas no início da sessão de *login* mediante a digitação de uma senha, que apesar de ser o método padrão, apresenta vulnerabilidades [Silva et al., 2019], pois, posteriormente, não é verificada nenhuma outra vez a genuinidade do usuário logado. Desta forma, o compartilhamento de informações de *login* e senha ocasionam problemas de autenticidade ou falsificação de identidade. Por exemplo, algumas pessoas comportam-se de forma desonesta autenticando-se com credenciais de outras pessoas, fingindo serem usuários genuínos, e assim, praticam fraudes. Outros convidam terceiros para fazer seus exercícios e avaliações, visando benefícios próprios [Lavareda Filho et al., 2020].

Deste modo, em ambientes juiz on-line, é um grande desafio verificar à autenticidade dos estudantes que concluem os exercícios e avaliações, garantindo que o usuário que faz as atividades seja a mesma pessoa que recebe crédito pela conclusão. Portanto, saber se o aluno é realmente o usuário legítimo ao realizar as atividades em ambientes de juiz online é essencial para diminuir as possíveis ocorrências de trapaça por parte dos estudantes desonestos [Caldarola & MacNeil, 2009], garantindo às instituições e professores segurança adicional contra fraude. Além disso, é importante para o aluno, uma vez que testifica sua integridade acadêmica em todas as atividades. Portanto, é desejável que este tipo de sistemas garanta que os usuários que realizam as atividades sejam genuínos [Ichihara & Nizam, 2018].

Segundo Muhammad [2018], a autenticação por biometria é a ciência alinhada à tecnologia de autenticação, que tem por finalidade identificar tanto atributos fisiológicos como comportamentais de uma pessoa. Isto é, a autenticação é o processo usado para confirmar a identidade de um usuário [Pisani & Lorena, 2013]. Deste modo, a fim de adicionar mais segurança em ambiente juiz on-line, faz-se com que a utilização de biometrias não intrusivas, seja uma solução apropriada para a autenticação de alunos.

1.1 Motivação

A autenticação biométrica pode ser realizada utilizando biometria fisiológica, do qual realiza a identificação a partir de atributos físicos do usuário, como impressões digitais, reconhecimento facial ou de voz [Quraishi & Bedi, 2018]. Além dessa, existe a biometria comportamental, que está relacionada ao padrão de comportamento de uma pessoa, como por exemplo: dinâmica da digitação (pressionamento de tecla), dinâmica do mouse [Tan et al., 2019], entre outros.

Muitos métodos de autenticação têm adotado a biometria comportamental para resolver problemas de autenticidade em sistemas on-line, pois a mesma não requer uso de hardware adicional para executar a coleta de dados biométricos, cujos atributos coletados podem ser obtidos silenciosamente, de forma transparente, sem atrapalhar o aluno genuíno e sem alertar o aluno trapaceiro que esteja sob avaliação [Mondal & Bours, 2017].

Existem inúmeras técnicas de autenticação baseadas no comportamento do usuário. Dentre essas, a dinâmica de digitação tem sido bastante estudada em diversos contextos [Pisani & Lorena, 2013], inclusive educacional [Young et al., 2019]. Foi demonstrado que cada indivíduo possui uma forma única de digitar [Mondal & Bours, 2017]. Deste modo, levanta-se a hipótese de que se o ritmo de digitação de cada indivíduo é único, então é possível autenticá-lo através dessa característica. Sendo assim, o recurso biométrico utilizado nesta pesquisa será a dinâmica de digitação, especificamente no contexto de desenvolvimento de códigos de computadores. A intuição é que cada pessoa possua um perfil particular de digitação no momento em que está codificando em uma dada linguagem de programação, e que esse perfil pode ser utilizado para identificar o aluno que esteja implementando uma dada solução para um dado exercício dentro da IDE de um ambiente juiz on-line.

Logo, os padrões de digitação podem ser ferramentas essenciais para autenticação de alunos em ambientes juiz on-line. Por exemplo, ainda no *login* inicial do aluno no sistema juiz on-line, suas informações biométricas (tempo de pressionamento, latência entre duas teclas, etc) do ritmo da digitação poderão ser verificadas, evitando assim, problemas de autenticação e falsificação de identidade.

Algumas vantagens quanto a utilização de biometrias comportamentais, mais especificamente a dinâmica de digitação como adicional a segurança em ambientes juiz on-line são apresentadas abaixo:

Maior segurança: o método de autenticação será mais seguro em comparação aos métodos tradicionais que utilizam somente login e senha, uma vez que será adicionado as vantagens das características biométricas comportamentais;

Baixo custo: Será necessário somente um teclado convencional para a aquisição de informações, tornando o custo relativamente baixo comparado aos hardwares responsáveis pela coleta de informações em sistemas de autenticação baseados em biometrias fisiológicas;

Não intrusivo: a coleta de amostras será feita de maneira silenciosa e transparente. Para algumas tecnologias biométricas, o processo de coleta é bastante invasivo. Por exemplo no reconhecimento da íris, quando um escaneamento no olho é realizado utilizando um feixe de luz, fazendo com que muitos de seus usuários sintam-se desconfortáveis.

1.2 Justificativa

Diferentes algoritmos podem ser aplicados no campo da autenticação biométrica. Entre eles estão os algoritmos de aprendizagem profunda (do inglês *Deep Learning - DL*). A principal vantagem de utilizar aprendizagem profunda é a sua capacidade de criar modelos capazes de analisar e aprender o comportamento humano a partir de conjuntos de dados [Chong et al., 2019]. Além disso, esses modelos possuem a característica de possuir múltiplas camadas que aprendem diferentes níveis de representações [Bhanu & Kumar, 2017]. Por exemplo, os modelos da Rede Neural Convolucional (CNN) de uma dimensão (1D) são eficazes para extração de recursos em segmentos de comprimento fixo, extraindo características complexas de cada um dos segmentos do todo (conjunto de dados geral) [Lavareda Filho et al., 2020]. Essa tarefa é essencial para identificação e autenticação do usuário em sistemas juiz on-line. Além de que, as CNN 1D são normalmente mais rápidas e simples para treinar em comparação com Redes Neurais Recorrentes (RNN) [Bradbury et al., 2016]. Essas redes, quando utilizadas como subredes (ou rede base) em Redes Neurais Siamesas (RNS), na função de extratores de *features*, obtiveram bons resultados [Centeno et al., 2018].

As redes RNS, de outro modo, são comumente desenvolvidas para determinar se pares de entradas de dados são ou não da mesma classe, como identificação facial ou autenticação de usuários de *smartphone* [Acien et al., 2021]. Uma vez que o modelo é treinado, as *features* geradas podem ser usadas para identificar usuários nunca vistos antes [Giot & Rocha, 2019]. Além disso, as RNS alcançam bons resultados inclusive quando a quantidade de dados por usuário é limitada [Acien et al., 2020]. Uma vez que a rede RNS foi treinada, o modelo de extração de *features* pode ser utilizado em uma configuração conhecida como *One-shot learning* (Aprendizado Único) na literatura científica, capaz de extrair *features* relevantes de usuários nunca vistos pelo modelo

durante o treinamento.

Portanto, o presente trabalho propõe o desenvolvimento e avaliação de um método de autenticação baseado em biometria comportamental utilizando aprendizagem profunda, com intuito de autenticar de forma não intrusiva alunos em ambientes juiz on-line. Para isso, será projetado um modelo de rede neural convolucional siamesa que quantifica a similaridade entre pares de usuários, e com isso, é aplicado um critério de autenticação. Para validar esse método, será utilizado dados da dinâmica da digitação de 42 alunos no ambiente juiz on-line CodeBench.

1.3 Hipótese

Embora a utilização da biometria comportamental na autenticação de usuários tenha tido resultados satisfatórios [Mondal & Bours, 2017], ainda não existem trabalhos que mostram a eficácia de tais métodos em sistemas juiz on-line. Diante dessa perspectiva, formulou-se a seguinte hipótese:

- É possível autenticar os alunos em ambientes Juiz on-line através do uso de aprendizagem profunda aplicada à dinâmica de digitação dos alunos durante as atividades iniciais de programação.

1.4 Objetivo geral e específicos

Desenvolver e avaliar um método de autenticação baseado em biometria comportamental por padrões de digitação utilizando aprendizagem profunda, com intuito de autenticar de forma não intrusiva alunos em ambientes juiz on-line.

Objetivos específicos:

- investigar como a distribuição de amostras das mesmas atividades nos conjunto de treino e teste impactam na acurácia;
- investigar se a incorporação de informação dos caracteres pressionados agregam informação à dinâmica de digitação e melhora o reconhecimento;
- provar a eficácia das redes convolucionais siamesa, tendo como entrada caracteres brutos da dinâmica de digitação para autenticação de usuários; e
- elaborar uma metodologia de avaliação que considere a evolução temporal (ou por atividades) da aprendizagem dos alunos para verificar se os padrões biométricos mudam.

1.5 Considerações

O processo de autenticação será realizado sem exigir nenhuma ação explícita do usuário. Para isso, foi projetada uma arquitetura de CNN Siamesa que buscará aprender, a partir dos dados brutos do ritmo da digitação dos usuários, a representação necessária para o reconhecimento dos alunos, dispensando, desta forma, a extração manual de *features*, no qual é um processo que exige muito tempo e a participação de especialistas. Além disso, a arquitetura de Rede Neural proposta é utilizada como um extrator de *features* genérico que poderá ser utilizado para classificar novos alunos, novas turmas, sem que seja necessário atualizá-la (retreino).

Com a utilização do método proposto, pretende-se diminuir as ocorrências reais e prováveis de fraude por parte de estudantes em exercícios e avaliações que ocorrem nos sistemas juiz on-line. Considerando que a biometria comportamental ajuda a evitar trapaças, monitorando e analisando as entradas de dados de um usuário com base em seus padrões habituais de comportamento.

Cabe destacar que os resultados descritos na seção 5 fazem parte de um artigo aceito no Simpósio Brasileiro de Informática na Educação 2020.

1.6 Organização

No Capítulo 2 é apresentada a fundamentação teórica necessária para o entendimento do presente estudo, onde são explanados os principais conceitos sobre juízes on-line, biometria comportamental e autenticação. Além disso, conceitos e técnicas de aprendizagem profunda. No Capítulo 3 são apresentados os trabalhos que possuem objetivos e métodos de pesquisa semelhantes e que são o presente estado da arte. Fez-se ainda uma descrição das publicações, categorizando-as pelo autor e ano, o classificador utilizado e tipo de autenticação, bem como as bases de dados utilizadas, o desempenho e o contexto onde foram empregados. Finalmente, faz-se uma comparação sintetizada dessas pesquisas. No Capítulo 4 é explanada a arquitetura da solução proposta para autenticação de alunos em ambientes juiz on-line, utilizando biometria comportamental e aprendizagem profunda. No Capítulo 5 é apresentado e discutido sobre os resultados obtidos nos experimentos conduzidos. No Capítulo 6 são explanadas as considerações finais, as limitações do método e os possíveis trabalhos futuros.

2

Fundamentos Teóricos

Neste capítulo são apresentados os conceitos prévios necessários para o entendimento desta proposta. A fundamentação teórica é dividida em seis seções. Na seção 2.1 são apresentados os principais conceitos sobre Juízes on-line. A seção 2.2 descreve as definições formais de biometria, autenticação biométrica e biometria comportamental. Na seção 2.4 são descritos conceitos e métodos de aprendizagem de máquina e aprendizagem profunda. A seção 2.4.1.2 descreve o método de aprendizagem profunda que será adotado neste trabalho. Na seção 2.5 são detalhadas as métricas de avaliação aplicadas. E, por fim, na seção 2.6 encontram-se as considerações finais deste capítulo.

2.1 Juiz On-line

Juízes on-line são sistemas cujo propósito é compilar, executar e avaliar códigos-fonte. Tais sistemas permitem que os alunos desenvolvam seus próprios códigos como resposta a determinados exercícios de programação solicitados pelos professores. Além disso, esses ambientes são capazes de avaliar se os códigos submetidos pelos alunos estão corretos ou incorretos. Para que isso ocorra, é comparada a saída retornada com uma saída esperada, mediante a execução do código.

Segundo Santos & Ribeiro [2012], existem diversos juízes on-line que podem ser encontrados na Internet. Entre eles pode-se citar o *BOCA Online Contest Administrator*[1]; um sistema que visa apoiar as competições de programação organizadas pela SBC (Sociedade Brasileira de Computação). No Brasil, dentre os juizes on-line mais conhecidos está o sistema *URI Online Judge*[2], que foi criado para servir de apoio e complemento de estudos para estudantes de Engenharias e Ciência da Computação. O

[1]https://code.launchpad.net/boca
[2]https://www.urionlinejudge.com.br/judge/en/login

sistema URI oferece um ambiente de interação entre alunos, que tem como uma de suas facilidades a troca de experiências, além de possibilitar que alunos iniciantes obtenham auxílio de estudantes mais avançados [Galvão et al., 2016].

No contexto de apoio pedagógico, segundo Galvão et al. [2016] existem diversas plataformas juiz on-line. Por exemplo, o *The Huxley*[3] possui um número significativo de problemas de programação cadastrados, e permite que os alunos desenvolvam e submetam códigos-fonte da solução desenvolvidos em diferentes linguagens de programação. Além disso, corrige automaticamente os códigos e emite (caso necessário) dicas personalizadas de como solucionar os erros presentes em tais códigos.

Outro diferencial do *The Huxley* é que ele favorece a interação entre aluno e professor, permitindo ao professor acompanhar a evolução constante de seus alunos. Nesse cenário, outra ferramenta que vale destaque é o *MOJO*, um módulo de integração entre juízes online e o *Moodle*, desenvolvido para diminuir o trabalho de elaborar questões e de corrigir códigos dos alunos.

Nesta pesquisa, o juiz on-line utilizado será o *CodeBench*[4], desenvolvido na Universidade Federal do Amazonas - UFAM com o objetivo de oferecer aos professores melhor controle durante a aplicação de avaliações em classe [Galvão et al., 2016]. Além disso, esse sistema serve de ambiente para resolução de exercícios de programação solicitados por diversas disciplinas ofertadas na universidade.

No CodeBench, os professores podem disponibilizar exercícios de programação para seus alunos, que por sua vez podem codificar soluções e submetê-las através da interface do sistema [Galvão et al., 2016], mostrada na Figura 2.1. Além de que, essa plataforma, possui um Ambiente de Desenvolvimento Integrado (IDE) na qual é possível coletar dados durante a resolução dos exercícios.

Entretanto, o registro de autenticação de usuários no CodeBench ainda é feito somente de maneira estática. Ou seja, caso alunos trapaceiros peçam para terceiros fazerem suas atividades, não serão descobertos. Além disso, qualquer indivíduo que tenha a senha de um aluno pode logar no sistema, consequentemente, poderá ocorrer casos de fraudes. Nesta pesquisa busca-se amenizar esse tipo de problema, utilizando autenticação baseada em biometria comportamental.

2.2 Biometria

Biometria é o estudo de características fisiológicas e comportamentais de um indivíduo (ser vivo) com intuito de identificá-lo [Handa et al., 2019]. Nesse sentido, a auten-

[3]https://www.thehuxley.com/
[4]http://codebench.icomp.ufam.edu.br/

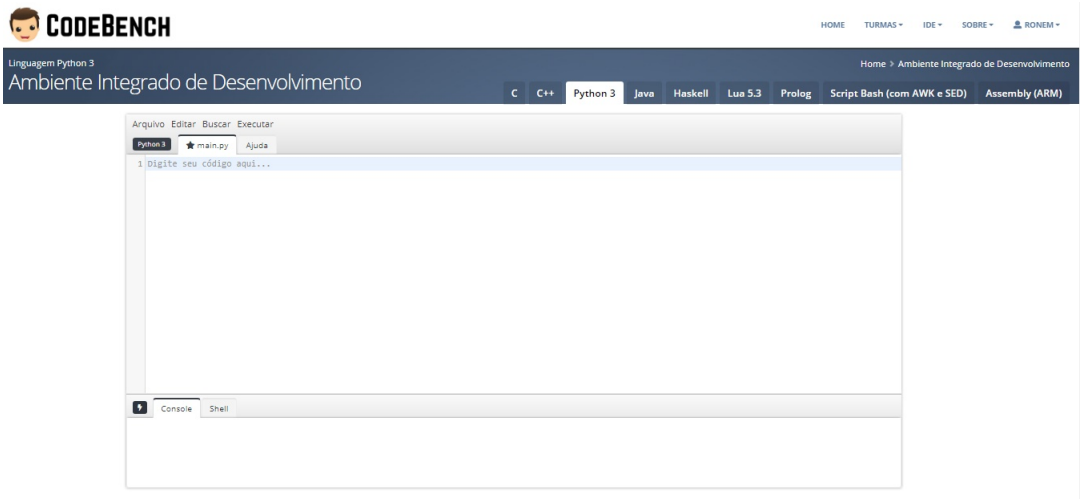

Figura 2.1: Interface do juiz on-line Codebench.

ticação biométrica avalia uma ou mais características biológicas de uma pessoa, por exemplo, impressão digital, íris, dinâmica na digitação de textos e reconhecimento de fala, buscando assim, reconhecê-la [Adetunji et al., 2018]. Essas são algumas das características biométricas exclusivas de cada indivíduo. *features* como essas são utilizadas principalmente para autenticação, identificação e controle de acesso seguro em sistemas computacionais.

Segundo Feher et al. [2012], há muitas vantagens em utilizar características biométricas para autenticação porque:

- é mais seguro, pois a pessoa a ser autenticada deve apresentar-se no ponto de autenticação;
- tem a capacidade de alta precisão de identificação individual;
- tem facilidade de uso, consome menos tempo;
- exige treinamento simples e menos dispendioso a longo prazo;
- características biométricas são únicas de cada pessoa, logo, elas não podem ser roubadas ou conjecturadas; e
- essa tecnologia é capaz de identificar indivíduos de forma consistente, com rapidez e confiabilidade.

Enquanto métodos comuns de autenticação como senhas são fáceis de esquecer, fazendo com que as pessoas as anotem e consequentemente, podem ser roubadas [Mondal & Bours, 2017], a autenticação biometria é única. Além disso, é extremamente

difícil de ser duplicada. Esse método ganhou ampla aceitação em muitas empresas na forma de *scanners* de impressão digital, sendo utilizada para autenticação de funcionários. Conforme ilustrada na Figura 2.2, a biometria é classificada ou subdividida em duas classes diferentes: baseadas nas características fisiológicas e nas características comportamentais.

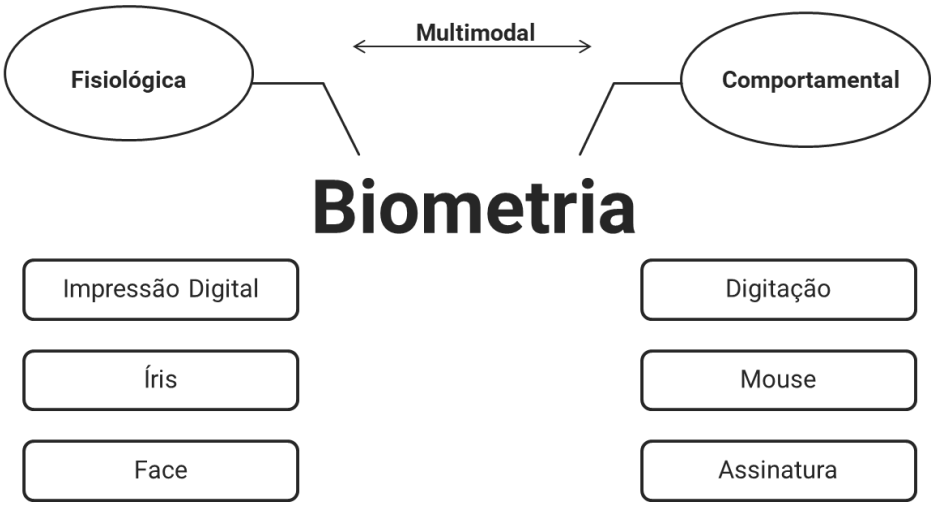

Figura 2.2: Classificação Biométrica.

Características fisiológicas incluem face, impressão digital, íris e geometria da mão. A biometria fisiológica é mais precisa em comparação com biometria comportamental. Entretanto, é exigido a utilização de dispositivos dedicados especiais para a aquisição das informações. Logo, leva-o a um alto custo de implementação [AV & Rathi, 2019].

Por outro lado, características comportamentais são, por exemplo, a fala do usuário, assinatura manuscrita, dinâmica de digitação e etc. Para utilizar a biometria comportamental, não é preciso hardware extra e é menos invasivo para o usuário, considerando que ele funciona em segundo plano [Adetunji et al., 2018].

2.2.1 Autenticação Biométrica

Comumente, sistemas computacionais autenticam seus usuários, nesse caso, autorizar ou certificar como legítimo ou autêntico [Andrade et al., 2019]. Segundo Mondal & Bours [2015], métodos de autenticação podem ser classificados em duas abordagens diferentes: os métodos de autenticação estáticos e contínuos.

A autenticação estática *(Static Authentication - SA)* refere-se ao ato de verificar a autenticidade do usuário geralmente no acesso inicial ao sistema, por exemplo, mediante a digitação de uma senha, enquanto que a autenticação contínua *(Continuous Authentication - CA)*, a verificação é feita continuamente com base na atividade que ele executa no sistema [Mondal & Bours, 2017].

Ambas as abordagens podem ser implementadas de formas: através da identificação ou da autenticação, normalmente utilizando biometria. Na identificação, o sistema distingue os padrões biométricos disponibilizados pelo usuário, em seguida, faz-se uma comparação com todos os que estão armazenados, nesse caso, por exemplo, a pergunta a ser respondida é: esta pessoa está na base? Enquanto que na autenticação ou verificação, o usuário apresenta a identidade (perfil biométrico), em seguida, o sistema verifica se este é quem afirma ser, nesse caso, a pergunta a ser respondida é: esta pessoa é quem afirma ser? [Silva Filho et al., 2005].

Essa biometria de sistema de autenticação funciona com as características do usuário. Nesse caso, primeiramente são capturadas e selecionadas *features* do usuário, em seguida, essas *features* são enviadas como entrada para um algoritmo de inteligência artificial que aprenderá um padrão biométrico nesse dados e posteriormente será capaz de determinar a identificação do indivíduo, nesse caso, se é um usuário genuíno ou impostor.

Neste contexto, a mensuração das *features* constituem o modelo ou perfil biométrico do usuário. No registro ou cadastramento de um determinado usuário no sistema, são capturados seus primeiros dados biométricos. Em seguida, o perfil biométrico é gerado a partir desta base de conhecimento, isto para cada indivíduo, posteriormente esse perfil biométrico será utilizado para possíveis comparações. Assim sendo, o perfil biométrico do usuário é o fator de identificação do proprietário e sua determinada conta [Estrela, 2020].

Neste cenário, a dinâmica de digitação tornou-se a principal fonte biométrica comportamental para fornecer autenticação de usuários [Feher et al., 2012]. A verificação do comportamento do usuário baseado no seu ritmo de digitação, provou-se ser útil para detectar se o usuário é um usuário legítimo ou não, pois o teclado faz parte integrante dos dispositivos de entrada do sistema computacional.

2.3 Biometria Comportamental por Dinâmica de Digitação

O termo "dinâmica de digitação" também é conhecido na literatura como ritmo de digitação ou padrão de digitação. Esse termo está relacionado a ação de pressionamento das teclas do teclado [Vinayak & Arora, 2015]. De acordo com Lv et al. [2018], os mesmos fatores que tornam a assinatura manuscrita única também são oportunizados pelo padrão de digitação dos usuários. Por exemplo, quando uma pessoa digita em um teclado, essa deixa uma assinatura digital na forma de velocidade e latências entre as teclas digitadas. O método de autenticação através da dinâmica da digitação é fundamentado na ideia de que cada indivíduo digita no teclado de maneira única, isto é, na ideia de que o ritmo de digitação varia de pessoa para pessoa. Deste modo, se o ritmo de cada indivíduo é único, é possível autenticá-lo através dessa característica. Entretanto, é necessário destacar que o que importa não é o que o indivíduo digita, mas sim como ele digita [Young, 2018].

As vantagens da utilização da dinâmica de digitação incluem seu baixo custo, uma vez que necessita apenas de um teclado para aquisição dos dados, hardware bastante comum. Além disso, por ser um método de autenticação não intrusivo, permite que os usuários sejam autenticados de maneira contínua, adicionando assim, uma camada a mais de segurança, permitindo que apenas usuários genuínos tenham acesso e uso contínuo do ambiente.

Vinayak & Arora [2015] destaca algumas outras vantagens na utilização da dinâmica de digitação em sistemas de autenticação, por exemplo, (a) a dinâmica de digitação pode ser usada por qualquer pessoa que sabe manusear o teclado; (b) todo indivíduo digita de maneira única; portanto, o padrão de digitação de dois usuários não pode ser o mesmo; (c) não pode ser reproduzido; e (d) desde que o usuário interaja com o sistema, o padrão da digitação pode ser constantemente monitorado.

A autenticação baseada em dinâmica de digitação, embora empregadas em vários contextos, é especialmente conveniente para sistemas on-line, como ferramentas de EaD [Longi et al., 2015]. Neste caso, essa técnica pode verificar se o aluno que conclui o trabalho da disciplina é igual à pessoa que se matriculou para fazer o curso. A autenticação baseada em dinâmica de digitação pode ser um pouco menos precisa comparada a autenticação por características fisiológicas, pois frequentemente o usuário pode mudar ligeiramente sua maneira de digitar devido a inúmeras circunstâncias [Acien et al., 2020]. Entretanto, são menos invasivas e podem ser utilizadas constantemente, em vez de somente no início de uma sessão de trabalho [Vinayak & Arora,

2015], oportunizando assim, uma real diminuição de possíveis ocorrências de trapaça por parte do aluno enquanto estiver logado no sistema.

De acordo com Young [2018], a autenticação através da dinâmica de digitação pode ainda ser dividida, quanto ao momento da autenticação, em duas abordagens, sendo elas:

- **Abordagem estática** – onde realiza-se uma análise somente de textos de tamanho fixo como, por exemplo, uma senha. Esta abordagem é utilizada geralmente na autenticação de usuários no acesso inicial ao sistema. Entretanto, possui uma falha inerente: após o usuário realizar o *login*, o sistema não verificará ou detectará se o usuário atual é um usuário genuíno no decorrer de toda sessão.

- **Abordagem dinâmica** – onde a análise ocorre para qualquer texto digitado pelo usuário, não importando a quantidade de caracteres. Neste caso, o usuário é autenticado continuamente pelo método de autenticação, podendo desta forma constatar se houve troca de usuários do decorrer da execução do sistema.

Sendo assim, o recurso biométrico utilizado nesta pesquisa será a dinâmica de digitação. Entretanto, voltada para o contexto de desenvolvimento de códigos de computadores. Por exemplo, quando um aluno codifica, ele possui restrições próprias da atividade e da linguagem de programação utilizada. Desta forma, este recurso biométrico possui suas particularidades do contexto. Assim sendo, uma vez que o usuário manuseia o teclado codificando, são coletados dados temporais referentes aos eventos de pressionamento de cada tecla [Lavareda Filho et al., 2020]. Esses dados coletados são denominados, neste trabalho, de dados brutos. Após a coleta desses dados, normalmente é feita extração manual de *features*, denominada de Engenharia de Atributos [Zhong & Deng, 2015], onde são gerados novos atributos a partir dos dados brutos [Cruz & Goldschmidt, 2018]. Os dois atributos mais comuns são: o tempo de pressão e a latência entre duas teclas, ambos considerados relevantes para o processo de reconhecimento do usuário [Cruz et al., 2017].

A Figura 2.3 apresenta um esquema que ilustra graficamente essas duas *features*. Na Figura, três teclas fictícias são apresentadas em ordem temporal de pressionamento. Cada tecla é representada por K_n, sendo que n indica a ordem em que a tecla foi pressionada. Além disso, cada tecla possui dois registros de tempo, um em que a tecla foi pressionada (K_n^D) e outro em que foi liberada (K_n^U).

O tempo de pressionamento de uma tecla H_K é obtido pela subtração do tempo em que ela foi liberada pelo de quando ela foi pressionada [Cruz & Goldschmidt, 2019].

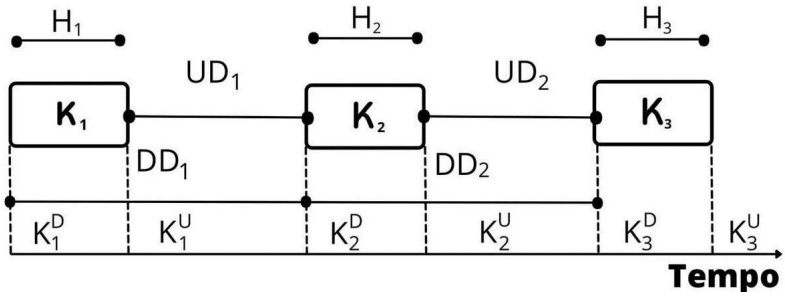

Figura 2.3: Tempos de Pressionamento e Latências.

Na Figura 2.3, por exemplo, o tempo de pressionamento de uma tecla n é representado por H_n, onde $H_n = (K_n^U) - (K_n^D)$.

Segundo Cruz & Goldschmidt [2019], existem diferentes maneiras de encontrar as latências entre duas teclas. A latência UD_n, conforme ilustrada na Figura 2.3, é o intervalo entre o momento que uma tecla é liberada e a tecla seguinte é pressionada. Logo, $UD_n = (K_{n+1}^D) - (K_n^U)$. A latência DD_n é obtida ao subtrair o tempo em que duas teclas consecutivas foram pressionadas. Assim, $DD_n = (K_{n_{+1}}^D) - (K_n^D)$. Cabe destacar que só é possível calcular ambas latências se a tecla K_{n+1} existir.

Os diferentes tipos de latência e o tempo de pressionamento são exemplos de *features* muito utilizadas no treinamento de algoritmos de Aprendizado de Máquina para reconhecer usuários. entretanto, nesta pesquisa, além da utilização das *features* citadas acima, iremos adicionar como *feature* extra informação textual (caracteres digitados na resolução de atividades). A hipótese é que existe uma correlação entre os caracteres digitados com as *features* de pressionamento e latências da dinâmica de digitação, já que ambos possuem particularidades do contexto e restrições de uma determinada linguagem de programação.

2.4 Aprendizagens de máquina

Diferentes algoritmos podem ser utilizados para a autenticação de usuários a partir de suas características biométricas comportamentais. Entre eles estão os algoritmos de aprendizagem de máquina.

A aprendizagem de máquina preocupa-se com reconhecimento de padrões e aprendizado computacional no contexto de inteligência artificial [Sundararajan & Woodard, 2018]. Técnicas de aprendizagem de máquina têm sido utilizadas em diversas áreas. Por exemplo, aplicações no comércio, administração, segurança, dentre outras. A principal vantagem de utilizar aprendizagem de máquina é a sua capacidade de criar modelos discriminativos que aprendem a partir de conjuntos de dados [Cruz et al., 2017].

Modelos de aprendizagem de máquina podem ser classificados em dois tipos: classificação ou regressão, onde a diferença é o tipo de predição que se deseja realizar. No caso de problemas de classificação, os modelos são denominados classificadores.

A principal função dos classificadores é analisar um conjunto de dados buscando reconhecer padrões que permitam categorizá-los, ou seja, separá-los em categorias. Neste estudo, denominaremos classificadores deste tipo como métodos de aprendizagem de máquina clássica. Dentre os mais utilizados, podemos citar: Naive Bayes, kNN (do inglês: K nearest neighboors), Floresta aleatória (do inglês: RandomForest) e o SVM (do inglês: support vector machine) [Adetunji et al., 2018].

O SVM tornou-se um dos classificadores mais utilizados devido sua boa precisão utilizando menos recurso computacional [Centeno et al., 2018]. O objeto do SVM é encontrar um hiperplano em um espaço N-dimensional que classifica distintamente os pontos de dados. Para separar as duas classes, existem muitos hiperplanos possíveis que podem ser escolhidos. Nesse caso, o objetivo é encontrar um plano que tenha a margem máxima, ou seja, a distância máxima entre os pontos de dados de ambas as classes. Hiperplanos são limites de decisão que ajudam a classificar os pontos de dados. Os pontos de dados que caem em ambos os lados do hiperplano podem ser atribuídos a diferentes classes. Além disso, a dimensão do hiperplano depende do número de *features*. Se o número de *features* de entrada for dois, o hiperplano é apenas uma linha. Se o número de *features* de entrada for três, o hiperplano se tornará um plano bidimensional.

Entretanto, os modelos SVMs tradicionais não podem ser usados diretamente para problemas de classificação multiclasse, para este caso, foram criados métodos para estender os SVMs para este tipo de problema, onde o problema multiclasse é dividido em vários problemas de classificação binária e, em seguida, é treinado um modelo de classificação binária para cada um e as previsões são feitas usando o modelo mais confiável. A forma de decomposição mais utilizadas na realização desse tipo tarefa é chamado de um-contra-todos (One-vs-All)). Um-contra-todos é um método heurístico onde utiliza algoritmos de classificação binária para classificação de várias classes [Yao et al., 2001].

Além desses, um dos principais modelos utilizados para classificação são as redes neurais artificiais (RNA). As redes neurais artificiais são consideradas um dos mais importantes modelos da área de aprendizado de máquina. Como o nome sugere, as RNA são inspiradas no funcionamento do cérebro humano, tentando replicar a maneira que aprendemos. Esse modelo é formado por um ou mais tipos de neurônios e pode ser dividido normalmente em camada de entrada, camada oculta e camada de saída, onde a camada de entrada não possui neurônios e a saída tem como base funções de

ativação predefinidas [Lin et al., 2018]. Além disso, as RNA possuem múltiplas vantagens sobre os classificadores anteriores, tanto em desempenho quanto em escalabilidade [Sundararajan & Woodard, 2018].

2.4.1 Aprendizagem profunda

Métodos clássicos de aprendizagem de máquina são limitados, exigindo muito conhecimento específico sobre o domínio em que serão aplicados para transformar os dados de maneira adequada aos classificadores.

Métodos de aprendizagem profunda ou *Deep Learning*, por outro lado, são capazes de aprender, a partir dos dados de entrada (sem extração manual de atributos), a melhor maneira de representar os dados para a tarefa de classificação [Cruz & Goldschmidt, 2019]. Além disso, esses modelos possuem a característica de possuir múltiplas camadas, aprendendo diferentes níveis de representações [Bhanu & Kumar, 2017].

Deep Learning, em tradução livre, significa aprendizado profundo. Essa é uma técnica que busca construir um modelo capaz de extrair uma representação alto nível dos dados. O termo *Deep* é decorrente do conceito que as camadas são conectadas uma após a outra, o qual fornece o conceito de profundidade. Dentre os diversos algoritmos que são classificados como *Deep Learning*, destaca-se a CNN (*Convolutional Neural Network*, em português Redes Neurais de Convolução), a qual será descrita na próxima seção.

2.4.1.1 Redes Neurais Convolucionais

As Redes Neurais Convolucionais (CNNs) são tipos especiais de redes neurais artificiais que obtiveram sucesso considerável na classificação de imagens. Em sua essência, as redes neurais convolucionais são um conjunto de operações matemáticas não lineares sobre uma imagem. [Xiaofeng et al., 2019].

Uma das principais vantagens das CNNs em relação às outras redes neurais é que essa rede pode receber como entrada dados brutos coletados sem qualquer pré-processamento [Dwivedi et al., 2018]. O recurso de convolução permite extrair a partir desses dados *features* complexas essenciais para tarefa de classificação [Çeker & Upadhyaya, 2017].

Entre as redes CNNs destacam-se os componentes descritos a seguir [Xiaofeng et al., 2019]:

Camada convolucional - Essa camada é responsável pela aplicação sistemática de um ou mais filtros a uma entrada, com o objetivo de extrair um conjunto de características chamadas de mapas de recursos *(Feature Maps)*. Para isso, (em imagens) o

filtro é aplicado repetidamente a cada parte da imagem de entrada, resultando em um mapa de ativação de saída bidimensional (mapa de recursos). O filtro contém os pesos que precisam ser aprendidos durante o treinamento da camada. Os pesos representam a estrutura ou característica que o filtro detecta e a intensidade da ativação indica o grau em que a característica foi detectada. A camada requer que o número de filtros e a forma dos filtros sejam especificados. Segundo Medeiros et al. [2019], o processo de convolução pode ser representado conforme as equações abaixo:

$$conv_{x,y} = \sum_i w_i v_i, \tag{2.1}$$

onde $conv_{x,y}$ é o resultado da convolução sobre a entrada da CNN; w_i são os pesos de cada filtro; v_i são os valores correspondentes no mapa de entrada para cada região em que o filtro é deslocado; e i varia de 1 até o número de canais de entrada da CNN.

O valor obtido na Equação 2.1 é somado ao valor de *bias*, conforme mostrado na Equação 2.2. O valor de *bias* b insere não linearidade no cálculo de cada convolução, o interesse com a utilização do valor de *bias* é inserir um limiar mínimo para ressaltar as características encontradas na convolução, o que permite controlar a ativação dos neurônios.

$$Z_{x,y} = \sum_i w_i v_i + b, \tag{2.2}$$

onde $Z_{x,y}$ é o resultado da convolução e dá origem a cada elemento (x, y); b é o valor do *bias*; e i varia de 1 até o número de canais do mapa de entrada.

O valor calculado através da Equação 2.2 é submetido a uma função de ativação h para gerar um elemento $a_{x,y}$ do mapa de características, conforme $a_{x,y} = h(Z_{x,y})$. Geralmente, esta função de ativação é não linear.

Função de ativação - A camada convolucional normalmente é seguida pela função de ativação não linear aplicada de maneira pontual às suas saídas para aprender limites de decisão não lineares. Três funções de ativação comumente usadas são *sigmoidal*, *tangente hiperbólica* e *ReLU*.

A *rectified linear activation function ReLU* é função de ativação mais utilizada, devido a sua rapidez de execução computacional, pois para valores de entrada inferiores a zero a saída será zero e para valores superiores a zero o valor de saída será o respectivo valor de entrada. A função *ReLU* é ilustrada na Figura 2.4.

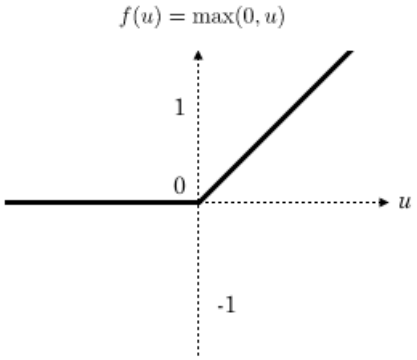

Figura 2.4: Função *ReLU* - Ilustração Gráfica.

Para resolver defeitos de inconsistência na distribuição de dados, costuma-se utilizar "Normalização em lote"***(Batch Normalization)***. Ao normalizar a entrada de cada camada, esse método pode garantir que a distribuição de dados de entrada de cada camada seja estável e consistente, de modo a convergir rapidamente [Lv et al., 2018].

Camada de pool - Um conceito importante para CNNs é a compressão dos recursos de entrada. Duas opções convencionais para fazer isso são obter uma média ou máximo de pequenos blocos retangulares dos dados. O processo de compressão irá manter os recursos, reduzindo a complexidade computacional e destacando as principais características.

Camada totalmente conectada - Após várias camadas convolucionais e máximas, a saída dessas camadas é achatada em um vetor unidimensional e usada para a classificação. Nesta fase, recursos adicionais podem ser empilhados juntos com esse vetor. Para aprender dependências não lineares, a CNN possui uma ou mais camadas totalmente conectadas que executam a classificação.

Camada *soft-max* - Por fim, a saída da última camada é passada para uma camada máxima suave que calcula a distribuição de probabilidade nas classes previstas.

Os modelos CNN convencionais foram desenvolvidos para classificação em dados 2D, como imagens e vídeos. Por isso, comumente são chamados de CNNs 2D, nos quais o modelo aceita uma entrada bidimensional representando os *pixels* e os canais de cores de uma imagem, em um processo chamado de aprendizado de recursos *(feature learning)* [Liu & Guan, 2017]. Esse mesmo processo pode ser aplicado a sequências unidimensionais de dados (CNN 1D). O modelo extrai *features* de dados de sequências e mapeia as *features* internas da sequência. Uma CNN 1D é muito eficaz para derivar *features* de um segmento de comprimento fixo do conjunto de dados geral, onde não é tão importante onde a *features* está localizada no segmento [Zabihi et al., 2019].

Além disso, os modelos CNN 1D são tipicamente mais rápidos e mais simples de treinar em comparação com redes neurais recorrentes (RNN), como LSTM [Bradbury et al., 2016]. Para certas aplicações, as CNNs 1D são vantajosas e, portanto, preferíveis comparadas a CNN 2D [Zabihi et al., 2019].

2.4.1.2 Rede Neural Siamesa

Apesar de todos os avanços ofertados pelas redes neurais convolucionais, uma das principais limitações acerca desse método é a necessidade de consumir muitos dados rotulados para treinamento. Em muitos casos, a coleta de tantos dados inviabiliza o uso do método. As Redes Neurais Siamesas (RNS) visam solucionar esse problema.

As redes siamesas foram introduzidas pela primeira vez para resolver a verificação de assinaturas [Santos et al., 2018]. Nesse caso, observou-se que a estratégia de minimização de medidas de distância entre vetores de *features* extraídos em seu treinamento, permitiu a utilização de menos parâmetros a serem otimizados e mais rápida convergência da rede neural. Desde então, as redes siamesas têm sido aplicadas em diversas tarefas, em especial para identificação de pessoas em fotos e vídeos [Jmila et al., 2019].

As redes neurais siamesas são muito boas em encontrar semelhanças entre dados [Santos et al., 2018], diferentemente das redes neurais convolucionais que aprendem a identificá-los. Por exemplo, as redes CNNs buscam identificar o que é um determinado objeto, enquanto que as RNS são utilizadas para aprender semelhanças entre objetos (o que faz dois objetos serem iguais). Além disso, as RNS não precisam de milhares de exemplos para treinamento; esse é um dos motivos para que esse tipo de redes esteja sendo muito utilizadas em sistemas de reconhecimento, pois possibilita também trabalhar com um único aprendizado (*One-Shot Learning*).

O aprendizado único é uma tarefa de classificação em que um exemplo (ou um número muito pequeno de exemplos) de cada classe é usado para preparar um modelo, que por sua vez deve fazer previsões futuras sobre muitos exemplos desconhecidos. Por exemplo, considerando o problema de reconhecimento facial, um modelo ou sistema pode ter apenas um exemplo registrado do rosto de uma pessoa e deve verificar corretamente novas fotos dessa pessoa, talvez todos os dias. Como tal, o reconhecimento de rosto é um exemplo comum de aprendizado único.

As redes siamesas popularizaram-se devido à sua eficácia em tarefas de aprendizado único. O termo siamesa se deve ao fato dessa arquitetura ser composta por sub-redes idênticas que compartilham os mesmos parâmetros [Santos et al., 2018], conforme ilustrado na Figura 2.5. Segundo Vorugunti et al. [2019], cada sub-rede recebe uma entrada, e o modelo produzirá uma pontuação de similaridade indicando as chan-

ces das duas entradas pertencerem à mesma classe.

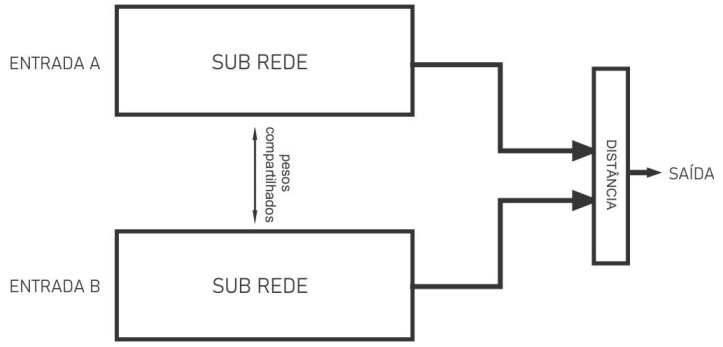

Figura 2.5: Arquitetura de uma rede neural siamesa clássica

Por exemplo, dada duas entradas, a rede siamesa cria um espaço vetorial n-dimensional aplicando a função $G_w(X) \rightarrow \mathbb{R}^n$. Cada entrada X_1 e X_2 é mapeada por G_x e cada saída é submetida a uma função de similaridade $Sim(G_w(X1), G_w(X2)) \rightarrow \mathbb{R}^1$. A função de similaridade é normalmente baseada na medida de distância Euclidiana (D), e é obtida da seguinte forma:

$$D = Sim(G_w(X1), G_w(X2)) = ||G_w(X1) - G_w(X2)||_2 \tag{2.3}$$

onde $G_w\,(X1)$ e $G_w\,(X2)$ são dois pontos em um espaço vetorial multidimensional criados pelos parâmetros compartilhados w quando mapeiam as entradas $X1$ e $X2$. Uma vez obtida a medida de similaridade, é possível decidir se os dados de entrada pertencem ou não a uma mesma classe (ou um novo aluno, no contexto deste trabalho) dado um limiar de distância definido.

Para cada par de amostras, a distância D entre os vetores de saída das duas redes é calculada pela função de perda *(loss function)*. Existem duas funções de similaridade por meio das quais a rede siamesa pode ser treinada, a *Contrastive Loss* e *Triplet loss*. A *Triplet loss* é uma alternativa para *Contrastive Loss* que utiliza três entradas: uma âncora (A), um exemplo positivo (P), e um exemplo negativo (N), conforme ilustrada na Figura 2.6. De acordo com o Schroff et al. [2015], a *Triplet loss* é uma função de perda que treina uma rede neural para extrair *embeddings* (vetor de características) similares entre A e P, enquanto maximiza a distância dos *embeddings* P e N por serem de classes diferentes. A ideia é que dados da mesma classe têm seus *embeddings* próximos no espaço vetorial. Enquanto que, dados de classes diferentes têm seus *embeddings* distantes. A função de perda é descrita como uma diferença de distâncias Euclidianas:

$$loss(A, P, N) = \max(||A_e - P_e||_2 - ||A_e - N_e||_2 + \alpha, 0) \tag{2.4}$$

onde α é um hiperparâmetro chamado margem de separação.

As triplas de exemplos para treinamento podem ser divididas em: fáceis, semi-difíceis, e muito difíceis. Cada uma dessas definições depende de onde está a amostra negativa em relação à âncora e à amostra positiva. A escolha do tipo de *Triplet* de treinamento tem grande impacto nos resultados. As triplas para treinamento podem ser geradas de forma *offline* ou durante o tempo de treinamento, *online*. Neste trabalho, utilizaremos a abordagem semi-difícil que emprega o aprendizado online para selecionar uma âncora.

No aprendizado online, a escolha das triplas é feita durante o processo de treinamento. Conforme Schroff et al. [2015], esta versão apresenta melhores resultados. Nesta formulação, a amostra negativa tentará maximizar a sua distância euclidiana da âncora, mas ainda será capaz de oferecer uma perda positiva [Schroff et al., 2015], conforme:

$$||\mathrm{d}(A_i, P_i)||_2 \leq ||\mathrm{d}(A_i, N_i)||_2 \leq ||\mathrm{d}(A_i, P_i)||_2 + \alpha \tag{2.5}$$

onde α é a margem, A_i é a âncora, P_i a localização da amostra positiva, N_i da negativa e d é a distância entre as as amostras.

Figura 2.6: Triplet loss clássica

Para verificar a similaridade entre padrões da dinâmica de digitação é preciso que a arquitetura utilizada para extrair *features* do conjunto de treinamento seja robusta. Sendo assim, utilizaremos redes neurais convolucionais siamesa. Além disso, para guiar o treinamento desta rede será utilizada a função de perda *Triplet loss*.

2.5 Medidas de Desempenho

Uma vez treinados, os algoritmos de reconhecimento precisam ser avaliados através de métricas específicas [Cruz et al., 2017]. As métricas neste contexto são:

Taxa de falsa rejeição (FRR - *False Rejection Rate*): que indica quantos usuários genuínos foram considerados impostores, ou ainda, não foram reconhecidos pelo sistema biométrico. Essa taxa também é conhecida como: *erro tipo I* ou *False Alarm*

Rate. A FRR é definida pela seguinte equação:

$$FRR = (TFR/TCT)\% \tag{2.6}$$

onde TFR representa o total de falsa rejeição e TCT representa o total de contas tentadas.

Taxa de falsa aceitação (FAR - *False acceptance Rate*): que indica a taxa de usuários impostores que foram considerados como usuários legítimos pelo sistema biométrico. Essa taxa também é conhecida como: *erro tipo II* ou *Impostor Pass Rate*. A FAR é definida pela seguinte equação:

$$FAR = (TFA/TCT)\% \tag{2.7}$$

onde TFA representa o total de falsa aceitação e TCT representa o total de contas tentadas.

Como essas duas taxas são inversamente proporcionais, adota-se a Taxa de erro igual (EER), que representa o ponto de equilíbrio entre as taxas (Figura 2.7). Se um sistema biométrico tem EER de 0,1%, isso significa que sua eficácia será 99,9%. A EER é derivada da curva de características operacionais *(receiver operating characteristic - ROC)* que normalmente é usada para comparar o desempenho de diferentes sistemas biométricos. Logo, o valor de EER pode ser calculado a partir dessa curva, que traça FRR em função de FAR [Cruz & Goldschmidt, 2019]. A AUC é a área sob a curva ROC, quanto maior a AUC, melhor o ROC. O ROC, AUC e EER são ilustrados na Figura 2.7.

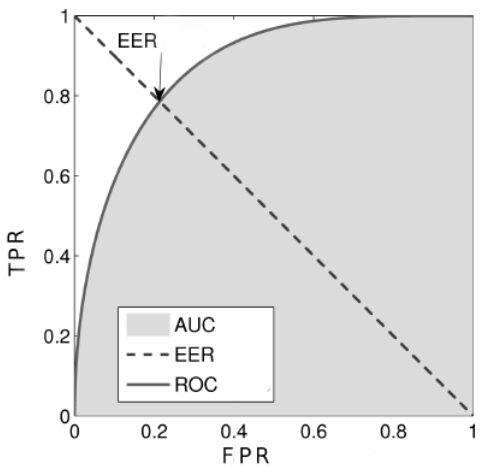

Figura 2.7: Curva ROC

Além dessas, uma medida de desempenho bastante popular é a acurácia. Ela informa o percentual de acertos do modelo em relação ao total de amostras do conjunto de testes [Cruz et al., 2017]. Desse modo, quanto maior a acurácia do modelo, melhor seu desempenho.

2.6 Considerações finais

Neste capítulo, apresentamos os fundamentos teóricos utilizados para o desenvolvimento deste trabalho, incluindo os conceitos de ambientes juiz on-line, autenticação biométrica, biometria comportamental e relacionados; além da descrição de conceitos e métodos de aprendizagem de máquina, aprendizagem profunda, e um breve resumo sobre medidas de desempenho. No próximo capítulo, serão apresentados os trabalhos relacionados, que também fazem uso de conceitos apresentados neste capítulo.

3

Trabalhos Relacionados

Neste capítulo, será apresentado uma revisão da literatura sobre a utilização da biometria comportamental como recurso para a autenticação de usuários. É realizada uma classificação dos trabalhos segundo o ambiente empregado, isto é: dinâmica da digitação em contextos gerais (seção 3.1), dinâmica da digitação em cursos on-line (seção 3.1.1), dinâmica de digitação em sistemas juiz on-line (seção 3.1.2). Além disso, na seção 3.2 é apresentado alguns trabalhos que utilizam dinâmica de digitação como dados de entrada em redes siamesas. Na seção 3.3 é feita uma discussão acerca dos trabalhos apresentados. Por fim, na seção 3.4 são feitas as considerações finais do capítulo.

3.1 Dinâmica da digitação em contextos gerais

Quraishi & Bedi [2019] desenvolveram um método de autenticação utilizando biometria comportamental, com o objetivo de autenticar de maneira contínua usuários em sistemas *desktops*. Para isso, foram extraídos um conjunto de *features* da dinâmica da digitação de cada participante, dentre as quais, o tempo de duração do pressionamento e latência de cada tecla. A aquisição de dados foi feita em ambiente real, sem nenhuma restrição ao lugar ou tipo de teclado. Para classificação do usuário, utilizaram o classificador SVM. Apesar do desempenho do sistema ser regular, nos experimentos é mostrado que o método de autenticação foi capaz de detectar de 42% a 44% das ações dos usuários impostores como sendo genuínas (*False Acceptance Rate* - FAR). Enquanto que para um usuário genuíno, suas ações podem ser consideradas impostoras (*False Rejection Rate* - FRR) de 46% a 49%. Segundo os autores, o trabalho possui grandes possibilidades de melhoria, pois os experimentos foram feitos em ambiente

aberto, não controlado e sem restrição em relação à configuração do ambiente ou do sistema.

Xiaofeng et al. [2019] propuseram um modelo de redes neurais híbrida (CNN + RNN) para aprender características discriminativas da dinâmica de digitação dos usuários em textos sem limites de digitação, com o intuito de autenticar continuamente esses usuários. No trabalho, inicialmente os dados digitados são vetorizados de acordo com as combinações de recursos oriundos do tempo da dinâmica da digitação de cada usuário. Em seguida, são divididos em sequências de recursos de comprimento fixo e repassados para serem processados pela CNN, onde será extraído as características únicas da digitação de cada indivíduo. Essas características discriminativas de cada usuário serão entradas de sequência na rede RNN. Nos experimentos, o modelo foi testado usando um conjunto de dados aberto da universidade de Clarkson [Vural et al., 2014][1] e a melhor Taxa de Erro Igual (EER) foi de 3,04%. O modelo pode alcançar o melhor resultado de reconhecimento de identidade utilizando segmentos fixos de sequência de 30 teclas, atingindo boa praticidade.

3.1.1 Dinâmica da digitação em cursos on-line

Young et al. [2019] examinou o potencial uso da dinâmica da digitação para criar assinaturas digitais de digitação (impressões digitais), com o objetivo de autenticar alunos em cursos on-line. Além de replicar alguns trabalhos anteriores que compartilhavam da mesma finalidade, esse estudo explorou as melhores práticas para implementar as assinaturas de digitação para a autenticação, em situações em que a verificação de senha não possa ser aplicada de forma simples. Os resultados desse estudo indicam que impressões digitais de digitação podem indicar de forma confiável quando uma amostra de digitação não é de um aluno genuíno.

Em Cruz et al. [2017] propuseram um mecanismo de autenticação de usuário nos Ambientes Virtuais de Aprendizagem (AVAs), com o objetivo de realizar autenticação periódica não intrusiva de usuários. Para isso, utilizaram diversos classificadores de aprendizado de máquina clássica para construir modelos de reconhecimento baseados na dinâmica da digitação dos usuários. Um protótipo do mecanismo proposto foi implementado, integrado ao *Moodle*, e aplicado em uma turma de pós-graduação com dezessete usuários. Os modelos de reconhecimento gerados pelo protótipo no estudo de caso apresentaram desempenho acima de 92% de acurácia, fornecendo um indicativo favorável acerca da viabilidade de utilização do mecanismo proposto.

[1] https://www.clarkson.edu/citer/research/collections/index.html

3.1.2 Dinâmica da digitação em sistemas Juiz On-line

Não foram encontrados trabalhos que abordem a utilização da dinâmica de digitação para a autenticação de alunos em ambientes juiz on-line. Entretanto, o estudo de Longi et al. [2015] mostra que alunos podem ser autenticados ou identificados a partir de seus dados da dinâmica da digitação feitas em sessões de disciplinas de programação. Foram utilizados dados de dois cursos de programação realizados na Universidade de Helsinque no ano de 2014 com duração de 6 semanas. Nesse trabalho, os autores apresentaram um estudo sobre como a quantidade de dados disponíveis afeta a precisão da identificação através de dois experimentos. No primeiro, quando utilizado apenas uma única semana de dados coletados como conjunto de treinamento, segundo os autores, possíveis impostores ou trapaceiros podem ser bem identificados. Entretanto, quando o tamanho do conjunto de treinamento foi aumentado de uma única semana para seis semanas de dados, foi constatado com precisão a identificação do aluno. O aumento foi de 78% para mais de 95% na precisão de identificação.

O modelo de Longi et al. [2015] é inspirado no algoritmo de classificação kNN. Em kNN, os vizinhos são determinados com base em algumas funções de distância. Deste modo, em um cenário de exame, é possível criar um perfil de digitação com base nos exercícios para cada aluno e, em seguida, criar perfil de digitação semelhante nas avaliações. Diante disso, é possível calcular a distância do perfil de digitação dos exercícios para o perfil de digitação das avaliações dos alunos.

O modelo de Longi et al. [2015] foi examinado em mais detalhes por Peltola et al. [2017] que descobriram que é possível identificar estudantes em diferentes contextos textuais, de texto comuns a códigos de programação. Segundo os autores, mesmo que no contexto de identificação em códigos de programação seja menos precisa do que dentro de um outro contexto, ainda é possível identificar o usuário genuíno. Isso indica que a identificação baseada em padrão de digitação seja um método de autenticação confiável, mesmo em cursos onde o conteúdo das atribuições difere: por exemplo, nos cursos de programação com tarefas de codificação.

Por fim, Byun et al. [2020] propuseram uma abordagem para detectar alunos trapaceiros em cursos *on-line* utilizando dinâmica de digitação. Segundo os autores, o método pode ser usado em aulas de introdução à programação com alunos iniciantes. Para isso, foi desenvolvido um protótipo de sistema no qual avaliou o método em uma classe de alunos universitários e mostrou que o sistema detecta casos de fraude com maior precisão do que os estudos anteriores. Como resultados, obtiveram a taxa FAR de 15,6% e a taxa FRR de 14,1%.

Nos trabalhos citados acima, nas sub-seções 3.1, 3.1.1 e 3.1.2, os modelos precisam ser treinados com grande quantidade de dados de cada usuário. Além disso, caso precise adicionar um novo usuário ao sistema, será necessário não apenas de novos dados, mas a arquitetura precisará ser modificada e atualizada. Para resolver esses problemas, trabalhos recentes têm adotado *Deep Learning*, mas especificamente os modelos de redes neurais siamesas.

3.2 Dinâmica da digitação com redes siamesas

Em Giot & Rocha [2019], foi analisada a viabilidade do uso de redes neurais siamesas para autenticação de usuários utilizando dinâmica de digitação em textos estáticos. Segundo os autores, através dessa rede, no contexto da autenticação biométrica, é possível adicionar um novo usuário ao sistema sem coletar inúmeras amostras e treinamento de um modelo complexo. Nos experimentos, o método proposto é comparado a vários modelos da mesma linha na literatura. Seu *Equal Error Rate* (EER) supera o seu melhor trabalho de referência [Killourhy & Maxion, 2009] em 28% em um contexto único (com poucas amostras) e 31% ao usar 200 amostras. Isso prova a viabilidade de tal abordagem e abre o caminho para melhorias para usá-lo em outros contextos de autenticação utilizando a dinâmica da digitação. Entretanto, segundo os autores, é necessário melhorá-lo, a fim de ser utilizável em sistemas que utilizam textos dinâmicos (textos sem tamanho específico).

Em Acien et al. [2020], foi estudada a adequação da dinâmica da digitação em textos dinâmicos para a autenticação de 100.000 usuários. Para isso, foi utilizada uma rede neural recorrente siamesa capaz de autenticar usuários quando a quantidade de dados por usuário é limitada. Nos experimentos, o modelo obteve uma taxa de erro igual de 4,8% usando apenas 5 sequências de recursos da dinâmica da digitação e 1 sequência de teste por usuário com 50 pressionamentos de tecla por sequência. Usando a mesma quantidade de dados por usuário, conforme o número de usuários de teste é aumentado até 100 mil, o desempenho em comparação com 1 mil decai em menos de 5%, demonstrando o potencial uso desse modelo para escalar o número de usuários.

3.3 Discussão

Os métodos apresentados foram classificados de acordo o autor e o ano publicado. A Tabela 3.1 sumariza os trabalhos relacionados incluindo outros fatores, como o classificador empregado, tipos de abordagem (utilizando textos estáticos ou dinâmicos),

contexto, base de dados e o desempenho. Além disso, em negrito é destacado nosso *baseline*, pois este possui uma das menores EER e emprega um modelo RNS.

Tabela 3.1: Sumarização dos trabalhos relacionados.

Autor/ano	Classificador	Tipo de Abordagem	Contexto	Base de dados	Desempenho
Suhail et al. [2019]	SVM	Dinâmica	Sistema computacional	Privada (23 usuários)	44% acurácia
Xiaofeng et al. [2018]	CNN+RNN	Dinâmica	Sistema computacional	Pública (157 usuários)	3,04% EER
Cruz et al. [2017]	Diversos	Dinâmica	Cursos on-line	Privada (17 usuários)	92,44% acurácia
Longi et al. [2015]	k-NN	Estática	Ambiente de programação	Privada (406 usuários)	90% acurácia
Peltola et al.[2017]	k-NN	Estática	Ambiente de programação	Privado (113 usuários)	96% acurácia
Giot & Rocha. [2019]	**RNS**	**Estática**	**Sistema computacional**	**Pública (51 usuários)**	**8% EER**
Acien et al. [2020]	RNS	Dinâmica	Cursos on-line	Pública (100.000 usuários)	9,53% a 3,33% EER

A sumarização dos trabalhos relacionados recentes permite observar que não há pesquisas que utilizam biometria comportamental voltados para o problema de autenticação no contexto específico de sistemas juiz on-line. Além disso, há poucos trabalhos voltados para ambientes de programação. Isso, possivelmente, se dá pelo fato de existirem poucas bases de dados públicas disponíveis. Entretanto, a maioria é utilizada em contextos gerais de sistemas computacionais, incluído cursos on-line, viabilizando assim a adaptação para o contexto específico de sistemas juiz on-line.

Dentre os classificadores empregados, constatou-se que os modelos clássicos são mais populares entre os trabalhos selecionados. Isso se deve principalmente ao grande avanço nesta área, com modelos aplicáveis a diferentes problemas, obtendo resultados eficientes e com alta eficácia. No entanto, essa abordagem possue algumas limitações. Principalmente, por exigir a necessidade de engenharia de atributos na base de dados, neste caso, é necessário construir uma base de dados com estas características, o que é um processo que exige muito esforço manual e a participação de especialistas.

Por fim, embora sejam poucos trabalhos que utilizam aprendizagem profunda como estratégia, a presença desses sugere que é uma abordagem que está surgindo e passível de ser mais explorada futuramente. O uso de aprendizagem profunda pode reduzir a necessidade de extração manual de características, uma vez que o aprendizado da representação dos dados ocorre de maneira automática.

O diferencial deste trabalho em relação aos apresentados é que, além de propor um método de autenticação para ambientes juiz on-line utilizando a dinâmica da digitação, ele também projeta um modelo de rede convolucional siamesa que extrai automaticamente, a partir de dados brutos, novas características necessárias para o reconhecimento dos alunos. Além disso, a arquitetura proposta neste trabalho criará um extrator de característica genérico (mesmo com quantidade limitada de dados) que poderá ser usado para classificar novos alunos, sem a necessidade de retreiná-lo.

Por último, nossa abordagem agrega a informação dos caracteres pressionados junto à dinâmica de digitação, com a finalidade de reconhecer melhor cada padrão biométrico.

3.4 Considerações finais

Neste capítulo, apresentamos uma revisão dos trabalhos relacionados que investigaram métodos de autenticação baseados em biometria comportamental utilizados na autenticação de usuários. Os trabalhos foram divididos de acordo com os autores e ano da publicação. Ao analisarmos os trabalhos, percebemos que existe oportunidade de investigar a utilização da biometria comportamental na autenticação de alunos em ambientes juiz on-line, observando o impacto em eficiência e eficácia em relação a métodos existentes atualmente. No próximo capítulo, será apresentada a solução proposta deste trabalho.

4

Solução Proposta

Este capítulo descreve o método de autenticação de alunos para sistemas juiz online proposto que utiliza biometria comportamental e aprendizagem profunda. Cabe ressaltar que o método proposto é complementar à autenticação inicial de estudantes feita por meio de *login* e senha. A seguir serão apresentadas as etapas necessárias para o desenvolvimento do método de autenticação.

4.1 Visão Geral

A execução do método proposto é feita por meio do reconhecimento de padrões da dinâmica de digitação de cada aluno. Essas características biométricas comportamentais são avaliadas neste trabalho para identificar indivíduos com base em seus ritmos de digitação. A Figura 4.1 fornece uma visão geral do método de autenticação. A metodologia está dividida em quatro etapas: coleta dos dados, pré-processamento, treinamento e autenticação do usuário, a serem detalhadas abaixo:

- **Coleta:** Nessa etapa é feita a coleta dos dados, que inicia a partir do momento em que o aluno faz o *login* e começa a codificar uma solução para um dado exercício de programação através do ambiente de desenvolvimento integrado (IDE) do juiz-online, e prosseguirá até o término da sessão. Os primeiros dados coletados de cada usuário são utilizados para criação de uma espécie de cadastro, em seguida, é executado o treinamento dos dados com o objetivo de gerar o modelo biométrico (identificador único). O modelo biométrico somente é criado a partir do momento que este usuário tenha uma quantidade mínima de dados, conforme será detalhado no próximo capítulo, o restante dos dados são utilizados para testes de autenticação desse usuário.

- **Pré-Processamento:** Uma vez coletados os dados estes são normalizados e segmentados para entrar à rede neural responsável pelo treinamento;

- **Treinamento:** Nesta etapa, é realizado um treinamento de uma rede convolucional siamesa na qual é responsável pela extração automática de novas *features* biométricas oriundas dos dados brutos da dinâmica de digitação de cada aluno. Em seguida, é produzido um modelo genérico que classifica os estudantes com base em seu ritmo de digitação.

- **Autenticação do usuário:** após o treinamento, o modelo gerado na etapa anterior, é utilizado em conjunto com um autenticador biométrico para verificar a autenticidade do aluno.

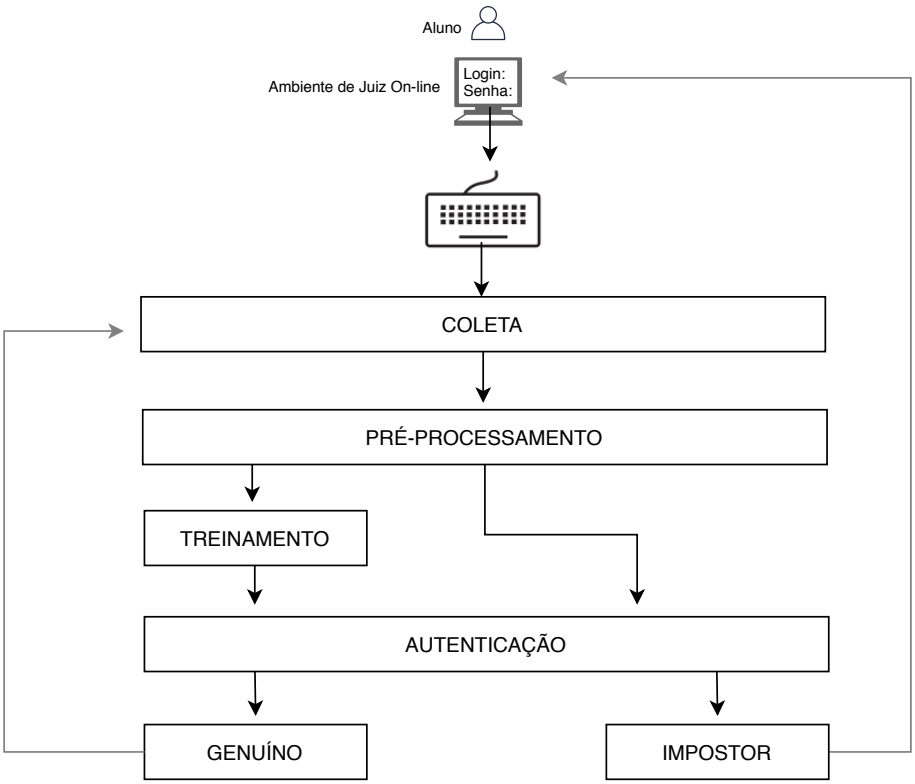

Figura 4.1: Esquema da metodologia proposta.

4.2 Captura de dados

Nesta etapa, a medida em que os estudantes resolvem os problemas de programação, um componente de *software* executando no *background* faz a coleta das ações do teclado realizadas pelos estudantes. Assim, são registrados os dados brutos da digitação dos

alunos feitos em cada instante. Nesta pesquisa, denominamos dados brutos todos dados que não tenham recebido nenhum tratamento após a captura. Os dados brutos são armazenados em uma estrutura simples, conforme ilustrada na Tabela 4.1.

Tabela 4.1: Estrutura dos dados brutos da dinâmica de digitação

Seq	**TpEvent**	**Value**	**Action**	**Time**	**IdActivity**	**IdStudent**
n	K	string	U D	ms	int	int

A Tabela 4.1 mostra o formato de dados bruto capturados no juiz on-line CodeBench, onde **Seq** representa a sequência da ocorrência dos eventos; **TpEvent** representa o tipo do evento, que no caso é sempre 'K' (eventos de teclado); **Value** informa qual tecla foi pressionada ou liberada; **Action** indica se o evento é pressionamento de tecla (D) ou liberação de tecla (U); **Time** registra a data e hora (com milissegundos) do evento, com um intervalo de amostragem de 16 ms; **IdActivity** informa o identificador da atividade executada pelo aluno; e por último, **IdStudent** informa um identificador numérico anonimizado do aluno.

Cada aluno, durante a execução de suas atividades da disciplina, têm seus dados biométricos comportamentais da dinâmica de digitação capturados de maneira silenciosa. É importante que o processo de coleta seja discreto, para não gerar interferência na captura. Esta captura serve para formar o conjunto de dados contendo os padrões biométricos comportamentais dos estudantes, mediante a interação do teclado. Os primeiros dados coletados de cada usuário serão utilizados para criação do seu perfil biométrico (via treinamento de um classificador), e o restante dos dados serão utilizados para testes de autenticação. Dessa forma, as amostras compõem dois conjuntos distintos: treinamento e autenticação (ou teste).

4.3 Pré-Processamento

Nesta etapa, é feito o pré-processamento dos dados brutos capturados na etapa anterior. Inicialmente, somente as informações de tempo de pressionamento (TP), latências entre teclas (DD e UD) e informação textual de codificação são utilizadas (key). Como Redes neurais artificiais esperam que os dados de *input* estejam no formato numérico, os dados textuais devem ser codificados em números antes de serem utilizados para treinar e avaliar um modelo. Logo, nesta etapa, as informações textuais (key) são codificadas em valores numéricos. Em seguida, os dados são normalizados, de forma que os valores das variáveis sejam dispostos dentro de uma escala de valores. Para isso, os dados são

normalizados pela normalização *z-scores* na qual é baseada na média e desvio padrão do atributo, conforme a fórmula abaixo:

$$X' = \frac{X - \mu}{\sigma} \tag{4.1}$$

Onde X' é o valor normalizado, μ é a média do atributo X e σ é o desvio padrão. Desta forma, os dados são normalizados na mesma ordem de grandeza, com intuito de evitar sobre ponderar uma variável em relação a outra.

4.3.1 Segmentação dos dados

Com os dados sendo capturados constantemente pelo sistema, é importante que os mesmos sejam segmentados, isto é, particionados em segmentos de tamanhos fixos aplicado um algoritmo de janelas deslizantes. Assim, cada janela corresponde a uma série temporal de eventos.

Por exemplo, conforme ilustrado na Figura 4.2, quando o usuário pressiona uma sequência fixa de 30 caracteres, a janela deslizante avança um passo de 15 caracteres (equivalente a 50% de *overlap*) e o modelo produz um novo segmento. Cada segmento será utilizado para treinar o classificador, e posteriormente, em tempo de inferência, estes segmentos serão utilizados para autenticar o aluno. Dessa forma, à medida que a janela desliza, o sistema de autenticação pode continuar verificando a identidade do usuário.

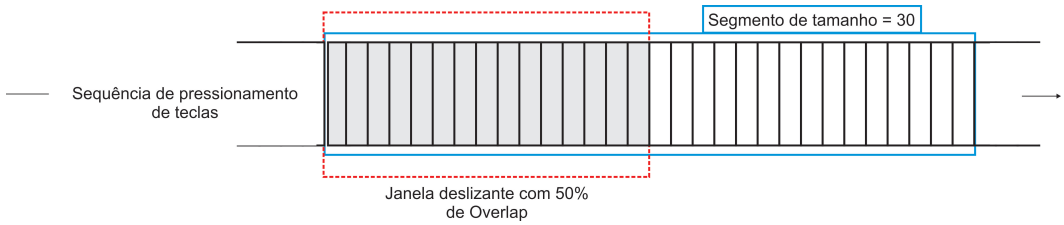

Figura 4.2: Janela deslizando.

4.4 Arquitetura da rede neural convolucional siamesa proposta

Em sistemas de autenticação, normalmente são definidas duas classes de usuários: legítimo e impostor. Entretanto, primeiro é treinado um algoritmo de aprendizagem profunda para criar um modelo genérico capaz de transformar os dados segmentados em vetores densos (*embeddings*) que melhor representem os padrões comportamentais

dos usuários. Os vetores de *embeddigns* podem ser interpretados como uma assinatura única do usuário. Estes vetores são novas *features* extraídas pela rede neural siamesa, e podem ser utilizados por um classificador *shallow* que realiza a autenticação final.

Nesta pesquisa, é implementada e treinada uma rede convolucional siamesa que tem como entrada dados brutos do ritmo da digitação dos alunos. Nesse cenário, a hipótese é que redes siamesas treinadas com a função de perda *Triplet Loss* geram melhores *embeddings*, e que estes irão ajudar na identificação dos alunos. Para que a função *Triplet Loss* guie o treinamento da rede siamesa são necessárias três entradas:

- Âncora (A): uma amostra pertencente a um usuário (usuário genuíno);
- Positiva (P): uma amostra semelhante à amostra âncora (usuário genuíno); e
- Negativo (N): uma amostra pertencente a um usuário impostor.

Como apresentado no capítulo 2, o objetivo da *Triplet Loss* é minimizar a distância entre *embeddings* da mesma classe enquanto maximiza a distância entre *embeddings* de classes diferentes. No caso desta pesquisa, pressupomos como parte de nossa hipótese que há uma menor distância entre os *embeddings* A e P, se pertencentes a um mesmo estudante (usuário genuíno), quando comparada à distância entre A e N. Caso contrário, se a distância entre A e P é maior que A e N, isto pode indicar aluno impostor.

Entretanto, gerar todas as triplas de A, P e N possíveis resultaria em uma abordagem com custo exponencial em relação à quantidade de exemplos de treinamento, causando uma convergência mais lenta do treinamento. Nesse caso, é crucial selecionar triplas A, P e N que possam contribuir com melhores informações e acelerar o treinamento do modelo. Nesta pesquisa, é utilizada a estratégia de seleção de triplas conhecida como *Triplet* semi-difícil, que emprega uma estratégia de escolha em tempo de treinamento, apresentando à rede exemplos não tão difíceis que colapsem o treinamento do modelo, nem tão fáceis que evitem ajustar o modelo. Conforme Schroff et al. [2015], esta versão apresenta melhores resultados do que realizar um treinamento com seleção aleatória das triplas A, P e N.

Uma vez que o modelo de rede siamesa for treinado, podemos executar a extração de *embeddings*, conseguindo mapear as janelas produzidas pelo segmentador em vetores densos com os quais podemos treinar um classificador que indica as chances de uma amostra pertencer à mesma classe ou não. A arquitetura rede neural convolucional siamesa proposta é ilustrada na Figura 4.3. Esta é composta por três sub-redes CNN 1D idênticas, que compartilham os mesmos parâmetros. A entrada *input* consiste em

segmentos de amostras da dinâmica de digitação dos usuários que serão mapeados no espaço de *embeddings* aprendido pela rede.

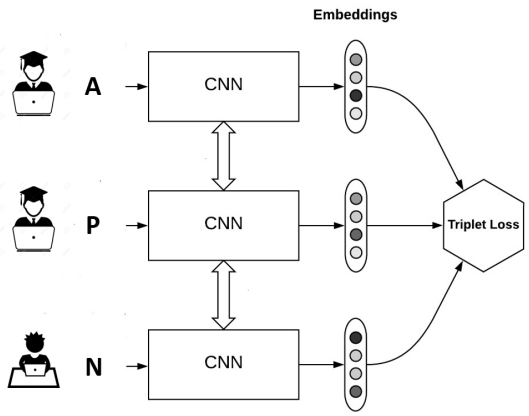

Figura 4.3: Arquitetura da rede neural proposta.

Uma grande vantagem das CNNs 1D é que podem ser implementadas utilizando hardware de baixo custo, devido sua configuração ser simples e compacta. Além disso, as redes CNNs 1D são eficazes na extração de *embeddings* em segmentos de tamanho fixo e normalmente utilizadas em dados de séries temporais.

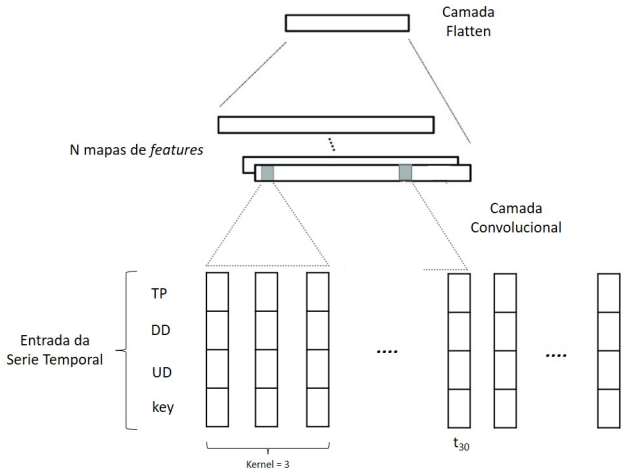

Figura 4.4: Sub-redes CNNs 1D

As sub-redes CNNs 1D, conforme ilustradas na Figura 4.4, possuem diversas camadas convolucionais responsáveis pela extração automática de novas *features* geradas a partir dos dados brutos (TP, DD, UD e key, descritos na seção 5.3) e consideradas relevantes para a identificação do usuário. Assim, não há necessidade de descoberta manual de *features*, uma vez que o aprendizado da representação dos dados ocorre de forma automática. Além disso, nossa arquitetura possui núcleos de convolução que ex-

traem diferentes sequências de *features*, onde é adotado um ***kernel*** de tamanho 3 para produzir um mapa de características. A técnica de ***Batch Normalization*** também é adotada para melhorar a velocidade, o desempenho e a estabilidade da rede. Para fazer a CNN aproximar as funções complexas e induzir não linearidade, é usado a (***ReLu***) como função de ativação. A camada ***Flatten*** também é adotada resultando em um vetor de dados. Por fim, utiliza-se uma normalização do tipo L2 para obter os vetores de *embeddings* finais.

Para o treinamento da rede siamesa foi utilizado o otimizador *Adam(learning_rate= 0.001)*, *batch size = 128* e acurácia como métrica de treinamento. O modelo final foi programado usando a biblioteca *Keras-Tensorflow*.

Uma vez que o modelo foi treinado e executada a etapa de extração de *embeddings* correspondentes a cada usuário, a classificação e a avaliação do autenticador biométrico já podem ser implementados.

4.5 Autenticador biométrico

Dado um modelo da rede siamesa treinado e finalizada a extração dos *embeddings*, conforme ilustrado na Figura 4.5 ainda é preciso de um modelo que possa fazer previsões sobre um determinado par de *embeddings*, ou seja, determinar se eles são do mesmo usuário (y = 1) ou usuários diferentes (y = 0).

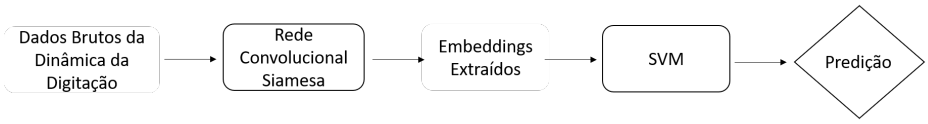

Figura 4.5: Pipeline da abordagem proposta.

Para criar esse modelo, é treinado uma Máquina de Vetores de Suporte (SVM) com decomposição um-contra-todos (One-vs-all), que faz as previsões. Conforme ilustrado na Figura 4.6, essa estratégia consiste em treinar um classificador por classe. Para cada classificador, os *embeddings* da classe são comparados em relação aos *embeddings* de todas as outras classes (N classes), e as previsões são feitas usando o modelo com a saída de maior confiança. Por exemplo, a classe N_i é rotulada como positivo (aluno genuíno) enquanto que o restante como negativo (aluno impostor). Desta forma, o SVM aprende diferenças elementares entre pares de *embeddings*, elemento a elemento), ou seja, 1 se os *embeddings* são do mesmo usuário e 0 se os *embeddings* forem de usuários diferentes.

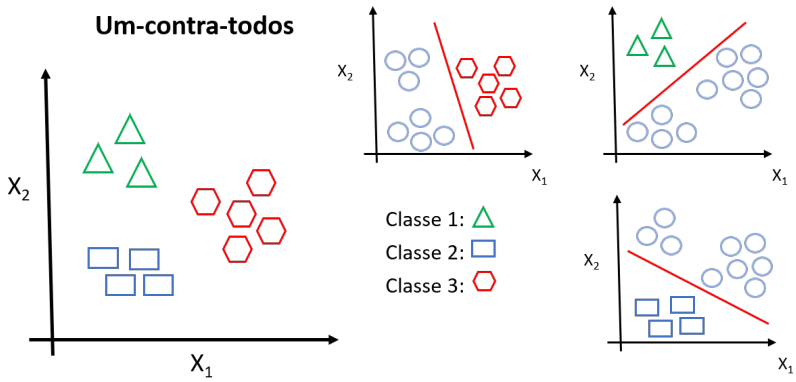

Figura 4.6: Abordagem de decomposição um-contra-todos.

Assim, o autenticador proposto neste trabalho consiste em verificar se uma sequência de pressionamento de teclas corresponde ao padrão comportamental de um aluno logado no sistema juiz on-line. As amostras de texto digitadas pelo referido usuário, já segmentadas, são inseridas ao modelo de autenticação treinado, em seguida, será produzido um vetor de probabilidades referentes ao reconhecido ou não do usuário. As métricas FAR e FRR de cada modelo são calculadas durante a autenticação (teste). Desta forma, o modelo de autenticação precisa ser treinado previamente, em seguida, armazenado em uma base de conhecimento prévio, pois o mesma será utilizado sempre que for necessária a autenticação do usuário nas novas atividades.

4.6 Considerações finais

Como citado anteriormente, o método proposto será um adicional ao método tradicional de login e senha. Além disso, funcionalidades de bloqueio ou alerta, que apoiem os professores ou monitores, caso o aluno seja identificado como impostor, precisam ser providenciadas pela equipe de gestão da instituição.

Neste capítulo, descrevemos a solução proposta para autenticação de alunos em ambientes juiz on-line, utilizando biometria comportamental e aprendizagem profunda. No próximo capítulo, descreveremos experimentos e resultados.

5

Experimentos e Resultados

Este capítulo descreve os experimentos e resultados alcançados. O capítulo inicialmente detalha os materiais utilizados, como as bases de dados e a arquitetura da rede. Em seguida, são descritos os experimentos e resultados. Por fim, apresentamos as considerações finais.

5.1 Base de dados I

A primeira base de dados utilizada nesta pesquisa é pública e altamente citada em pesquisas relacionadas à dinâmica de digitação, chamada de *CMU Keystroke Dynamics – Benchmark Data Set*[1]. Esta base de dados contém dados de 51 pessoas distintas que foram convidadas a digitar a senha ".tie5Roanl" 400 vezes cada uma, sendo 50 vezes por sessão, resultando no total de 8 sessões. Foi levado em consideração um dia de folga entre as sessões com objetivo de capturar a variação de padrões de digitação ao longo de dias.

Nesta base de dados, cada sessão de coleta capturou três tipos de *features* oriundas de textos de tamanho fixo (senha), são elas o tempo de pressionamento (TP) e as duas latências entre as teclas (DD e UD). Apesar desta base de dados possuir dados de um contexto geral e não de sistemas juiz on-line, a escolha da mesma se dá pelo fato de inúmeros trabalhos de referência, inclusive os artigos base desta pesquisa a utilizarem em seus experimentos. Além disso, permite validar o método proposto no contexto de autenticação utilizando dados de tamanho fixo. No mais, não foi necessário nenhum tipo de pré-processamento nesta base de dados.

[1]https://www.cs.cmu.edu/ keystroke/

5.2 Base de dados II

A segunda base de dados desta pesquisa é própria e foi coletada nas quatro primeiras semanas de aula de sete turmas da disciplina de Introdução à Programação dos Computadores da Universidade Federal do Amazonas - UFAM. Essas turmas foram ofertadas no segundo semestre de 2019 para diferentes cursos de graduação, e possuíam um total de 260 estudantes. Neste período, os estudantes resolviam problemas de programação diretamente na IDE do juiz on-line CodeBench. Os conteúdos estudados nas quatro primeiras semanas de aula foram: (a) variáveis e (b) estrutura de programação sequencial. Além disso, os alunos tiveram acesso a 5 atividades (listas de exercícios) e foram submetidos a uma avaliação no final da quarta semana.

Para a aquisição dos dados da dinâmica de digitação de cada estudante, ao aluno efetuar o *login* e acessar a IDE do juiz on-line, um componente de *software* iniciou a coleta das ações do teclado desempenhadas dentro da IDE. Logo, na medida em que os alunos desenvolviam suas soluções para os exercícios de programação disponibilizados pelos professores, as ações desempenhadas durante o processo de digitação eram registradas a cada instante de maneira silenciosa. Os dados de registro foram armazenados no servidor em arquivos *logs*, que posteriormente foram convertidos em arquivos *csv*. A ferramenta de registro também capturou algumas informações adicionais relacionadas às atividades dos alunos, como por exemplo o código da atividade, dentre outras mostradas na Figura 5.1.

A coleta dos dados é uma tarefa fundamental para autenticação em ambientes juiz on-line, pois nesses dados está contido o padrão biométrico da dinâmica da digitação de cada usuário do sistema, registrado nas colunas *down* e *up*. Logo, mediante a interação do teclado, é possível autenticá-los de maneira não intrusiva em ambiente juiz on-line.

5.3 Pré-Processamento

Neste trabalho, foram considerados quase todos eventos de teclado, por exemplo, eventos de teclado simples (ou seja, eventos que envolviam uma única tecla do teclado) e qualquer outro evento envolvendo duas ou mais teclas simultâneas. Exceto, eventos de copiar (Ctrl+c) e colar (Ctrl+v). Além disso, foram eliminados eventos com longos intervalos de tempo, por exemplo, atividades iniciadas em um determinado dia e finalizadas em outro, nesse caso, foi limitado um período de tempo de 2 horas no máximo de intervalo. Ademais, neste estudo não foram incluídos dados de estudantes que desistiram da disciplina nos primeiros dias de aula, ou alunos que utilizam diferentes

ambientes de programação além do CodeBench. Após essa etapa, restaram dados de 158 alunos, dos quais foram selecionados 42, por fazerem parte da mesma turma.

IdStudent	IdClass	IdActivity	data	down	up	key	TP	DD	UD
2719	259	1	2019-8-20	22:48:17.431	22:48:17.527	p	0.096	0.328	0.232
2719	259	1	2019-8-20	22:48:17.759	22:48:17.855	r	0.096	0.089	0.007
2719	259	1	2019-8-20	22:48:17.848	22:48:17.935	i	0.087	0.151	0.064
2719	259	1	2019-8-20	22:48:17.999	22:48:18.135	n	0.136	0.104	0.032
2719	259	1	2019-8-20	22:48:18.103	22:48:18.216	t	0.113	0.456	0.343

Figura 5.1: Dados brutos da dinâmica de digitação capturados pela IDE do CodeBench representando as *Features* biométricas dos usuários.

A característica mais comum dos dados capturados é o tempo de pressão entre duas teclas [Cruz et al., 2017]. Essa *feature* é obtida pela subtração entre o tempo em que uma tecla é liberada pelo aluno e o tempo em que essa mesma tecla inicialmente foi pressionada (*Up-Down*). Conforme mostradas na Figura 5.1, para os experimentos realizados nesta pesquisa utilizando esta base de dados, somente as *features* tempo de pressionamento (TP), latência entre teclas (DD e UD) e caracteres brutos (key) digitados no juiz on-line foram utilizadas como *features* biométricas dos usuários.

Quanto aos caracteres (key) digitados, foram selecionados exclusivamente os 110 mais frequentes em todos os códigos. A hipótese é que quando acrescentamos os caracteres pressionados à dinâmica da digitação estes agreguem informação e melhorem o reconhecimento dos alunos. Além disso, antes da construção do modelo de aprendizado profundo, foi preciso codificar todos os atributos da variável key, que eram categóricos, em valores numéricos, pois a maioria dos algoritmos de aprendizado de máquina trabalham exclusivamente com dados numéricos. Na Figura 5.2 mostra a codificação realizada, onde um valor numérico é atribuído a cada um dos valores categóricos da variável key. Para isso, foi utilizado o recurso *LabelEncoding* da biblioteca *scikit-learn*, que faz codificação de forma automática.

IdStudent	IdClass	IdActivity	data	down	up	key	TP	DD	UD
2719	259	1	2019-8-20	22:48:17.431	22:48:17.527	98	0.096	0.328	0.232
2719	259	1	2019-8-20	22:48:17.759	22:48:17.855	100	0.096	0.089	0.007
2719	259	1	2019-8-20	22:48:17.848	22:48:17.935	91	0.087	0.151	0.064
2719	259	1	2019-8-20	22:48:17.999	22:48:18.135	96	0.136	0.104	0.032
2719	259	1	2019-8-20	22:48:18.103	22:48:18.216	102	0.113	0.456	0.343

Figura 5.2: *Features* biométricas dos usuários.

Em seguida, os dados foram analisados e normalizados. Como a coleta dos dados para verificação foi feita de forma contínua, enquanto o aluno programava, foi necessário particionar os *logs* em segmentos de tamanhos fixos menores. Neste trabalho, foram analisados segmentos de tamanhos 10, 30, e 50. Esses tamanhos equivalem a quantidade caracteres consecutivos digitados pelo aluno. Em seguida, os *embeddings* são extraídos de forma automática pela rede neural descrita no capítulo anterior.

5.4 Protocolos de treino e validação

Nesta etapa, foram considerados quatro protocolos diferentes de separação dos dados para avaliar o método de autenticação proposto. Estes protocolos são denominados de "Protocolo I", "Protocolo II", "Protocolo III" e "Protocolo IV".

5.4.1 Protocolo I

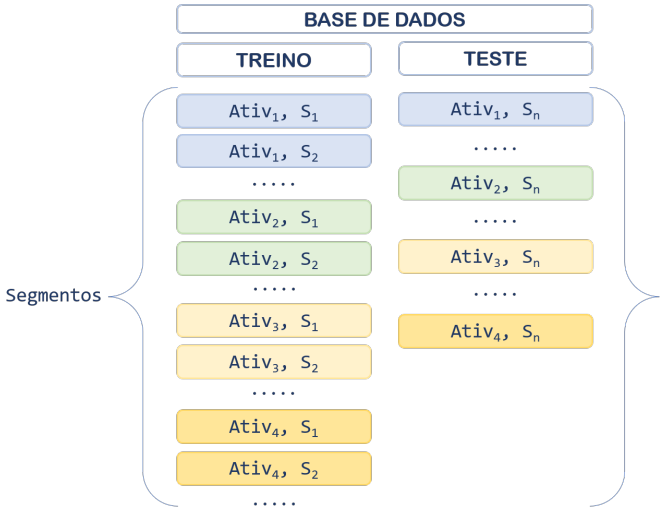

Figura 5.3: Protocolo I.

No "Protocolo I", conforme mostrado na Figura 5.3, onde $Ativ_n$ representa o número da atividade e S_n representa o número do segmento, os dados (segmentos) das cinco atividades realizadas no juiz on-line dos 42 usuários são usados para treino e para teste, mas especificamente 70% dos segmentos de dados de cada atividade para treino e 30% para teste. Ou seja, são separados segmentos da mesma atividade para treino e teste. Para efetuar essa divisão foi utilizada a biblioteca *scikit-learn* da linguagem *Python*, que possui a função *train-test-split()* capaz de escolher de forma aleatória os segmentos da dinâmica de digitação utilizados para o treinamento e para teste.

Com a utilização deste Protocolo, podemos investigar como a presença de amostras (segmentos) das mesmas atividades nos conjunto de treino e teste impactam na acurácia. Este geralmente é o protocolo padrão utilizado em aprendizagem de máquina e inclusive por algumas referências bibliográficas, mas não se ajusta ao contexto específico do juiz on-line, uma vez que segmentos do mesmo aluno e da mesma atividade não deveriam estar misturados nos dois conjuntos (treino e teste).

5.4.2 Protocolo II

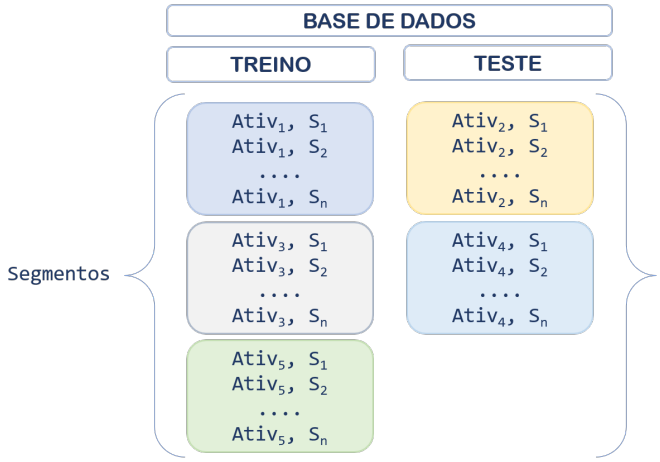

Figura 5.4: Protocolo II.

No segundo protocolo, "Protocolo II", as cinco atividades $Ativ_n = \{Ativ_1, Ativ_2, Ativ_3, Ativ_4, Ativ_5\}$ realizadas no juiz on-line pelos 42 usuários são distribuídas de forma aleatória para treino e para teste. Entretanto, conforme ilustrada na Figura 5.4, diferente do protocolo anterior, segmentos de dados das atividades separadas para o treinamento não são utilizados no teste. Por exemplo, se utilizados segmentos da atividade $Ativ_1 = \{s_1, s_2, \ldots, s_n\}$ do usuário u_n para o treinamento, segmentos de dados desta atividade e deste mesmo usuário não poderão ser utilizados no teste. A divisão é realizada da seguinte forma: dados de 3 atividades são para treino e 2 para teste, de forma manual utilizando a linguagem *Python*.

Através deste Protocolo, é possível investigar como a distribuição de amostras de atividades diferentes nos conjunto de treino e teste impactam na acurácia em relação ao "Protocolo I". Desta forma, será possível verificar se existe vazamento de informações durante o treino ao se utilizar o "Protocolo I".

5.4.3 Protocolo III

O "Protocolo III", possui configurações específicas para ambientes juiz on-line onde considera a evolução temporal da aprendizagem dos alunos. Através dele, será possível verificar se os padrões biométricos dos alunos mudam no decorrer do tempo à medida em que eles realizam as atividades no ambiente juiz on-line. Neste protocolo, são utilizados todos os dados das cinco atividades realizadas no juiz on-line CodeBench de todos os 42 usuários em treino e teste, respeitando a ordem cronológica. A divisão dos dados é feita de forma manual utilizando a linguagem *Python*.

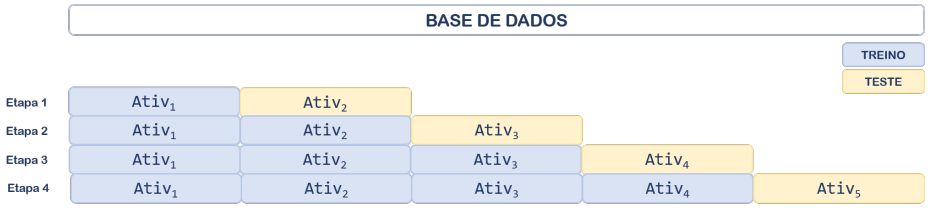

Figura 5.5: Protocolo III

Conforme ilustrada na Figura 5.5, este protocolo separa primariamente todos os dados da primeira atividade ($Ativ_1 = \{s_1, s_2, \ldots, s_n\}$) de cada usuário para treinamento e, os segmentos da próxima atividade realizada ($Ativ_2 = \{s_1, s_2, \ldots, s_n\}$) para teste (autenticação). Em seguida, é calculada a acurácia do modelo. Incrementalmente, na próxima etapa, separa todos os dados da primeira ($Ativ_1 = \{s_1, s_2, \ldots, s_n\}$) e segunda atividade ($Ativ_2 = \{s_1, s_2, \ldots, s_n\}$) para treino e, os dados da próxima atividade ($Ativ_3 = \{s_1, s_2, \ldots, s_n\}$) para teste, e assim sucessivamente, até que todas as atividades tenham sido utilizadas para treinamento, exceto a última atividade ($Ativ_5$), desta forma, considerando o aspecto temporal.

5.4.4 Protocolo IV

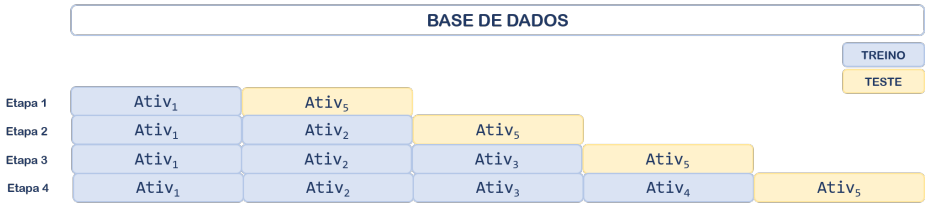

Figura 5.6: Protocolo IV

Por fim, no "Protocolo IV", são separando todos os dados da primeira atividade ($Ativ_1 = \{s_1, s_2, \ldots, s_n\}$) para treinamento e, os dados da última atividade

($Ativ_5 = \{s_1, s_2, \ldots, s_n\}$) para teste. Em seguida, é calculada a acurácia do modelo. Incrementalmente, na próxima etapa, separa todos os dados da primeira ($Ativ_1 = \{s_1, s_2, \ldots, s_n\}$) e segunda atividade ($Ativ_2 = \{s_1, s_2, \ldots, s_n\}$) para treino e, os dados da última atividade ($Ativ_5 = \{s_1, s_2, \ldots, s_n\}$) para teste, e assim sucessivamente, sempre considerando o aspecto temporal, conforme ilustrada na Figura 5.6. Este protocolo é útil para simular a aplicação de uma prova que neste exemplo seria a quinta atividade ($Ativ_5$).

O Protocolo I, protocolo de treinamento aleatorizado tradicional, é utilizado na maioria dos *baselines*. Entretanto, não é recomentado para sistemas juiz on-line, uma vez que o mesmo não leva em consideração a ordem cronológica da realização das atividades, nem a evolução dos padrões biométricos. Destacamos que, a hipótese avaliada nestes dois últimos protocolos relacionasse com a observação de que o padrão de digitação dos alunos muda conforme eles começam a se familiarizar com a linguagem e com a IDE de programação, assim, quanto mais habituados à linguagem os alunos se tornam mais rápidos ao digitar palavras que aparecem frequentemente nos programas. Por exemplo, digitar sucessivamente o comando "print()" leva a uma evolução no padrão de digitação, e portanto altera a biometria comportamental. Todavia, apesar dos protocolos II e III e IV possuírem configurações específicas para esses ambientes, nada impede de serem adaptados para outros contextos, por exemplo o reconhecimento de senhas no *login*.

5.5 Configurações dos modelos

Em seguida, para treinar o modelo de rede neural siamesa proposto na Seção 4.3 foi utilizada a abordagem de *triplet semihard online* descrita na Seção 2. Nessa abordagem, as triplas de exemplos são geradas durante a fase de treinamento utilizando exemplos de lote de dados (do *minibach* de 64). Em outras palavras, as triplas (A, P, N) são encontradas dentro do lote de dados em tempo de treinamento e não gerados *offline*, gerando um treinamento mais eficiente.

Para fazer previsões sobre um determinado par de *embeddings*, ou seja, determinar se eles são do mesmo usuário (y = 1) ou usuários diferentes (y = 0). Foi treinado uma Máquina de Vetores de Suporte (SVM) em conjunto ao método de decomposição um-contra-todos (One-vs-All), que fez as previsões. Ou seja, 1 se os *embeddings* são do mesmo usuário e 0 se os *embeddings* forem de usuários diferentes. Os melhores resultados foram encontrados utilizando os hiperparâmetros *gamma=0.1*, *kernel='rbf'*, e $C = 1$.

5.6 Experimentos e Resultados I

5.6.1 Experimentos I

Um dos principais fatores que influenciam o desempenho de um algoritmo de autenticação é a quantidade de amostras na segmentação [Xiaofeng et al., 2019]. Deste modo, o primeiro experimento foi conduzido para avaliar a quantidade de amostras necessárias dentro do problema proposto, dividindo-as em segmentos de tamanhos iguais.

Para isso, foram avaliados os seguintes tamanhos de segmentos fixos: T = {10, 30 e 50}, onde T é o tamanho do segmento fixo inserido no modelo para treinamento e teste. Foi utilizado janelas deslizantes com 50% de *overlap* e considerando somente dados da base de dados II. Para este experimento, foram selecionadas somente *features* da dinâmica da digitação (TP,DD e UD). Além disso, foi utilizado o protocolo I (protocolo tradicional, descrito na seção 5.4.1), onde separa os dados de todas as atividades realizadas no juiz on-line CodeBench dos 42 usuários em 70% dos segmentos de dados de cada atividade para treino e 30% para teste.

Tabela 5.1: Acurácias e Taxas de Erro (%) alcançadas para diferentes tamanhos de segmentos

Tamanho Segmento	**Acurácia**	**EER**
10	78,4	0,042
30	**81,1**	**0,021**
50	72,7	0,047

A Tabela 5.1 resume a acurácia e taxas de erro alcançadas para diferentes valores de segmentos no experimento. Conforme podemos observar na tabela, os melhores resultados são alcançados para segmentos fixos de tamanho = 30, com acurácia de 81,1%, e taxa de erro de 0,021%. Também podemos observar que para sequências de T = 10 e 50 não há melhora significativa nos resultados. Assim, sequências de T = 10 que apesar do resultado com acurácia de 78,4%, resultado satisfatório, talvez não tenha a quantidade de dados (informações biométricas) suficiente para aprender completamente o padrão biométrico da dinâmica de digitação do usuário, enquanto que sequências de T = 50 tenha bastante dados, ao ponto de conter ruídos e consequentemente influenciar nos resultados, nesse caso, diminuindo a acurácia. Cabe destacar que cada usuário possui quantidades variadas de segmentos, uma vez que cada aluno pode digitar quantidades diferentes de caracteres comparado a outro aluno que realizou a mesma atividade. Neste experimento não foram utilizados os caracteres digitados.

5.6.2 Experimentos II

Neste experimento, foi verificado o impacto nos resultados quando são incorporados os caracteres digitados como sendo uma das *features* de entrada para a rede siamesa. Como citado na seção 2, a biometria tradicional considera apenas os tempos de pressionamento, tornado difícil para a rede neural distinguir se letras diferentes, mas com latências parecidas, fazem parte do perfil biométrico do aluno, uma vez que existem muitas letras e ordens que se repetem em programação. Portanto, neste experimento, além dos tempos de pressionamento entre as teclas, incorporamos os caracteres pressionados.

Para isso, foram utilizados segmentos de tamanho 30, janelas deslizantes com 50% de *overlap* e considerando somente dados da base II. Além disso, foi utilizado o Protocolo I de separação de dados com partição: 70% dos segmentos de dados de cada atividade para treino e 30% para teste. A Tabela 5.2 apresenta os resultados quando utilizada somente as *features* da dinâmica da digitação e quando acrescentado informação do caractere pressionado (key).

Tabela 5.2: Acurácias e Taxas de erro (%) obtidas a partir da quantidade de features

Features	**Acurácia**	**EER**
TP, DD e DU	88,1	0,021
TP,DD,UD + Key	**98,0**	**0,003**

Observamos que os resultados melhoram consideravelmente quando acrescentamos os caracteres pressionados à dinâmica da digitação. Ou seja, a informação dos caracteres pressionados agregam informação à dinâmica de digitação e melhora a biometria comportamental.

5.6.3 Experimentos III

Desta vez, comparamos a arquitetura proposta com trabalhos relacionados recentes. Entretanto, há algumas diferenças entre ambas. Por exemplo, os autores do *baseline* Giot & Rocha [2019] desenvolveram um modelo de rede neural siamesa que tem como base uma rede MLP (*Multi-layer perceptron*), que é diferente da arquitetura proposta, na qual desenvolvemos uma rede CNN 1D como base. Escolhemos uma rede CNN 1D pelo fato das entradas dos caracteres digitados possuírem uma ordem temporal (uma série de tempo), característica ignorada pelas camadas MLP. Além disso, neste trabalho foi utilizado textos livres, ou seja, qualquer texto digitado pelo usuário, não importando a quantidade de caracteres (por exemplo, cada aluno codifica de acordo

com sua lógica, deste modo, o tamanho dos códigos variam, enquanto que os autores do *baseline* utilizaram textos de tamanho fixo como, por exemplo, uma senha.

Primeiramente, utilizamos dados da base II, considerando o Protocolo I de separação para treinamento e teste (protocolo padrão), de forma a realizar comparações justas. Os dois modelos são treinados com quantidade de usuários iguais (40 usuários) e segmentos fixos de tamanhos iguais (tamanho = 30). Na Tabela 5.3, são apresentadas as acurácias e taxas de erro obtidas pelas duas arquiteturas. Podemos observar que a arquitetura apresentada neste trabalho supera o *baseline*. Com acurácia de 98,0%, e taxa de erro de 0,003%. Desta forma, observamos que a rede convolucional 1D siamesa extraiu características robustas dos dados brutos da dinâmica da digitação dos alunos, levando a resultado superior no reconhecimento dos usuários. Concluímos também que a capacidade de processar vetores de *features* que formam séries de tempo não pode ser ignorada pelas soluções propostas ou existentes.

Tabela 5.3: Acurácias e Taxas de erro (%) obtidas a partir das duas arquiteturas nas duas bases de dados comparadas contra o baseline.

Base de dados	Arquitetura	Acurácia	EER
Base I	Giot & Rocha [2019]	95,0	0,010
	CNN 1D Proposta	**99,4**	**0,002**
Base II	Giot & Rocha [2019]	92,6	0,017
	CNN 1D Proposta	**98,0**	**0,003**

Em seguida, comparamos as duas arquitetura, desta vez utilizando dados da base de dados I, base de dados utilizada pelo *baseline* e altamente citada em pesquisas relacionadas a dinâmica de digitação, onde as amostras que compõem a base são de textos de tamanho fixo, nesse caso, senhas. Conforme podemos observar na Tabela 5.3, a arquitetura proposta neste trabalho também supera o trabalho *baseline*, mesmo utilizando textos de tamanho fixo e de contexto geral. Os melhores resultados são alcançados para segmentos fixos de tamanho = 30, com acurácia de 99,4%, taxa de erro de 0,002% e considerando o Protocolo I para separação de dados de treino e teste. Com isso, fica evidente a eficácia do método de autenticação proposto também para textos de tamanho fixo e de contexto geral. De acordo com a nossa revisão bibliográfica, o presente trabalho supera o estado da arte para métodos de autenticação utilizando textos fixos da dinâmica de digitação e redes siamesas.

5.6.4 Experimentos IV

Desta vez, verificamos como a distribuição de amostras da mesma atividade nos conjuntos de treino e teste impactam na acurácia. Para isso, inicialmente, executamos e comparamos os resultados quando utilizados os Protocolo I e Protocolo II para separar os dados de treino e teste. Cabe ressaltar, que no Protocolo I há segmentos da mesma atividade nos conjuntos de treino e teste. Em contrapartida, no Protocolo II não há segmentos da mesma atividade nos dois conjuntos (treino e teste), a divisão dos dados foi feita de forma manual, embora a seleção das atividades na forma aleatória. Ambos os protocolos são descritos na seção 5.4. Desta forma, foi possível verificarmos como a distribuição dos segmentos das atividades no conjunto de treino e teste impactam na acurácia e, se os padrões biométricos mudam mediante a escolha dos protocolos de separação de dados. Neste experimento, os melhores resultados são alcançados para segmentos fixos de tamanho = 30, com acurácia de 98,1% e taxa de erro de 0,003%, janelas deslizantes com 50% de *overlap* e considerado somente dados da base de dados II.

Tabela 5.4: Acurácias e Taxas de Erro (%) alcançadas utilizando diferentes Protocolos de separação de dados

Protocolos	**Acurácia**	**EER**
Protocolo I	98,1	0,003
Protocolo II	80,4	0,040

Conforme mostrado na Tabela 5.4, podemos observar que quando considerado o Protocolo I para separação de dados, onde há segmentos da mesma atividade no treino e teste, o reconhecimento possui acurácia de 98,1%. Entretanto, esse protocolo de separação de dados não se adéqua no contexto específico do juiz on-line, uma vez que segmentos do mesmo aluno e da mesma atividade estão misturados nos dois conjuntos (treino e teste). Por outro lado, quando considerado o Protocolo II, onde não têm segmentos da mesma atividade no treino e teste, a acurácia no reconhecimento é 80,4%, mostrando uma diferença significativa em relação ao resultado do Protocolo I. Esta diferença indica que há um vazamento de informações durante o treino ao se utilizar o Protocolo I. Além disso, fica evidente que a distribuição de amostras das atividades nos conjunto de treino e teste impactam na acurácia, além de que, a efetividade no reconhecimento dos padrões biométricos mudam mediante a escolha dos protocolos de separação de dados.

5.6.5 Experimentos V

Neste experimento, verificamos a viabilidade do método proposto levando em consideração a evolução temporal da aprendizagem dos alunos, investigando se os padrões biométricos dos mesmos mudam no decorrer do tempo à medida em que eles realizam as atividades no ambiente juiz on-line.

Desta vez, utilizamos o Protocolo III (descrito na seção 5.4) que possui configurações específicas para juiz on-line. Para isso, foram utilizados segmentos de tamanho 30, janelas deslizantes com 50% de *overlap* e considerando somente dados da base de dados II. Além disso, a divisão dos dados foi feita de forma manual utilizando a linguagem *Python*.

Neste experimento, inicialmente utilizamos os dados da primeira atividade de todos os usuários para treino e os dados da próxima atividade (segunda atividade) dos mesmos usuários, para teste. Em seguida, foi calculada a acurácia do modelo. O processo se repetiu de maneira incremental, até que todas as atividades, exceto a última, tenham sido utilizadas para treinamento, só então, foi calculada a média das acurácias, levando em consideração os valores de cada etapa.

Tabela 5.5: Acurácias e Taxas de Erro (%) alcançadas para diferentes quantidades de atividades utilizadas para treinamento

Atividade(s) (Treinamento)	**Atividade (Teste)**	**Acurácia**	**EER**
1	2	76,1	0,044
1 e 2	3	82,2	0,038
1,2 e 3	4	82,7	0,038
1,2,3 e 4	5	83,1	0,037
MÉDIA		**81,0**	**0,039**

Os resultados alcançados para diferentes quantidades de atividades utilizadas para treinamento são mostrados com detalhes na Tabela 5.5. Como podemos observar na tabela, os melhores resultados são alcançados quando utilizados 4 atividades para treinamento, com acurácia de 83,1% e taxa de erro de 0,037%. Entretanto, caso utilizado unicamente amostras da primeira e segunda atividades para compor o conjunto de treinamento e, consequentemente criar o perfil biométrico do aluno, o método reconheceria os alunos genuínos com 82,2% acurácia. Nesta pesquisa, perfil biométrico é definido como a quantidade mínima de atividades necessárias para criar o modelo e ajustá-lo.

A Figura 5.7 ilustra a curva ROC do resultado quando utilizado as 4 primeiras atividades para treinamento e a 5 para teste. Observe que quanto mais a curva

ROC se aproxima do canto superior esquerdo, melhor o teste quanto à capacidade para discriminar os grupos. Além disso, podemos observar a variância em torno dela (áreas cinzas) na qual indica quão estável é o modelo, desta forma demostra resultado satisfatório.

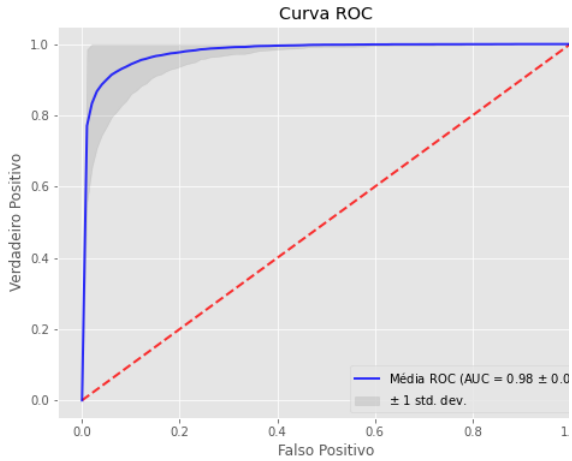

Figura 5.7: Curva Roc

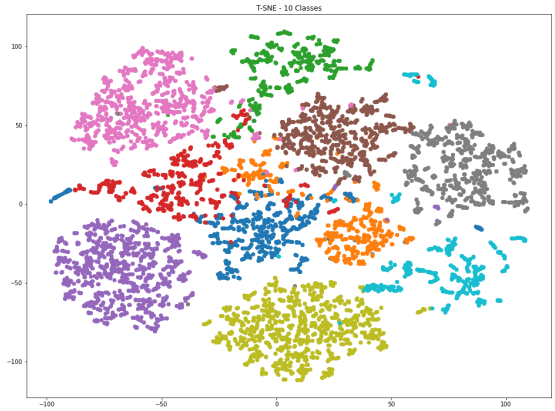

Figura 5.8: T-SNE de 10 usuários

Sobre a visualização dos *embeddings* dos usuários, o espaço de *embeddings* considera todas as dimensões que existem nos vetores que o compõem, logo, a visualização desse espaço ficaria impraticável sem a utilização de um método que viabilizasse isso, nesse caso, um método de redução de dimensionalidade. Assim, na Figura 5.8 mostra uma redução de dimensionalidade usando o T-SNE [2]. Originalmente, o espaço gera *embeddings* com 64 dimensões (concebidos na última camada da rede) e após a aplicação do T-SNE ele é representado por duas dimensões. Para isso, foram considerados os hiperparâmetros *n_components=2*, *learning_rate=1000* e *perplexity=10*, para ilustrar a distribuição dos *embeddings* para amostras de 10 usuários que foram utilizados durante o teste, onde cada cor representa um usuário diferente. Assim, podemos observar que os *embeddings* extraídos da dinâmica de digitação de um mesmo usuário estão próximos, demonstrando que nosso método tem potencial no uso para identificação de alunos no contexto da programação como um método *One-Shot Learning*.[3]

[2] O T-SNE é um método não linear para redução de dimensionalidade adequado para a visualização de conjuntos de dados de alta dimensão.

[3] *One-Shot Learning* é uma tarefa de classificação onde um extrator de *features* já treinado é utilizado novos exemplos.

5.6.6 Experimentos VI

Por fim, neste experimento verificamos se os padrões biométricos dos mesmos usuários mudam no decorrer do tempo à medida em que eles realizam as atividades no ambiente juiz on-line. Desta vez, utilizando o Protocolo IV de separação de dados. A divisão dos dados foi feita de forma manual utilizando a linguagem *Python*, onde cada atividade foi utilizada para treinamento, exceto os segmentos da atividade 5 (última atividade que simula uma avaliação) que sempre foram utilizados para teste. Em cada etapa, foi calculada a acurácia do modelo. No final é calculada a média das acurácias, levando em consideração os valores de cada etapa. O resultado alcançado foi uma acurácia média de 82,9% e taxa de erro médio de 0,038%, conforme mostrado na Tabela 5.6.

Tabela 5.6: Acurácias e Taxas de Erro (%) alcançadas para diferentes quantidades de atividades utilizadas para treinamento

Atividade(s) (Treinamento)	**Atividade (Teste)**	**Acurácia**	**EER**
1	5	67,0	0,096
1 e 2	5	75,1	0,045
1,2 e 3	5	81,7	0,039
1,2,3 e 4	5	82,9	0,038
MÉDIA		**76,6**	**0,054**

Entretanto, como observado nos resultados deste experimento e também no anterior (5.6.5), há indícios que os padrões biométricos dos alunos mudam. À medida em que as atividades selecionadas para treinamento estão mais distantes no aspecto temporal das que separadas para teste, fica evidente que os padrões biométricos do mesmo usuário em atividades diferentes realizadas pelo mesmo usuário, se diferem. Por ser disciplinas iniciais de introdução a programação, o ritmo de digitação do aluno pode ter sofrido variações com o tempo, à medida em que ele realiza as atividades, uma vez que o mesmo está se adaptando a uma dada linguagem de programação. Se esse for o caso, isso causa problemas, pois as previsões se tornam menos precisas com o passar do tempo.

Por conta disso, utilizamos a base de dados I para realizarmos o mesmo experimento. Foram utilizados segmentos de tamanho 30 e janelas deslizantes com 50% de *overlap*, considerando o Protocolo IV. Como mostrado no experimento anterior, o perfil biométrico pode ser obtido quando utilizado o quantitativo de dados de duas ou mais atividades realizadas pelo aluno. Para simularmos a mesma situação, dessa vez utilizando a base de dados I, separamos os segmentos de dados das 2 primeiras sessões da base de dados para compor o conjunto de treinamento, em seguida, os dados da sessão

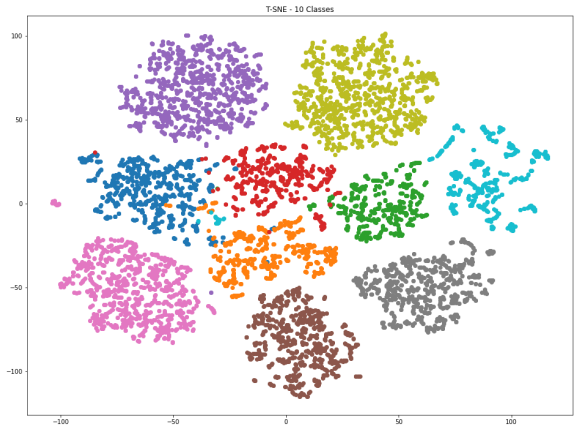

Figura 5.9: T-SNE de 10 usuários (Base de Dados I)

8 (última sessão) para o conjunto de teste. A acurácia alcançada foi de 95,0%, e a taxa de erro de 0,012%. Como observado no resultado deste experimento, a acurácia não sofreu grandes alterações, assim, fica evidente que variações no perfil biométrico dos usuários acontece somente na base de dados II. A Figuras 5.9 mostram a distribuição dos *embeddings* para amostras de 10 usuários da base de dados I que foram utilizados durante os testes. Como observado, a rede extraiu *embeddings* mais próximos para cada usuário, separando melhor seu padrão biométrico comportamental.

5.7 Considerações finais

Neste capítulo apresentamos a avaliação experimental da abordagem proposta. Os resultados apresentados mostram a viabilidade da abordagem para autenticação de alunos em ambientes de juízes on-line, utilizando biometria comportamental e aprendizagem profunda. As análises dos resultados mostram que a incorporação de informação dos caracteres pressionados agrega informação à dinâmica de digitação e melhora o reconhecimento dos usuários do sistema juiz on-line. Além disso, mostramos a eficácia das redes convolucionais siamesa 1D, tendo como entrada caracteres brutos da dinâmica de digitação para autenticação de usuários. Também provamos que a distribuição de amostras das mesmas atividades nos conjunto de treino e teste impactam na acurácia causado possivelmente por um vazamento de informações entre os conjuntos de treino e teste, e por fim, foi elaborada uma metodologia de avaliação que considerou a evolução temporal da aprendizagem dos alunos e constatou-se através desta que os padrões biométricos mudam no decorrer da realização das atividades.

6

Conclusões

Este trabalho demonstrou a viabilidade de uma nova abordagem para autenticar de forma não intrusiva alunos em ambientes juiz on-line. Em ambientes juízes on-line, a autenticação do aluno normalmente é feita apenas no início da sessão de *login* mediante a digitação de uma senha, que apesar de ser o método padrão, apresenta vulnerabilidades, pois, posteriormente, não é verificada nenhuma outra vez a genuinidade do usuário logado e isso pode causar inúmeros problemas, por exemplo, falsificação de identidade.

Nesta pesquisa, utilizamos biometria comportamental, mas especificamente a dinâmica de digitação para tratar esse tipo de problema uma vez que o processo de autenticação utilizando essa técnica não exigir nenhuma ação explícita do usuário, além do baixo custo, pois é necessário somente um teclado convencional para a aquisição de informações, tornando o custo relativamente baixo. Para isso, foi projetada uma arquitetura de CNN 1D Siamesa que aprendeu, a partir dos dados brutos do ritmo da digitação dos usuários, a representação necessária para o reconhecimento dos alunos, dispensando, desta forma, a extração manual de *features*, no qual é um processo que exige muito tempo e a participação de especialistas.

Nos resultados apresentados, identificamos a quantidade de amostras necessárias para a autenticação, dividindo-as em segmentos de tamanhos iguais. Os melhores resultados são alcançados para segmentos fixos de tamanho 30. Provamos que a incorporação de informação dos caracteres pressionados agregam informação à dinâmica de digitação e melhora o reconhecimento. Além disso, nossa rede convolucional siamesa proposta extraiu *features* robustas dos dados brutos da dinâmica da digitação dos alunos, levando a uma precisão superior no reconhecimento dos usuários em comparação ao trabalho *baseline*. Logo, ficou evidente a confiabilidade e a eficácia do método de autenticação proposto também para textos de tamanho fixo e de contexto geral. Além disso, verificamos que a distribuição de amostras das mesmas atividades nos conjunto

de treino e teste impactam na acurácia e, por fim, foi elaborada uma metodologia de avaliação que considerou a evolução temporal da aprendizagem dos alunos e constatou através desta que os padrões biométricos mudam no decorrer da realização das atividades. Assim, este trabalho demonstrou a viabilidade de uma nova abordagem para autenticar de forma não intrusiva alunos em ambientes juiz on-line.

De acordo com a nossa revisão bibliográfica, o presente trabalho é o primeiro método de autenticação de alunos em ambientes juiz on-line. Cabe destacar que os resultados descritos na seção 5 fazem parte do artigo "Autenticação contínua de alunos utilizando biometria comportamental em ambiente Juiz On-line" que foi apresentado no XXXI Simpósio Brasileiro de Informática na Educação (SBIE 2020).

6.1 Impactos do trabalho para o âmbito educacional

Com a utilização do método proposto, pretende-se diminuir as ocorrências reais e prováveis de fraude por parte de estudantes em exercícios e avaliações que ocorrem nos sistemas juiz on-line, uma vez que será possível ser feita a verificação do aluno durante toda a sessão. Monitorando e analisando as entradas de dados de cada usuário com base em seus padrões habituais de digitação, evitando que alunos desonestos obtenham ajuda de terceiros para fazer suas atividades. Além disso, será possível autenticar alunos novatos, ou seja, usuários nunca vistos antes, sem a necessidade de modificação ou atualização do método. Dessa forma, a utilização do método proposto de autenticação de alunos em ambientes de programação baseado em biometria comportamental impacta positivamente o âmbito educacional, uma vez que garante às instituições e professores uma segurança complementar contra fraudes, principalmente em casos em que os alunos têm a opção de fazer os exercícios e avaliações em casa. Além disso, é importante para o aluno, uma vez que testifica sua integridade acadêmica em todas as atividades, recebendo de forma justa as notas. O método também poderá ser utilizado para informar o professor ou monitor da classe, antes mesmo que o usuário impostor seja bloqueado, no qual poderá enviar um alerta ao aluno, o que poderia mudar a percepção dos alunos em relação a fraudes.

6.2 Limitações do método

Vale ressaltar que os dados capturados foram obtidos em ambiente real, dessa forma, foi possível controlarmos o ambiente quando os alunos estavam nos laboratórios da

própria universidade realizando suas atividades no sistema juiz on-line, nesse caso, os teclados dos laboratórios eram padronizados (fator que pode causar efeito na precisão da identificação), além disso, os alunos estavam constantemente acompanhados de monitores e professores. Entretanto, não foi possível o controle do ambiente quando a realização das atividades foi executada em ambiente externo, por exemplo, na casa do aluno. Todavia, a realização das atividades em ambiente externo pode ser um fator positivo, levando em consideração que o mesmo venha a se sentir mais à vontade e confortável, diferente da pressão em uma avaliação, entre outros fatores, que possa alterar os padrões de digitação, que consequentemente venha influenciar os resultados. De certa forma, em caso como este, não sabemos se os alunos realmente fizeram os exercícios sozinhos. No entanto, se este fosse o caso, e se fosse típico, também é provável que nossos resultados fossem piores.

Além disso, percebemos alguns desafios recentes. Por exemplo, trabalhos recentes como Pisani et al. [2019] e Mhenni et al. [2020] identificaram que o ritmo de digitação muda com o transcorrer do tempo em alguns contextos *(Concept Drift)*[1]. Portanto, os dados estáticos podem ficar defasados, diminuindo o desempenho preditivo do sistema. Diante disso, se esse for o caso, por exemplo, havendo mudanças no ritmo de digitação do aluno à medida que ele avança na disciplina de programação, então, surge a necessidade da utilização de técnicas de autenticação biométrica baseadas na dinâmica de digitação dos alunos, que sejam capazes de adaptar dinamicamente os modelos do usuário ao longo do tempo.

6.3 Trabalhos futuros

A arquitetura de Rede Neural proposta poderia ser utilizada como um extrator de *features* genérico para classificar novos alunos, novas turmas, sem que seja necessário atualizá-la (retreino), em uma configuração *One-shot Learning*. Esta hipótese sustenta uma observação para um trabalho futuro, onde um modelo treinado em uma turma seria avaliado em outra turma. Finalmente, também pretendemos avaliar outras arquiteturas de redes neurais, aumentar a quantidade de participantes e por fim avaliar as regras de decisão baseada no nível de confiança da classificação.

[1] O termo *Concept Drift* (desvio de conceito) significa que as propriedades estatísticas da variável de destino, que o modelo está tentando prever, mudam com o tempo de maneiras imprevistas. Isso causa problemas porque as previsões se tornam menos precisas com o passar do tempo

Referências Bibliográficas

Acien, A.; Morales, A.; Monaco, J. V.; Vera-Rodriguez, R. & Fierrez, J. (2021). Typenet: Deep learning keystroke biometrics. *IEEE Transactions on Biometrics, Behavior, and Identity Science.*

Acien, A.; Morales, A.; Vera-Rodriguez, R.; Fierrez, J. & Monaco, J. V. (2020). Typenet: Scaling up keystroke biometrics. Em *2020 IEEE International Joint Conference on Biometrics (IJCB)*, pp. 1--7. IEEE.

Adetunji, T. O.; Zuva, T. & Appiah, M. (2018). A framework of bimodal biometrics for e-assessment authentication systems. Em *2018 International Conference on Intelligent and Innovative Computing Applications (ICONIC)*, pp. 1--5. IEEE.

Andrade, C. H.; Gonçalves, P. H. N.; Bragança, H. L. & Souto, E. (2019). Autenticação contínua de usuários utilizando contadores de desempenho do sistema operacional. *Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais.*

AV, S. K. & Rathi, M. (2019). Keystroke dynamics: A behavioral biometric model for user authentication in online exams. Em *Biometric Authentication in Online Learning Environments*, pp. 183--207. IGI Global.

Bhanu, B. & Kumar, A. (2017). *Deep learning for biometrics.* Springer.

Bradbury, J.; Merity, S.; Xiong, C. & Socher, R. (2016). Quasi-recurrent neural networks. *arXiv preprint arXiv:1611.01576.*

Byun, J.; Park, J. & Oh, A. (2020). Detecting contract cheaters in online programming classes with keystroke dynamics. Em *Proceedings of the Seventh ACM Conference on Learning@ Scale*, pp. 273--276.

Caldarola, R. & MacNeil, T. (2009). Dishonesty deterrence and detection: How technology can ensure distance learning test security and validity. Em *Proceedings of the 8th European Conference on E-Learning*, pp. 108--115.

Çeker, H. & Upadhyaya, S. (2017). Sensitivity analysis in keystroke dynamics using convolutional neural networks. Em *2017 IEEE Workshop on Information Forensics and Security (WIFS)*, pp. 1--6. IEEE.

Centeno, M. P.; Guan, Y. & van Moorsel, A. (2018). Mobile based continuous authentication using deep features. Em *Proceedings of the 2nd International Workshop on Embedded and Mobile Deep Learning*, pp. 19--24.

Chaves, J. O. M. (2014). Uma ferramenta de apoio ao processo de ensino-aprendizagem em disciplinas de programação de computadores por meio da integração dos juízes online ao moodle.

Chong, P.; Elovici, Y. & Binder, A. (2019). User authentication based on mouse dynamics using deep neural networks: A comprehensive study. *IEEE Transactions on Information Forensics and Security.*

Cruz, M. A. & Goldschmidt, R. R. (2018). Algoritmos de aprendizado de má quina aplicados ao reconhecimento de usuário baseado na dinâ mica da digitação: Um estudo comparativo. Em *Anais do XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pp. 295--308. SBC.

Cruz, M. A. & Goldschmidt, R. R. (2019). Deep neural networks applied to user recognition based on keystroke dynamics: Learning from raw data. Em *Proceedings of the XV Brazilian Symposium on Information Systems*, p. 35. ACM.

Cruz, M. A. S.; Duarte, J. C. & Goldschmidt, R. R. (2017). Dinâmica da digitação aplicada à autenticação periódica de usuários em ambientes virtuais de aprendizagem. *Revista Brasileira de Informática na Educação*, 25(02):36.

Dwivedi, C.; Kalra, D.; Naidu, D. & Aggarwal, S. (2018). Keystroke dynamics based biometric authentication: A hybrid classifier approach. Em *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 266--273. IEEE.

Estrela, P. M. A. B. (2020). Autenticação contínua baseada em biometria comportamental para aplicações bancárias mobile.

Feher, C.; Elovici, Y.; Moskovitch, R.; Rokach, L. & Schclar, A. (2012). User identity verification via mouse dynamics. *Information Sciences*, 201:19--36.

Francisco, R.; Júnior, C. P. & Ambrósio, A. P. (2016). Juiz online no ensino de programação introdutória-uma revisao sistemática da literatura. Em *Brazilian Symposium*

on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE), volume 27, p. 11.

Galvão, L.; Fernandes, D. & Gadelha, B. (2016). Juiz online como ferramenta de apoio a uma metodologia de ensino híbrido em programação. Em *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 27, p. 140.

Giot, R. & Rocha, A. (2019). Siamese networks for static keystroke dynamics authentication. Em *IEEE International Workshop on Information Forensics and Security*.

Gomes, A.; Areias, C.; Henriques, J. & Mendes, A. J. (2008). Aprendizagem de programação de computadores: dificuldades e ferramentas de suporte. *Revista portuguesa de pedagogia*, pp. 161--179.

Handa, J.; Singh, S. & Saraswat, S. (2019). A comparative study of mouse and keystroke based authentication. Em *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, pp. 670--674. IEEE.

Hao, Q.; Smith IV, D. H.; Iriumi, N.; Tsikerdekis, M. & Ko, A. J. (2019). A systematic investigation of replications in computing education research. *ACM Transactions on Computing Education (TOCE)*, 19(4):42.

Ichihara, A. & Nizam, O. (2018). The use of business intelligence tools to analyze the influence of interactivity and interaction factors on the assessment of distance students' performance in virtual learning environments. *International Journal of Learning, Teaching and Educational Research*, 17(9).

Jmila, H.; Khedher, M. I.; Blanc, G. & El Yacoubi, M. A. (2019). Siamese network based feature learning for improved intrusion detection. Em *International Conference on Neural Information Processing*, pp. 377--389. Springer.

Killourhy, K. S. & Maxion, R. A. (2009). Comparing anomaly-detection algorithms for keystroke dynamics. Em *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, pp. 125--134. IEEE.

Lavareda Filho, R. M.; Colonna, J. G. & Oliveira, D. B. F. (2020). Autenticação contínua de alunos utilizando biometria comportamental em ambiente juiz on-line. Em *Anais do XXXI Simpósio Brasileiro de Informática na Educação*, pp. 1193--1202. SBC.

Lin, C.-H.; Liu, J.-C. & Lee, K.-Y. (2018). On neural networks for biometric authentication based on keystroke dynamics. *Sensors and Materials*, 30(3):385--396.

Liu, M. & Guan, J. (2017). User keystroke authentication based on convolutional neural network. Em *International Symposium on Mobile Internet Security*, pp. 157--168. Springer.

Longi, K.; Leinonen, J.; Nygren, H.; Salmi, J.; Klami, A. & Vihavainen, A. (2015). Identification of programmers from typing patterns. Em *Proceedings of the 15th Koli Calling Conference on Computing Education Research*, pp. 60--67. ACM.

Lv, K.; Liu, J.; Tang, P. & Li, Q. (2018). Keystroke biometrics for freely typed text based on cnn model. Em *PACIS*, p. 291.

Medeiros, A. P. et al. (2019). Classificação de eventos em monitoramento nilm de cargas elétricas residenciais utilizando rede neural convolucional.

Mhenni, A.; Rosenberger, C. & Amara, N. E. B. (2020). Splitting wolves category in doddington zoo: Impacts on keystroke dynamics. Em *2020 International Conference on Cyberworlds (CW)*, pp. 243--248. IEEE.

Mondal, S. & Bours, P. (2015). A computational approach to the continuous authentication biometric system. *Information Sciences*, 304:28--53.

Mondal, S. & Bours, P. (2017). A study on continuous authentication using a combination of keystroke and mouse biometrics. *Neurocomputing*, 230:1--22.

Muhammad, R. (2018). Optimum feature selection using firefly algorithm for keystroke dynamics. Em *Intelligent Systems Design and Applications: 17th International Conference on Intelligent Systems Design and Applications (ISDA 2017) held in Delhi, India, December 14-16, 2017*, volume 736, p. 399. Springer.

Nunes, D. (2018). Educação superior em computação, estatísticas 2017. *Sociedade Brasileira de Computação*.

Peltola, P.; Kangas, V.; Pirttinen, N.; Nygren, H. & Leinonen, J. (2017). Identification based on typing patterns between programming and free text. Em *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*, pp. 163--167.

Pisani, P. H. & Lorena, A. C. (2013). A systematic review on keystroke dynamics. *Journal of the Brazilian Computer Society*, 19(4):573--587.

Pisani, P. H.; Mhenni, A.; Giot, R.; Cherrier, E.; Poh, N.; Ferreira de Carvalho, A. C. P. d. L.; Rosenberger, C. & Amara, N. E. B. (2019). Adaptive biometric systems: Review and perspectives. *ACM Computing Surveys (CSUR)*, 52(5):1--38.

Quraishi, S. & Bedi, S. (2019). On keystrokes as continuous user biometric authentication. *International Journal of Engineering and Advanced Technology*, 8(6):4149–4153. cited By 0.

Quraishi, S. J. & Bedi, S. (2018). Keystroke dynamics biometrics, a tool for user authentication–review. Em *2018 International Conference on System Modeling & Advancement in Research Trends (SMART)*, pp. 248--254. IEEE.

Santos, J. C. & Ribeiro, A. R. (2012). Jonline: proposta preliminar de um juiz online didático para o ensino de programação. Em *Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)*, volume 1.

Santos, V. A. et al. (2018). Siamesevo-depth: odometria visual através de redes neurais convolucionais siamesas.

Schroff, F.; Kalenichenko, D. & Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815--823.

Silva, R. J. H.; Pazoti, M. A.; da Silva, F. A.; Pereira, D. R. & de Almeida, L. L. (2019). Autenticaçao biométrica para sistemas por meio da dinâmica da digitaçao. Em *Colloquium Exactarum. ISSN: 2178-8332*, volume 11, pp. 26--33.

Silva Filho, S. R. d. L. et al. (2005). Autenticação contínua pela dinâmica da digitação usando máquinas de comitê.

Sundararajan, K. & Woodard, D. L. (2018). Deep learning for biometrics: a survey. *ACM Computing Surveys (CSUR)*, 51(3):65.

Tan, Y. X. M.; Iacovazzi, A.; Homoliak, I.; Elovici, Y. & Binder, A. (2019). Adversarial attacks on remote user authentication using behavioural mouse dynamics. Em *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1--10. IEEE.

Ullah, A.; Xiao, H. & Barker, T. (2019). A dynamic profile questions approach to mitigate impersonation in online examinations. *Journal of Grid Computing*, 17(2):209--223.

Vinayak, R. & Arora, K. (2015). A survey of user authentication using keystroke dynamics. *International Journal of Scientific Research Engineering & Technology (IJSRET)*, 4(4):378--384.

Vorugunti, C. S.; Mukherjee, P.; Pulabaigari, V. et al. (2019). Osvnet: Convolutional siamese network for writer independent online signature verification. Em *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1470--1475. IEEE.

Vural, E.; Huang, J.; Hou, D. & Schuckers, S. (2014). Shared research dataset to support development of keystroke authentication. Em *IEEE International joint conference on biometrics*, pp. 1--8. IEEE.

Xiaofeng, L.; Shengfei, Z. & Shengwei, Y. (2019). Continuous authentication by free-text keystroke based on cnn plus rnn. *Procedia computer science*, 147:314--318.

Yao, Y.; Marcialis, G. L.; Pontil, M.; Frasconi, P. & Roli, F. (2001). A new machine learning approach to fingerprint classification. Em *Congress of the Italian Association for Artificial Intelligence*, pp. 57--63. Springer.

Young, J. R. (2018). Keystroke dynamics: Utilizing keyprint biometrics to identify users in online courses.

Young, J. R.; Davies, R. S.; Jenkins, J. L. & Pfleger, I. (2019). Keystroke dynamics: establishing keyprints to verify users in online courses. *Computers in the Schools*, 36(1):48--68.

Zabihi, M.; Rad, A. B.; Kiranyaz, S.; Särkkä, S. & Gabbouj, M. (2019). 1d convolutional neural network models for sleep arousal detection. *arXiv preprint arXiv:1903.01552*.

Zhigang, S.; Xiaohong, S.; Ning, Z. & Yanyu, C. (2001). Moodle plugins for highly efficient programming courses.

Zhong, Y. & Deng, Y. (2015). A survey on keystroke dynamics biometrics: approaches, advances, and evaluations. *Recent Advances in User Authentication Using Keystroke Dynamics Biometrics*, pp. 1--22.