Urbanno Pereira de Siqueira Leite

# Seleção de características aplicado ao *keystroke dynamics* em dispositivos móveis

RECIFE - PE

JULHO - 2018


**Universidade Federal Rural de Pernambuco**

**Departamento de Estatística e Informática**

**Programa de Pós-Graduação em Informática Aplicada**

# Seleção de características aplicado ao *keystroke dynamics* em dispositivos móveis

Dissertação de mestrado apresentada ao Curso de Pós-Graduação em Informática Aplicada da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Modelagem Estatística e Computacional

Orientador: Prof. Dr. Tiago Alessandro Espínola Ferreira

RECIFE - PE

JULHO - 2018

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema Integrado de Bibliotecas da UFRPE
Biblioteca Central, Recife-PE, Brasil

L533s Leite, Urbanno Pereira de Siqueira
Seleção de características aplicado ao keystroke dynamics em dispositivos móveis / Urbanno Pereira de Siqueira Leite. – 2018.
108 f. : il.

Orientador: Tiago Alessandro Espínola Ferreira.
Dissertação (Mestrado) – Universidade Federal Rural de Pernambuco, Programa de Pós-Graduação em Informática Aplicada, Recife, BR-PE, 2018.
Inclui referências.

1 Keystroke dynamics 2. Seleção de características 3. Dispositivos móveis 4. Classificador 5. Algoritmo genético 6. PSO I. Ferreira, Tiago Alessandro Espínola, orient. II. Título

CDD 004

Urbanno Pereira de Siqueira Leite

# Seleção de características aplicado ao *keystroke dynamics* em dispositivos móveis

Dissertação julgada adequada para obtenção do título de Mestre em Ciência da Computação, defendida e aprovada por unanimidade em 26/02/2018 pela banca examinadora.

**Banca examinadora:**

---

**Prof. Dr. Tiago Alessandro Espínola Ferreira**
Universidade Federal Rural de Pernambuco - UFRPE
Orientador

---

**Prof. Dr. George Darmiton da Cunha Cavalcanti**
Universidade Federal de Pernambuco - UFPE

---

**Prof. Dr. Cícero Garrozi**
Universidade Federal Rural de Pernambuco - UFRPE

---

**Prof. Dr. Péricles Barbosa Cunha de Miranda**
Universidade Federal Rural de Pernambuco - UFRPE

*Dedico este trabalho aos meus pais, que me criaram, e também meu irmão, da melhor maneira possível, garantindo a educação necessária para que eu pudesse estar desenvolvendo este projeto.*

# Agradecimentos

Os agradecimentos direciono a todos aqueles que contribuíram para o meu desenvolvimento tanto pessoal quanto intelectual. Sobretudo a Deus por ter me dado a vida e a capacidade da inteligência.

Aos meus familiares por todo o amor que me é dado, pelas coisas que pude aprender com cada um deles e pela saudade com que sempre sou acolhido.

Ao Prof. Tiago Alessandro Espínola Ferreira sou muito grato por sua orientação. O agradeço por toda a atenção, pelo vasto conhecimento compartilhado e pelos questionamentos que sempre serviram de motivação para a busca pelo desconhecido.

A minha namorada Geizibel pela compreensão, carinho e por nunca me faltar com a motivação. Não podendo esquecer dos cafés que nos serviam de companhia ao longo das madrugadas.

Aos colegas de curso, sobretudo, Renê, Junior e Fabrício, pelas conversas descontraídas carregadas de curiosidades, conhecimentos e muitas risadas. Agradeço a Diego Marinho e Isa Raquel por sempre perguntarem "como está indo o mestrado?" e pelos convites de festinhas que nos últimos tempos foram declinados drasticamente.

Agradeço a todos os professores do Programa de Pós Graduação em Informática Aplicada/UFRPE que contribuíram para a minha evolução acadêmica, levando-me a compreender as coisas novas e questionar aquelas já conhecidas.

# Resumo

Nesse trabalho são desenvolvidos modelos de seleção de características para o *keystroke dynamics* em dispositivos móveis. Dois modelos são elaborados, um baseado em Algoritmo Genético (GA) e outro em PSO. Os seletores propostos são aplicados a uma base de dados pública do *keystroke dynamics* construída a partir de dispositivos móveis. Ambos os métodos de seleção são utilizados em conjunto com variados classificadores, são eles: *Naive* Bayes, Bayes *Net*, C4.5 (árvore de decisão), *Random Forest*, k-NN, *Support Vector Machine* (SVM) e *Multilayer Perceptron* (MLP). Os métodos de seleção de características aqui desenvolvidos são avaliados a partir das métricas de taxa de acuracidade, falso positivo (FAR), falso negativo (FRR) - todas essas medidas são obtidas a partir das classificações - e também por meio das taxas de redução de características. Os resultados obtidos a partir da execução de vários experimentos mostram que os modelos propostos foram capazes de agregar melhorias às medidas de desempenho - quando comparados aos resultados das classificações sem seleção -, além de alcançarem altos níveis de redução de características. Através de uma análise comparativa foi possível também verificar que os modelos desenvolvidos nesse trabalho possuem desempenhos compatíveis com outros seletores já disponíveis na literatura. Os métodos propostos também chamam atenção pela estabilidade do seu comportamento, de tal forma que os resultados por eles gerados possuem baixos índices de variabilidade. Nesse trabalho foi possível ainda se identificar as características mais selecionadas e também aquelas menos escolhidas pelos modelos, sendo mostrado que um atributo pode ser bastante selecionado para um determinado método de classificação, porém não ser tão escolhido para um outro classificador. Já analisando a frequência de seleção da características de acordo com o seu tipo, verificou-se que as duas características mais selecionadas pelos métodos de seleção propostos são atributos inerentes aos dispositivos móveis.

**Palavras-chave**: *Keystroke dynamics*, Seleção de Características, Dispositivos Móveis, Classificador, Algoritmo Genético, PSO.

# Abstract

In this paper, feature selection models are developed for keystroke dynamics on mobile devices. Two models are elaborated, one based on Genetic Algorithm (GA) and another on PSO. The proposed selectors are applied to a public database of keystroke dynamics built from mobile devices. Both methods of selection are used in conjunction with various classifiers: Naive Bayes, Bayes Net, C4.5 (decision tree), Random Forest, k-NN, Support Vector Machine (SVM) and Multilayer Perceptron (MLP). The feature selection methods developed here are evaluated by accuracy, false positive rate (FAR), false negative rate (FRR) - all these measures are obtained from the classifications - and also by the reduction rates of characteristics. The results obtained from the execution of several experiments show that the proposed models were able to add improvements to the measures of performance - when compared to the results of the classifications without selection -, besides reaching high levels of reduction of characteristics. Through a comparative analysis it was also possible to verify that the models developed in this work have performances compatible with other selectors already available in the literature. The proposed methods also call attention for the stability of their behavior, in such a way that the results generated by them have low indices of variability. In this work it was possible to identify the most selected features and also those less chosen by the models, showing that an attribute can be quite selected for a particular classification method, but not so chosen for another classifier. Already analyzing the frequency of selection of characteristics according to their type, it was verified that the two characteristics most selected by the proposed selection methods are attributes inherent to mobile devices.

**Keywords**: Keystroke Dynamics, Feature Selection, Mobile, Classifier, Genetic Algorithm, PSO.

# Lista de ilustrações

# Lista de tabelas

# Lista de abreviaturas, siglas e símbolos

| | |
|---|---|
| CER | *Crossover Error Rate* - Taxa de Erro Cruzado |
| D | *Down* - Ação de pressionar uma tecla |
| $D^n$ | Espaço multidimensional de características com $n$ dimensões |
| DD | *Down-Down* - Intervalo de tempo entre pressionar uma tecla e pressionar outra tecla em seguida |
| DU | *Down-Up* - Intervalo de tempo entre pressionar e soltar uma tecla |
| EER | *Equal Error Rate* - Taxa de Igualdade de Erro |
| ELM | *Extreme Learning Machine* - Máquina de Aprendizagem Extrema |
| $f(\cdot)$ | Função de aptidão (*fitness*) |
| $F_s$ | Quantidade de *features* selecionadas |
| $F_t$ | Quantidade total de *features* |
| FAR | *False Acceptance Rate* - Taxa de Falsa Aceitação (Taxa de Falso Positivo) |
| FMR | *False Match Rate* - Taxa de Falsa Combinação |
| FNMR | *False Non-Match Rate* - Taxa de Falsa Não Combinação |
| FRR | *False Rejection Rate* - Taxa de Falsa Rejeição (Taxa de Falso Negativo) |
| GA | *Genetic ALgorithm* - Algoritmo Genético |
| H | *Hold* ou *Hold time* - Intervalo de tempo entre pressionar e soltar uma tecla |
| HTER | *Half Total Error Rate* - Taxa de Erro Total Médio |
| k-NN | k-*Nearest Neighbors* - k-Vizinhos Mais Próximos |
| LSA | Tecla *shift* esquerda solta depois da letra |
| LSB | Tecla *shift* esquerda solta antes da letra |
| MLP | *Multilayer Perceptron* - Rede neural *perceptron* de múltiplas camadas |
| $n$ | Número de amostras de um conjunto de dados |
| negUD | Característica UD com valor negativo |
| negUU | Característica UU com valor negativo |
| $\omega_j$ | Representação de uma classe dos dados |
| $output_i$ | Saída real da rede para o padrão $i$ |
| P | *Press* - Ação de pressionar uma tecla |
| $p(x)$ | Função de densidade de probabilidade de uma variável aleatória $x$ |

| | |
|---|---|
| $P(\cdot)$ | Função de probabilidade |
| PIN | *Personal Identification Number* - Número de Identificação Pessoal |
| PSO | *Particle Swarm Optimization* - Otimização por Enxame de Partículas |
| QWERTY | Tipo de *layout* de teclado para alfabeto latino |
| R | *Release* - Ação de soltar uma tecla |
| RBR | *Radial Basis Function* - Função de Base Radial |
| RSA | Tecla *shift* direita solta depois da letra |
| RSB | Tecla *shift* direita solta antes da letra |
| ROC | *Receiver Operating Characteristic* - Receptor de Características Operacionais |
| SVM | *Support Vector Machine* - Máquina de Vetor de Suporte |
| $target_i$ | Saída desejada da rede para o padrão $i$ |
| U | *Up* - Ação de soltar uma tecla |
| UD | *Up-Down* - Intervalo de tempo entre soltar uma tecla e pressionar outra tecla em seguida |
| UU | *Up-Up* - Intervalo de tempo entre soltar uma tecla e soltar outra tecla em seguida |
| VTF | *Virtual Key Force* - Força virtal em uma tecla |
| W | Número total de pesos sinápticos |
| WPM | *Words Per Minuto* - Valor médio de palavras digitadas por minuto |

# Sumário

# 1 Introdução

A segurança da informação é um tema cujo o desenvolvimento data antes mesmo dos sistemas computacionais. E se tornou ainda mais explorado com a criação e, posteriormente, popularização dos computadores, tendo posteriormente ganhado ainda maior enfoque após o uso generalizado dos dispositivos móveis, mais precisamente os *smartphones*. Os sistemas - sobretudo aqueles que lidam com dados sigilosos - que não fornecem mecanismos robustos de segurança estão suscetíveis a diversas vulnerabilidades e consequentemente terão uma baixa taxa de aceitação por parte dos usuários. Esta situação favorece o uso de métodos de autenticação cada vez mais bem elaborados, com maiores exigências acerca do seu alto nível de acuracidade, fazendo da autenticação um procedimento fundamental para a segurança da informação e que precisa estar em constante evolução.

A autenticação é o processo de verificação da identidade reivindicada. Esse procedimento pode ser feito através da análise de informações exclusivas do indivíduo. Para isso existem diversos métodos de autenticação que por sua vez podem ser divididos em 3 categorias, são eles: aqueles baseados em conhecimento; baseados em dispositivos (*token*) e; baseados em biometria (SHANMUGAPRIYA; PADMAVATHI, 2009; TEH; TEOH; YUE, 2013).

A autenticação baseada em "conhecimento" caracteriza-se pela necessidade do usuário informar algum dado que é do seu conhecimento, por exemplo, uma senha ou *PIN - Personal Identification Number*. Por conta do baixo custo e facilidade de implementação esse se tornou um dos métodos de autenticação mais conhecidos e utilizados pelos sistemas computacionais. Porém, alguns problemas estão associados a essa categoria, por exemplo, esquecimento de senha, o uso de senhas simples e também a utilização constante de uma mesma senha. Com exceção do primeiro problema descrito, os demais denotam fragilidades que podem ser exploradas por pessoas mal intencionadas. Autenticações desse tipo podem ser violadas através da aplicação de algumas técnicas como: uso de dicionários de senhas comuns, força bruta e uso de engenharia social, causando assim uma deterioração da qualidade da autenticação por conhecimento (KARNAN; AKILA; KRISHNARAJ, 2011; TEH; TEOH; YUE, 2013).

Já nas autenticações por *tokens* os usuários utilizam dispositivos físicos, como cartões magnéticos ou *smart-cards*, para a sua identificação. A princípio mostra-se como sendo uma maneira mais prática e segura de se autenticar, porém, por se tratar de um objeto, o *token* pode ser esquecido ou mesmo perdido pelo usuário, podendo ocasionar problemas de usabilidade ou mesmo de segurança. Com o intuito de se tentar minimizar o uso indevido do *token*, em muitos casos esse método é empregado em conjunto com senhas de acesso. Contudo, nesse caso, na medida em que se aumenta a segurança através da agregação dos métodos de autenticação também cresce o número de possíveis problemas, pois agora o usuário tanto pode estar sujeito a perder o seu *token* como também pode esquecer a sua senha (SHANMUGAPRIYA; PADMAVATHI, 2009; KARNAN; AKILA; KRISHNARAJ, 2011).

Por fim as autenticações baseadas em biometria são aquelas que fazem uso de atributos fisiológicos e ou comportamentais dos indivíduos. São exemplo de atributos físicos biométricos: a íris, retina e a impressão digital. Já dentre os comportamentais pode-se mencionar: a assinatura e a dinâmica de digitação (*keystroke dynamics*). Ao contrário do que acontece nas categorias anteriormente descritas, os atributos biométricos não podem ser esquecidos ou perdidos, sendo a biometria considerada como um dos métodos de autenticação com maior nível de precisão (KARNAN; AKILA; KRISHNARAJ, 2011).

Os níveis de segurança na biometria podem variar de acordo com as características utilizadas. Dentre alguns dos métodos biométricos que podem proporcionar maior nível de acuracidade estão: o de reconhecimento por impressão digital, geometria da mão e escaneamento de íris. Em contrapartida o processo de aquisição dos dados para esses métodos pode ser mais custoso, pois necessitam, em alguns casos, de dispositivos extras - sensores - para a extração das características dos indivíduos, além de serem abordagens mais intrusivas que dependem da interação direta do usuário. O reconhecimento facial e por voz também são métodos que possuem um alto nível de acuracidade, não tão alto quantos os apresentados anteriormente, mas, em compensação, possuem a vantagem de não necessitarem de sensores especializados para a aquisição dos dados, fazendo com que sejam métodos mais acessíveis e não intrusivos (CLARKE; FURNELL, 2005; SHANMUGAPRIYA; GANAPATHI, 2016).

Cada método biométrico possui suas vantagens e desvantagens, podendo elas estarem relacionadas: às dificuldades - ou facilidades - de implementação ou de obtenção das

características dos usuários; aos custos de implantação; ou ainda à forma com que interagem com o usuário. De forma geral as biometrias físicas, também conhecidas como biometrias fisiológicas(SAEVANEE; BHATARAKOSOL, 2008; MOSKOVITCH et al., 2009), possuem um maior nível de precisão, porém são também mais intrusivas quando comparadas às comportamentais (TEH; TEOH; YUE, 2013; SHANMUGAPRIYA; GANAPATHI, 2016).

A dinâmica de digitação (*keystroke dynamics*) é um outro exemplo de biometria comportamental. Esse método consiste na identificação do usuário através do seu ritmo, ou estilo, de digitação. Ao contrário de outras técnicas biométricas, nesse caso os dados dos indivíduos podem ser coletados utilizando-se apenas *software*, sem *hardware* adicional, isso porque todas as características são extraídas a partir dos dispositivos de digitação (teclado numérico, alfanumérico, virtual, etc), podendo inclusive a verificação ocorrer sem o conhecimento do usuário, fazendo do *keystroke dynamics* uma alternativa de autenticação amigável e econômica (ZHONG; DENG, 2015).

Muitos trabalhos foram desenvolvidos com o intuito de se promover o melhoramento da dinâmica de digitação. Melhorias expressivas ao desempenho do *keystroke dynamics* foram percebidas conforme novas técnicas de classificação de dados eram exploradas, sendo este um tema que vem sendo abordado desde as últimas três décadas (GAINES et al., 1980; JOYCE; GUPTA, 1990; MONROSE; RUBIN, 2000).

Essas técnicas, conhecidas como classificadores, permitem que um usuário seja identificado e diferenciado dos demais usuários por meio do reconhecimento de padrões da sua digitação. Para isso os classificadores realizam o processamento dos dados - características extraídas da digitação - e ao final é criado um modelo, ou padrão, que representa o comportamento de cada usuário. Fazendo da classificação de dados um tema de fundamental importância à dinâmica da digitação, pois é através do resultado da classificação que um usuário pode ser identificado como sendo autêntico ou impostor (LEE et al., 2016; SAINI; KAUR; BHATIA, 2016).

Uma outra maneira já conhecida para se maximizar o desempenho do *keystroke dynamics* é através da escolha dos melhores atributos que serão utilizados na classificação, processo conhecido por seleção de características (*feature selection*). Esse processo permite que sejam identificadas as características mais relevantes da digitação, ou seja, aquelas que melhor discriminem um usuário, removendo-se então as características irrelevantes que

tendem a prejudicar a eficiência da classificação (YU; CHO, 2004). Além disso a redução de características gerada pela seleção proporciona uma diminuição no tempo de processamento dos classificadores, uma vez que menos dados serão utilizados, contribuindo desta forma para o aumento da eficiência da classificação (GIOT; EL-ABED; ROSENBERGER, 2011).

Os resultados positivos provenientes dos estudos acerca do *keystroke dynamics*, que mostraram altos níveis de precisão na identificação dos usuários, serviram de motivação para o desenvolvimento de novas formas de aplicação da dinâmica de digitação. Uma das inovações sobre o tema foi a sua aplicação em dispositivos móveis, sendo isso possível graças à popularização e evolução desses aparelhos. Como mostrado em alguns estudos (TROJAHN; ORTMEIER, 2013; GIUFFRIDA et al., 2014; ANTAL; SZABÓ; LÁSZLÓ, 2015), a dinâmica de digitação em dispositivos móveis também gerou resultados relevantes e animadores, com níveis de eficiência iguais ou até mesmo superiores a sua aplicação convencional em computadores.

Na combinação de dispositivos móveis e *keystroke dynamics* um dos pontos mais explorados foi a extração de novas características. Isso porque esse dispositivos, sejam eles *smartphones* ou *tablets*, em sua grande maioria, possuem sensores que disponibilizam medidas acerca da interação do usuário com os aparelhos, podendo fornecer dados como a pressão exercida na tela, área da tela tocada, inclinação do dispositivo, além de outras informações que podem ser utilizadas como novos atributos para o *keystroke dynamics* (FENG et al., 2012; LEE et al., 2016).

## 1.1 Objetivos

Apesar da seleção de características ser um tema já bastante explorado no *keystroke dynamics* convencional, ou seja, em sua aplicação em computadores, e também mesmo conhecendo os benefícios que podem proporcionar à dinâmica da digitação, esses métodos ainda são pouco explorados na dinâmica de digitação em dispositivos móveis.

Assim o presente trabalho tem como objetivo geral construir modelos de seleção de características que possam otimizar o *keystroke dynamics* em dispositivos móveis.

Já os objetivos específicos deste trabalho podem ser enumerados como:

- Desenvolver um método de seleção de características baseado em PSO (*Particle*

*Swarm Optimization*);

- Desenvolver um método de seleção de características baseado em algoritmo genético;
- Avaliar o desempenhos dos métodos de seleção propostos quando combinados a variados classificadores;
- Identificar as características que otimizam o *keystroke dynamics* nos dispositivos móveis;
- Enumerar os atributos da digitação que impactam negativamente o *keystroke dynamics* em dispositivos móveis.

## 1.2 Organização da dissertação

No Capítulo 2, "Keystroke Dynamics", é apresentada uma visão geral sobre alguns aspectos do *keystroke dynamics*. Nele é feito uma breve descrição a respeito dos tipos de autenticação do *keystroke dynamics* que pode ocorrer de forma estática ou contínua. Além disso são demonstradas algumas das métricas que podem ser utilizadas para mensurar o desempenho do *keystroke dynamics*. Nesse capítulo são informadas algumas das bases de dados do *keystroke dynamics* que encontram-se disponíveis publicamente. São descritas também algumas das principais características da digitação que podem ser aplicadas a esse método de autenticação, um tópico crucial para o tema abordado. E por fim é descrito um dos novos desafios da autenticação por dinâmica de digitação que é a sua utilização em dispositivos móveis.

No Capítulo 3, "Classificação", é descrito o papel dos métodos de classificação para o *keystroke dynamics*. Além disso breves explanações são feitas para alguns dos principais classificadores disponíveis na literatura, sendo explicado brevemente o seu funcionamento.

No Capítulo 4, "Seleção de características", apresenta-se o conceito de seleção de características, além de serem apresentadas as vantagens e desvantagens das duas categorias de seleção de características. Também são mostrados alguns dos trabalhos que tratam da seleção de características no *keystroke dynamics*.

No Capítulo 5, "Algortimos de busca utilizados", é apresentado uma breve descrição teórica e prática a respeito do algoritmo genético (GA). Nesse capítulo são ainda

apresentados os operados que descrevem o funcionamento do GA. Também é apresentada uma breve descrição teórica e prática a respeito do algoritmo *Particle Swarm Optimization*, sendo ainda demonstradas algumas das variações da implementação do PSO padrão e também as principais topologias aplicáveis ao referido algoritmo.

No Capítulo 6, "*Metodologia*", são descritos os procedimentos metodológicos adotados no presente trabalho. Nele é explicado o processo de aquisição dos dados, além de serem definidos os parâmetros dos métodos de seleção de características aqui desenvolvidos e também os valores das propriedades dos classificadores.

No Capítulo 7, "Experimentos e resultados", são mostrados alguns dos atributos adotados na execução dos experimentos, além de serem apresentados os resultados obtidos com no trabalho. Neste capítulo são expostos os dados referentes às taxas de acertos e de erros das classificações, e os níveis de redução das características alcançados pelos métodos de seleção aqui implementados. É apresentado ainda uma análise acerca das características que foram selecionadas e também aquelas que não foram escolhidas.

No Capítulo 8, "Conclusão", são apresentadas as conclusões e considerações finais sobre o trabalho. Nele são descritas as contribuições provenientes dos estudos aqui desenvolvidos e ao final são mencionados alguns dos trabalhos a serem futuramente .

# 2 *Keystroke Dynamics*

Em 1985 alguns estudos mostraram indícios de que operadores de telégrafos possuíam padrões de envio de mensagens através de código Morse, ideia esta que foi reforçada pelo fato de que os operadores que recebiam o sinal geralmente conseguiam reconhecer quem estava transmitindo as informações ao ouvir o padrão característico de pontos e traços (MILLER, 1994). Tais estudos serviram de inspiração posteriormente para o desenvolvimento do *keystroke dynamics* (LEGGETT et al., 1991).

Um dos primeiros estudos sobre *Keystroke Dynamics* foi elaborado por Gaines (GAINES et al., 1980) no ano de 1980. Nesse trabalho foram empregados inicialmente 7 usuários que digitaram 3 textos previamente definidos. A partir da digitação foram extraídas as informações referentes ao intervalo de tempo entre cada tecla pressionada, característica esta chamada pelos autores de dígrafo. Todo o processo de digitação ocorreu em duas etapas com um intervalo de 4 meses entre elas, em cada etapa todo o processo de digitação era repetido utilizando-se os mesmos textos, ao final somente 6 digitadores concluíram todas as atividades. Gaines *et. al.* aplicou o teste estatístico $t$ de Student a fim de encontrar diferenças significativas entre os tempos médios de digitação que distinguissem os digitadores. Para isso definiu-se como hipótese nula o fato de os dados analisados serem de uma mesma pessoa, enquanto que a hipótese alternativa descreve que os dados da digitação são de usuários distintos. Ao final os autores apontam evidências para ser possível distinguir digitadores pelo seu perfil de digitação, deixando claro que os resultados obtidos ainda eram preliminares e que novos trabalhos sobre tema seriam necessários para fortalecer a capacidade do *keystroke dynamics*.

A partir daí vários outros estudos foram realizados aperfeiçoando e desenvolvendo múltiplos aspectos do *keystroke dynamics*. Ao longo desse capítulo vários desses aspectos são apresentados.

## 2.1 Autenticação estática e contínua

O *keystroke dynamics* dependendo do seu comportamento operacional pode ser classificado como estático ou dinâmico (SHINDE; SHETTY; MEHRA, 2016).

A abordagem estática analisa as características da digitação apenas em momentos específicos, podendo em um caso prático, por exemplo, ser utilizado durante a realização de *login* em um sistema (MONROSE; RUBIN, 2000). Neste cenário, a autenticação é baseada na entrada de um texto estático (FURNELL et al., 1996).

Essa abordagem apesar de ter sido amplamente utilizada ao longo dos vários primeiros trabalhos sobre o *keystroke dynamics* (GAINES et al., 1980; LEGGETT; WILLIAMS, 1988; JOYCE; GUPTA, 1990), ainda assim é bastante encontrada em publicações mais recentes (BALAGANI et al., 2011; IDRUS et al., 2013; RAVINDRAN; GAUTAM; TIWARI, 2015), pois é uma abordagem relativamente simples e que produz bons resultados (MONROSE; RUBIN, 2000).

Porém, apesar da abordagem estática fornecer uma verificação de usuário mais robusta do que o uso de uma simples senha, ela não proporciona segurança contínua, não conseguindo detectar, por exemplo, um intruso após a autenticação inicial de um usuário autêntico (MONROSE; RUBIN, 2000). Uma solução para esse problema é o uso da análise contínua da digitação. Um dos primeiros estudos sobre a abordagem contínua é apresentada por (LEGGETT et al., 1991), nesse trabalho os autores conseguiram identificar alguns digitadores em menos de 100 teclas digitadas.

Uma característica interessante acerca dos tipos de autenticação do *keystroke dynamics*, estático ou dinâmico, é a forma como se avalia o desempenho do método. Na autenticação estática é importante conhecer a probabilidade de um usuário genuíno não ser reconhecido e também a probabilidade de um usuário impostor ser classificado como genuíno, enquanto que na autenticação contínua é importante saber após quantas teclas digitadas um impostor é identificado (BOURS, 2012).

Em ambas as abordagens é possível avaliar o desempenho do *keystroke dynamics* sob vários aspectos. A seguir são apresentados mais detalhes sobre as avaliações de desempenho do *keystroke dynamics*.

## 2.2 Avaliação de desempenho

Uma das formas de se mensurar o desempenho do *keystroke dynamics* é através da sua efetividade. A efetividade indica a capacidade de se identificar corretamente um indivíduo genuíno e um impostor (TEH; TEOH; YUE, 2013). Nesse caso as duas principais

métricas para avaliar o desempenho de um sistema biométrico como o *keystroke dynamics* são: FAR (*False Acceptance Rate*) e FRR (*False Rejection Rate*) (TEH; TEOH; YUE, 2013; ZHONG; DENG, 2015).

O FAR, algumas vezes conhecido por FMR (*False Match Rate*) (BOURS, 2012), corresponde à probabilidade de um sistema classificar incorretamente um impostor como um usuário genuíno (SHANMUGAPRIYA; PADMAVATHI, 2009), ou seja, indica com que frequência um impostor consegue obter acesso a um sistema (ZHONG; DENG, 2015). Na estatística o FAR é referenciado como erro tipo 2 (SHANMUGAPRIYA; PADMAVATHI, 2009).

O FAR é calculado de acordo com a Equação 2.1, ou seja, é o quociente entre o número de aceitações concedidas incorretamente e a quantidade total de tentativas feitas por impostores, ao final esse valor é multiplicado por 100 obtendo-se assim a taxa de erro em termos percentuais (SAINI; KAUR; BHATIA, 2016).

$$FAR = \frac{\text{Número de aceitações erradas}}{\text{Número total de tentativas do impostor}} * 100 \tag{2.1}$$

Por outro lado, o FRR, também conhecido por FNMR (*False Non-Match Rate*), é a taxa de rejeição de um usuário autêntico (ALI et al., 2017), ou seja, a probabilidade de um sistema classificar incorretamente um usuário genuíno como sendo um impostor (ZHONG; DENG, 2015). É também conhecido como erro tipo 1 (SHANMUGAPRIYA; PADMAVATHI, 2009).

Como demonstrado na Equação 2.2, o valor de FRR é obtido pela razão entre o número de vezes em que um usuário foi rejeitado de forma incorreta e a quantidade total de tentativas realizadas por um usuário genuíno (SAINI; KAUR; BHATIA, 2016), sendo essa uma medida percentual.

$$FRR = \frac{\text{Número de rejeições erradas}}{\text{Número total de tentativas genuínas}} * 100 \tag{2.2}$$

É desejado que tanto os valores de FAR quando de FRR sejam os menores possíveis (ZHONG; DENG, 2015), sendo o valor ideal para ambas as taxas igual a 0% (SHANMUGAPRIYA; PADMAVATHI, 2009). Do ponto de vista de segurança, erros do tipo 2 devem ser minimizados a fim de se evitar que um impostor se passe por um usuário

genuíno, já o erro tipo 1 também deve ser de baixa ocorrência evitando assim que um usuário genuíno tenha o seu acesso rejeitado (OBAIDAT, 1995; SHANMUGAPRIYA; PADMAVATHI, 2009).

Além de FRR e FAR algumas outras métricas também podem ser utilizas para avaliar o desempenho de um sistema biométrico. Uma delas é a *Half Total Error Rate* (HTER). O HTER expressa o desempenho geral do sistema, sendo definido como o valor médio entre as taxas de FAR e FRR (GIOT; EL-ABED; ROSENBERGER, 2011), como demonstrado na Equação 2.3.

$$HTER = \frac{\text{FAR} + \text{FRR}}{2} \tag{2.3}$$

Outra maneira de avaliar o desempenho do *keystroke dynamics* é por meio da taxa de acurácia. Essa taxa mede a porcentagem de acertos na classificação alcançados por um algoritmo ou sistema (PISANI; LORENA, 2013; ZHONG; DENG, 2015).

Outra medida bastante comum nos sistemas biométricos é o *Equal Error Rate* (EER) (SHANMUGAPRIYA; PADMAVATHI, 2009), que pode também ser referenciado como *Crossover Error Rate* (CER) (ALI et al., 2017). O EER corresponde ao limiar em que as taxas de FAR e FRR são iguais (KILLOURHY; MAXION, 2009), como pode ser visto na Figura 1.

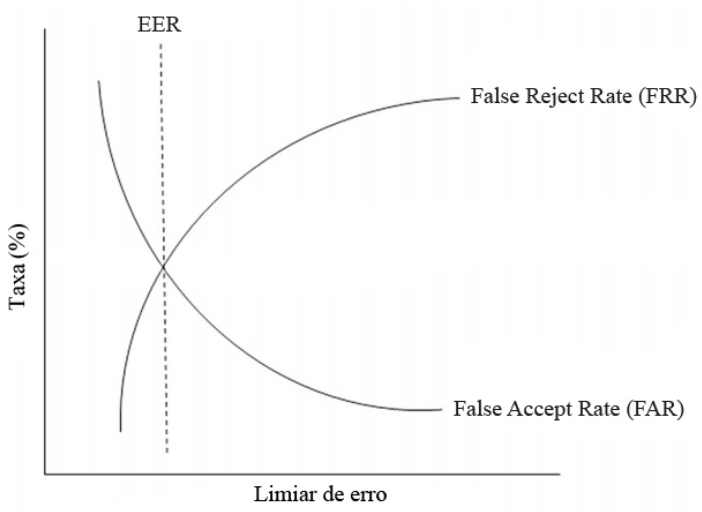


Figura 1 – Relação entre FAR, FRR e EER. Adaptado de (ALI et al., 2017)

Esta medida pode ser obtida a partir da curva ROC (*Receiver Operating Characteristic*), em um ponto específico da curva onde o FAR é igual a FRR. Ali estará então

definido o EER (ZHONG; DENG, 2015).

Conforme ilustrado na Figura 2, para se obter o EER a partir da curva ROC deve-se inicialmente construir a curva ROC plotando-se cada ponto FAR e FRR em uma escala logarítmica, depois é necessário traçar uma linha com ângulo de 45° a partir da origem (0,0) e no ponto em que esta linha cruzar a curva ROC este será o EER (REID, 2004).

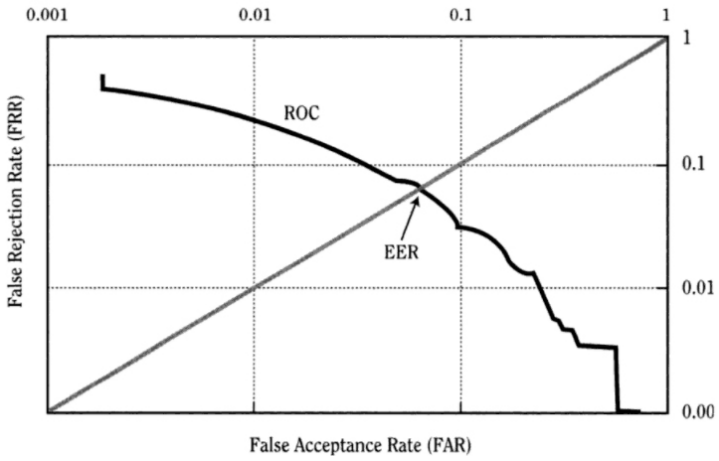


Figura 2 – Calculando o EER a partir da curva ROC (REID, 2004)

## 2.3 Base de dados

No *keystroke dynamics* a verificação de um usuário se dá pela comparação entre seus dados de referência e aqueles obtidos pontualmente durante a sua entrada em um sistema, assim caso os recursos extraídos forem similares aos de referência o usuário é autenticado, do contrário o usuário tem o seu acesso negado (ALSULTAN; WARWICK; WEI, 2017). Por isso, inicialmente múltiplas amostras de digitação devem ser registradas por um usuário para que sirvam como um referencial biométrico (SHANMUGAPRIYA; PADMAVATHI, 2009; GIOT; DORIZZI; ROSENBERGER, 2015), fazendo com que a atividade de aquisição de dados seja o primeiro passo no processo de criação dos sistemas de *keystroke dynamics* (TEH; TEOH; YUE, 2013; ALI et al., 2017)

No entanto, a criação dessas bases de dados, sobretudo quando se utilizado um grande número de indivíduos, pode demandar bastante tempo e esforço (CHERIFI et al., 2009; GIOT; EL-ABED; ROSENBERGER, 2009; GIOT; EL-ABED; ROSENBERGER, 2011). Por isso, com o intuito de se estimular o desenvolvimento de trabalhos sobre o

tema, muitas bases de dados (*benchmark*) vem sendo disponibilizadas publicamente na literatura (GIOT; EL-ABED; ROSENBERGER, 2011), possibilitando que pesquisadores possam trabalhar com os mesmos conjunto de dados, o que lhes permite comparar seus algoritmos com outros já desenvolvidos em um mesmo contexto (GIOT; EL-ABED; ROSENBERGER, 2009), além de lhes poupar a atividade de extração dos dados.

A seguir são apresentados alguns dos *benchmarks* do *keystroke dynamics* que estão disponibilizados publicamente na literatura.

Killourhy *et. at.* (2009)[1] apresenta uma base de dados que possui 20400 amostras. Para a criação dos dados foram recrutados 51 digitadores que participaram de 8 sessões de digitação. Em cada sessão os usuários digitavam 50 vezes a sequência de caracteres ".tie5Roanl" (sem aspas), totalizando 400 amostras por usuário. Dos digitadores 30 eram do sexo masculino e 21 feminino, eram 43 destros e 8 canhotos. Em cada repetição foram extraídas 31 características, sendo elas do tipo *Hold*, *Down-Down* e *Up-Down*. Mais informações sobre características (*features*) serão apresentadas na Seção 2.4.

Uma outra base de dados é proposta pelo grupo BioChaves, um grupo de pesquisa que desenvolve trabalhos sobre biometria[2]. A equipe BioChaves apresenta através de alguns trabalhos (FILHO; FREIRE, 2006; MONTALVAO; ALMEIDA; FREIRE, 2006) a descrição dos dados.

O *benchmark* desenvolvido pela equipe BioChaves é composto por 4 bases de dados (A, B, C e D), sendo que todas elas possuem apenas características do tipo *Down-Down*. Os dados das bases A e B foram extraídos a partir da digitação das palavras: chocolate, zebra, banana e taxi. Nessas bases cada usuário repetiu a digitação 10 vezes, 5 vezes na primeira sessão e mais 5 vezes na segunda sessão, existindo o intervalo de um mês entre cada sessão. Já o texto utilizado na base C foi "computador calcula" (sem aspas) e nesse caso os usuários estavam completamente livres para digitar a sequência onde e quanto quisessem. Por fim, a base de dados D foi construída a partir da extração de características da digitação de um texto livre, para isso cada usuário digitou um conjunto de 10 linhas de textos, cada linha contendo cerca de 110 "batidas" (*stroke*). No total foram utilizados 47 digitadores, 10 na base A, 8 na base B, 14 na C e 15 na base de dados D.

Outro importante *benchmark* é proposto por Giot *at. al.* (2009). Participaram do

[1] Disponível em <http://www.cs.cmu.edu/~keystroke>
[2] Disponível em <http://www.biochaves.com/en/download.htm>

processo de captura de dados 133 usuários, cada um digitando entre 5 a 107 vezes a senha "greyc laboratory". Em média cada usuário realizou 51 repetições, sendo que 100 desses usuários registraram mais de 60 amostras, obtendo ao final 7555 amostras. A maioria dos indivíduos participou de pelo menos 5 sessões com intervalos de 1 semana entre elas. Essa base de dados encontra-se disponível publicamente[3] e contém características do tipo *Down-Down*, *Down-Up*, *Up-Down* e *Up-Up*.

Muitos trabalhos foram feitos sobre sistemas de *keystroke dynamics* implantados em PCs ou *laptops*, porém nos últimos anos com o aumento do uso de telefones celulares (*smartphones*) alguns pesquisadores redirecionaram o foco da análise de dinâmica de digitação para dispositivos móveis (SAINI; KAUR; BHATIA, 2016), motivando a criação de *benchmarks* a partir desses aparelhos(*smartphone*, *tablet*, etc) (EL-ABED; DAFER; KHAYAT, 2014; ANTAL; SZABÓ; LÁSZLÓ, 2015).

Um *benchmark* desse tipo foi elaborado por (ANTAL; SZABÓ; LÁSZLÓ, 2015). Para a coleta de dados dessa base foram utilizados 42 duas pessoas, sendo 24 homens e 18 mulheres, com idade entre 20 e 46 anos. A obtenção das amostras ocorreu ao longo de várias sessões onde em cada sessão o usuário realizava 30 repetições do texto ".tie5Roanl" (sem aspas). As digitações que continham a ação de deleção - uso da tecla *delete* ou *backspace* - foram excluídas do conjunto de amostras, gerando assim uma base de dados com 51 padrões por usuário. Para a coleta dos dados foram utilizados 37 *tablets* e 5 *smartphones*. A partir da digitação nos dispositivos foi possível se obter 71 características, sendo elas do tipo: *Down-Down*, *Up-Down*, *Hold time*, *Hold pressure* e *Finger area*, além disso são consideradas ainda como características os valores médios de *Hold time*, *Hold pressure* e *Finger area*. Esta base de dados também encontra-se disponível publicamente[4].

Além dos *benchmarks* aqui apresentados, outros mais podem ser encontrados na literatura (LI et al., 2011; IDRUS et al., 2013; EL-ABED; DAFER; KHAYAT, 2014; VURAL et al., 2014)[5,6,7,8].

[3] Disponível em <http://www.epaymentbiometrics.ensicaen.fr/greyc-keystroke-dataset>
[4] Disponível em <https://www.ms.sapientia.ro/~manyi/keystroke.html>
[5] Disponível em <http://mpl.buaa.edu.cn/detail1.htm>
[6] Disponível em <http://www.epaymentbiometrics.ensicaen.fr/greyc-nislab-keystroke-dynamics-dataset>
[7] Disponível em <http://www.coolestech.com/rhu-keystroke/>
[8] Disponível em <http://internal.clarkson.edu/citer/research/collections/keystrokes.html>

## 2.4 Características do *keystroke dynamics*

O *Keystroke Dynamics* é o processo de medição e avaliação do ritmo de digitação de um usuário (ZHOU et al., 2016), por isso é comum se utilizar nesse método características (*features*) que representem o intervalo de tempo entre ações de digitação (ALSULTAN; WARWICK; WEI, 2017).

As características baseadas em tempo são extraídas a partir de dois eventos básicos da digitação, o pressionar de uma tecla (*down* - D, *press* - P) e o soltar de uma tecla (*up* - U, *release* - R) (LEE et al., 2016). Ao se combinar esses eventos vários tipos de *features* pode ser gerados, como veremos a seguir.

*Down-Up* (DU): O intervalo de tempo entre pressionar e soltar uma tecla. Representa o período em que uma determinada tecla foi mantida pressionada. É também conhecido como *hold time* (H) ou *dwell time* (TEH; TEOH; YUE, 2013; PISANI; LORENA, 2013; SHINDE; SHETTY; MEHRA, 2016). Característica expressa na Equação 2.4, sendo $n$ o $n$-ésimo elemento de uma sequência de caracteres.

$$DU = R_n - P_n \tag{2.4}$$

*Down-Down* (DD): Diferença de tempo entre pressionar uma tecla e pressionar a tecla seguinte (PISANI; LORENA, 2013; SHINDE; SHETTY; MEHRA, 2016; LEE et al., 2016). Característica formalizada de acordo com a Equação 2.5.

$$DD = P_{n+1} - P_n \tag{2.5}$$

*Up-Down* (UD): Intervalo de tempo entre soltar uma tecla e pressionar a tecla seguinte, conforme descrito na Equação 2.6. É também conhecido como *flight time* (TEH; TEOH; YUE, 2013; SHINDE; SHETTY; MEHRA, 2016). É possível que *features* do tipo UD apresentem valores negativos e isso ocorre quando pressiona-se uma tecla antes mesmo de soltar a tecla anterior (ALSULTAN; WARWICK; WEI, 2017).

$$UD = P_{n+1} - R_n \tag{2.6}$$

*Up-Up* (UU): Diferença de tempo entre soltar uma tecla e soltar a tecla se-

guinte (MOSKOVITCH et al., 2009; PISANI; LORENA, 2013). Essa característica é obtida seguindo a Equação 2.7.

$$UU = R_{n+1} - R_n \tag{2.7}$$

As características apresentadas anteriormente são provenientes da observação de 2 ações, porém outras *features* podem ser obtidas a partir da observação de 3 eventos sucessivos, conhecidas como trígrafos (TEH; TEOH; YUE, 2013), ou ainda por $n$ eventos básicos de digitação (MOSKOVITCH et al., 2009).

Existem também outras *features* que são menos comuns, ou ainda, que são poucas exploradas, chamadas de *features* não convencionais. Elas podem proporcionar melhorias na identificação de um digitador, pois agregam mais informações que discriminem o comportamento do digitador (ALSULTAN; WARWICK; WEI, 2017). São exemplos de características não convencionais as *features* baseadas na velocidade da digitação, sendo uma das mais conhecidas o tempo médio de palavras digitadas por minuto (*average words-per-minuto - WPM*) (HEMPSTALK; FRANK; WITTEN, 2008). Para se obter o WPM é necessário se calcular o tempo total da digitação, ou seja, o intervalo entre a primeira tecla pressionada e o instante em que a última tecla é liberada (*release*), então realiza-se o quociente entre o número total de palavras digitadas pelo tempo total da digitação (ALSULTAN; WARWICK; WEI, 2017), conforme mostrado na Equação 2.8.

$$WPM = \frac{\text{Número de palavras}}{\text{Tempo total de digitação em minutos}} \tag{2.8}$$

Outras características não convencionais podem ser obtidas a partir das *features* UD e UU com valores negativos. Uma delas é a taxa de UD negativo (negUD), que é computada pelo quociente entre a quantidade de UDs negativos e o número total de pares de teclas, ou seja, quando duas teclas são pressionadas sem que entre elas não tenha ocorrido a liberação de uma tecla (ALSULTAN; WARWICK; WEI, 2017), conforme demonstrado na Equação 2.9.

$$negUD = \frac{\text{Número de UDs negativos}}{\text{Número total de pares de teclas}} \tag{2.9}$$

Já a taxa de UU negativo (negUU) é obtida através da divisão entre o numero de

ocorrência de características UU negativa pelo número total de pares de teclas (ALSULTAN; WARWICK; WEI, 2017), cálculo demonstrado na Equação 2.10.

$$negUU = \frac{\text{Número de UUs negativos}}{\text{Número total de pares de teclas}} \tag{2.10}$$

É possível ainda se obter características da digitação que são atemporais (ALSULTAN; WARWICK; WEI, 2017). Um desses comportamentos é a forma como um usuário incorpora uma letra maiúscula à digitação. Alsultan *et. al.* 2017 apresenta a taxa de uso da tecla *CapsLock* como uma características do *keystroke dynamics*, para isso pega-se o número de vezes em que a tecla *CapsLock* foi pressionada e o divide pela quantidade total de teclas digitadas, conforme segue a Equação 2.11

$$CapsLock = \frac{\text{Número de } CapsLocks}{\text{Número total de teclas}} \tag{2.11}$$

Outro tipo de *feature* descrita por Alsultan *et. al.* (2017) aborda o uso da tecla *Shift*, podendo ser analisado se a tecla *Shift* pressionada está posicionada do lado esquerdo ou direito do teclado, além de observar se a tecla *Shift* foi solta antes ou depois da segunda tecla pressionada.

Ao final, através do uso da tecla *Shift* é possível obter 4 características, conforme descrito na Equação 2.12. Se $x$ for igual à tecla *Shift* direita liberada depois da letra, então $S = RSA$; Senão se x = *Shift* direito liberado antes da letra, então $S = RSB$; Caso contrário, se x = *Shift* esquerdo for solto depois da letra, então $S = LSA$; Por fim, caso x = *Shift* esquerdo solto antes da letra, então $S = LSB$.

$$S = \frac{\text{Número de x}}{\text{Número total de } Shifts} \tag{2.12}$$

Bartlow e Cukic (2006) também exploraram outras características provenientes da tecla *Shift*, dentre as: o tempo médio em que a tecla *Shift* permaneceu pressionada e também o seu tempo máximo e mínimo.

Além dessas características não convencionais obtidas através de teclas especiais, várias outras podem ser empregadas ao *keystroke dynamics*, por exemplo: teclas direcionais (*arrows*), *page up*, *page down*, *home*, *end*, *delete*, *insert*, combinações de teclas para copiar e colar (VILLANI et al., 2006; GONZALEZ; CALOT, 2015).

Alguns estudos propõem ainda uma abordagem do *keystroke dynamics* a partir da análise acústica da digitação (NGUYEN; LE; LE, 2010; ROTH et al., 2013; ROTH et al., 2015). Nesse caso os sons ou ruídos gerados na digitação também são utilizados como *features* do *keystroke dynamics*.

## 2.5 *Keystroke dynamics* em dispositivos móveis

A popularização dos *smartphones* fez com que cada vez mais dados confidenciais fossem armazenados nesses dispositivos, despertando a preocupação em se desenvolver métodos de segurança que pudessem agregar maior confiabilidade a esses aparelhos. O constante desenvolvimento de novas tecnologias para esses dispositivos móveis permitiu que vários métodos de autenticação fossem propostos, sendo vários deles do tipo biométricos, fazendo com que esses aparelhos possuíssem até mesmo mais mecanismos de autenticação do que um computador pessoal (LEE et al., 2016; ALZUBAIDI; KALITA, 2016).

Dentre os métodos de autenticação biométricos já propostos para os dispositivos móveis estão: reconhecimento facial, reconhecimento de digital, escaneamento de íris, reconhecimento de voz, análise de digitação (*keystroke dynamics*).

Porém nem todos os *smartphones* oferecem suporte a esses métodos, pois em muitos casos é necessário o uso de sensores especializados que podem não estar disponíveis nesses dispositivos. Assim, dentre as soluções em segurança atraentes para os dispositivos móveis está o *keystroke dynamics*, pois além de ser uma solução de baixo custo (não requisita recursos extras), apresenta também um bom nível de eficiência (LEE et al., 2016).

Um dos primeiros estudos sobre o *keystroke dynamics* aplicados aos dispositivos móveis foi desenvolvido por (CLARKE; FURNELL, 2006). Nesse trabalho os aparelhos utilizados possuíam apenas teclados físicos (*layout* de 12 dígitos) e através deles foram extraídas as características do tipo *up-down* e *hold-time*. Porém com a rápida evolução da tecnologia, os dispositivos móveis também ganharam maior poder de processamento, permitindo agregar a esses dispositivos vários outros recursos, como por exemplo, telas sensíveis ao toque (*touch screen*) (TEH; TEOH; YUE, 2013).

A inspiração para utilizar o *keystroke dynamics* nos *smartphones* antecede a época em que esses aparelhos se tornaram populares. Em (SAEVANEE; BHATARAKOSOL, 2008) é feita uma investigação do comportamento dos usuários ao digitar o número de

telefone em um dispositivo móvel, porém, considerando que na época esses dispositivos ainda eram escassos, o autor utilizou o *touchpad* de um notebook como forma a simular o toque da tela do *smartphone*, sendo desta forma obtidas as características de pressão do toque, o tempo em que uma tecla é mantida pressionada (*hold*) e o tempo entre soltar uma tecla e pressionar outra (*up-down*). Ao final o autor consegue uma taxa de 99% de acurácia ao utilizar apenas as características de pressão do toque.

Tais resultados serviram de motivação para o desenvolvimento de novos trabalhos permitindo que diversos aspectos do *keystroke dynamics* nos dispositivos móveis fossem explorados. Como é o caso do estudo de (TROJAHN; ORTMEIER, 2012) que mostra os diferentes desempenhos do *keystroke dynamics* quando aplicado em dispositivos com *layout* de 12 dígitos (teclado físico) e quanto utilizado em *layout* QWERTY (teclado virtual). Ou ainda no trabalho de (ROH; LEE; KIM, 2016) onde a precisão do *keystroke dynamics* é avaliado de acordo com a postura em que o usuário manuseia o celular, sendo as características da digitação extraídas em 3 situações: com os usuários caminhando, utilizando o telefone sobre a mesa e também segurando o telefone confortavelmente. Além de vários outros trabalhos, como os de (EL-ABED; DAFER; KHAYAT, 2014) e (ANTAL; SZABÓ; LÁSZLÓ, 2015) que criaram e disponibilizaram publicamente suas bases de dados (*benchmarks*), como forma a viabilizar novas pesquisas sobre o assunto.

Características diferenciadas podem ser obtidas, por exemplo, a partir do sensor de toque (tela) dos *smartphones* (ALZUBAIDI; KALITA, 2016; ZHOU et al., 2016). De acordo com Feng *et. al.* (2012) os dados do toque na tela podem ser classificados em 3 categorias: gestos de toque (por exemplo, gestos de deslizar, pinça, etc); digitação virtual (uso de um teclado virtual sensível ao toque); e desenho baseado em toque (quando através do toque se desenha formas). Além disso é possível ainda se obter a pressão do toque na tela, a área do toque e as coordenadas (X e Y) do toque na tela do *smartphone* (ANTAL; SZABÓ; LÁSZLÓ, 2015; ROH; LEE; KIM, 2016; ZHOU et al., 2016);

Outras *features* podem também ser extraídas a partir dos sensores de giroscópio e acelerômetro (TROJAHN; ORTMEIER, 2013; GIUFFRIDA et al., 2014; LEE et al., 2016; ROH; LEE; KIM, 2016). O acelerômetro mensura a aceleração do dispositivo nos eixos X, Y e Z, enquanto que o giroscópio calcula a orientação do dispositivo em relação à Terra (GIUFFRIDA et al., 2014), sendo que ambos obtem as informações da digitação a partir da movimentação dos dispositivos móveis (LEE et al., 2016).

Com esses variados tipos de características é possível realizar combinações entre eles e aplicá-los ao *keystroke dynamics*. No trabalho de (GIUFFRIDA et al., 2014), por exemplo, os autores utilizaram um vector de características hibrido, que além de conter *features* baseadas no tempo da digitação possui também dados extraídos a partir dos sensores acelerômetro e de orientação (giroscópio). Os seus resultados mostraram que a precisão produzida pelos recursos baseados em sensores supera a eficiência das *features* padrão (baseadas em intervalos de tempo), de tal forma que enquanto que os sensores obtiveram taxa de 0,08% EER, as demais características alcançaram o valor de 4,97% EER.

Um outro trabalho, agora proposto por (FENG et al., 2012), também fez uso de variados tipos de características. Além das características convencionais (baseadas em intervalos de tempo), foram extraídas também os gestos do deslizar dos dedos na tela, além dos movimentos de zoom (pinça). Esse trabalho ainda se diferencia pelo fato de ter sido adotada uma abordagem de verificação contínua.

# 3 Classificação

Após a etapa de extração das características é aplicado então os processos de classificação dos dados (TROJAHN; ORTMEIER, 2012).

Através das técnicas de classificação - também conhecidas como detectores de anomalias - é possível se criar modelos para cada usuário, ou classe, que representem o seu padrão de digitação, sendo esses modelos utilizados posteriormente na validação do digitador (KILLOURHY; MAXION, 2009; SHINDE; SHETTY; MEHRA, 2016; SAINI; KAUR; BHATIA, 2016).

A precisão ou acuracidade de um classificador é baseado na taxa de acertos dos testes de predição (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997). Para essa mensuração inicialmente é necessário dividir o conjunto total de dados em dois subconjuntos, um para aprendizado e outro para testes(MONROSE; RUBIN, 2000).

Os dados de aprendizado são utilizados no treinamento do classificador e ao final desse processo é gerado um *template*, que no problema do *keystroke dynamics* representa um modelo do comportamento da digitação de cada usuário. Enquanto que o conjunto de teste é utilizado para verificar o nível de acuracidade do classificador(KILLOURHY; MAXION, 2009; SHINDE; SHETTY; MEHRA, 2016).

Os métodos de classificação podem variar de acordo com a sua abordagem, podendo ser classificados como: técnicas estatísticas, redes neurais, reconhecimento de padrões, abordagens híbridas, dentre outras(KARNAN; AKILA; KRISHNARAJ, 2011; SHINDE; SHETTY; MEHRA, 2016; SAINI; KAUR; BHATIA, 2016). A seguir algumas dessas técnicas de classificação são apresentadas.

## 3.1 Distância Euclidiana

A distância euclidiana (*Euclidean Distance*) é um dos mais simples algoritmos para classificação (SHINDE; SHETTY; MEHRA, 2016), apesar disso é uma técnica que quando aplicado ao *keystroke dynamics* pode produzir resultados expressivos, como demonstrado por (KILLOURHY; MAXION, 2009) e Monrose (MONROSE; RUBIN, 2000).

Para o cálculo euclidiano considera-se que cada amostra de digitação, ou seja, uma instância dos dados, representa um ponto em um espaço $d$-dimensional conhecido como espaço Euclidiano (*Euclidean space*), sendo $d$ o número de elementos do vetor de características (DUDA; HART; STORK, 2012).

Para execução da classificação, que no *keystroke dynamics* representa a identificação do usuário, os dados do conjunto de treinamento são tratados como uma nuvem de pontos, então calcula-se a proximidade entre os pontos (dados) de teste e o centro dessa nuvem (KILLOURHY; MAXION, 2009).

Na Equação 3.1 é apresentado formalmente o cálculo da distância euclidiana, onde R e U representam na classificação os conjuntos de dados de treinamento e teste respectivamente, enquanto que $r$ e $u$ correspondem ao $i$-ésimo elemento do vetor de característica dos dados (MONROSE; RUBIN, 2000). É necessário salientar que para se obter a distância entre cada *feature* $r$ e $u$, deve-se considerar que $r_i$ é o valor médio da *feature* $r$ de cada amostrada (DUDA; HART; STORK, 2012).

$$D(R,U) = \left[\sum_{i=1}^{N}(r_i - u_i)^2\right]^{1/2} \tag{3.1}$$

## 3.2 Distância Manhattan

Uma outra técnica estatística de classificação é a distância Manhattan, um classificador similar à distância euclidiana, também conhecido como *city block* (ZHONG; DENG; JAIN, 2012; SHINDE; SHETTY; MEHRA, 2016). O cálculo da distância Manhattan é definida pela Equação 3.2.

$$D(a,b) = \sum_{i=1}^{d}|a_i - b_i| \tag{3.2}$$

Tanto a métrica de manhattan quanto a euclidiana são especializações da distância Minkowski (DUDA; HART; STORK, 2012). Na Equação 3.3, que formaliza a métrica Minkowski, a variável $k$ representa um grau de liberdade que na distância euclidiana é igual a 2 e na distância manhattan é 1, levando ambas as medidas serem conhecidas respectivamente como $L_2$ e $L_1$ (DUDA; HART; STORK, 2012; WANG; NESKOVIC;

COOPER, 2007).

$$D(a,b) = \left( \sum_{i-1}^{d} |a_i - b_i|^k \right)^{1/k} \tag{3.3}$$

A distância manhattan possui maior robustez na influência de *outliers*, quando comparada às métricas de distância Euclidiana ou Mahalabonis (ZHONG; DENG; JAIN, 2012), estimulando assim o uso desta técnica de classificação em estudos com o *keystroke dynamics* (RYBNIK; TABEDZKI; SAEED, 2008; KILLOURHY; MAXION, 2009).

## 3.3 *Naive* Bayes

Uma outra abordagem também explorada no *keystroke dynamics* são os classificadores Bayesianos (LEBERKNIGHT; WIDMEYER; RECCE, 2008; BALAGANI et al., 2011; ANTAL; SZABÓ; LÁSZLÓ, 2015). Eles consistem em métodos probabilísticos baseados na teoria de Bayes, que por sua vez corresponde à probabilidade de ocorrência de um evento dado que algum outro evento ocorreu (LEBERKNIGHT; WIDMEYER; RECCE, 2008). Para isso esses classificadores apoiam-se no pressuposto de que a probabilidade a priori e as distribuições dos padrões são conhecidas, sendo necessário utilizar as probabilidades a posteriori para atribuir uma classe a um determinado padrão (MURTY; DEVI, 2011; DUDA; HART; STORK, 2012).

A Equação 3.4 apresentada a fórmula de Bayes, através dela é possível se obter a probabilidade a posteriori.

$$P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)} \tag{3.4}$$

onde $\omega_j$ representa as classes, ou seja, a variável que se deseja descrever probabilisticamente, enquanto que $x$ pode ser considera como sendo uma variável aleatória cuja distribuição depende do estado da natureza, ou seja, da classe. Já $p(x|\omega_j)$ é a função de densidade de probabilidade condicional da classe, sendo também conhecida como a probabilidade de $\omega_j$ em relação a $x$. $P(\omega_j)$ corresponde à probabilidade a priori, enquanto que $p(x)$ é um fator de evidência, que na prática pode ser visto como um fator de escala que garante que as probabilidades a posteriori somem o valor 1 (DUDA; HART; STORK, 2012).

A partir dessa abordagem é possível se ter alguns tipos de classificadores.

O naive bayes, também conhecido como *idiot Bayes rule*, é um classificador baseado na teoria baysiana e parte do pressuposto de que considerando o valor de uma classe $\omega$ todas as *features* $x$ são condicionalmente independentes (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997; DUDA; HART; STORK, 2012).

No classificador *naive* bayes, conforme descrito na Equação 3.5, a probabilidade condicional da classe $p(\omega_k|\mathbf{x})$ é proporcional ao produtório da probabilidade a priori das $d$ características do vetor $\mathbf{x}$. Mais detalhes sobre a formalização do método pode ser visto em (MURTY; DEVI, 2011).

$$p(\omega_k|\mathbf{x}) \propto \prod_{i=1}^{d} p(x_i|\omega_k) \tag{3.5}$$

Apesar de receber o nome de ingênuo (*naive*), já que a hipótese de independência é assumida arbitrariamente sem análise dos dados, esse classificador apresenta desempenho competitivo com outras técnicas de classificação (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997) se mostrando uma solução candidata ao problema *keystroke dynamics* (BALAGANI et al., 2011; ANTAL; SZABÓ; LÁSZLÓ, 2015; HO; KANG, 2017).

## 3.4 Redes Bayesianas

Em algumas situações é possível identificar de forma segura quais variáveis são dependentes ou não, mesmo sem dados amostrais (DUDA; HART; STORK, 2012). Em alguns cenários a hipótese de independência dos atributos pode ser considerada irreal (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997), porém em muitos casos a dependência dos atributos é evidente.

Uma maneira de representar essas dependências em um classificador é por meio das redes baysianas, também conhecidas como *bayesian belief networks* (DUDA; HART; STORK, 2012).

As redes bayesianas são grafos acíclicos direcionados (DAG - *Directed Acyclic Graph*) cujos nós representam as variáveis e suas arestas codificam as dependências condicionais entra as variáveis (MURTY; DEVI, 2011).

Analisando a Figura 3, considera-se que os nodos $A, B, C, D, X$ do grafo representam os atributos, enquanto que $a, b, c, d, x$ correspondem aos possíveis valores ou estados dos

nós, considerando para tanto uma representação com valores discretos. De tal forma que $P(x|a)$, indica a probabilidade condicional de $X$ estar no estado $x$ considerando que o valor de $A$ corresponde a $a$ (DUDA; HART; STORK, 2012).

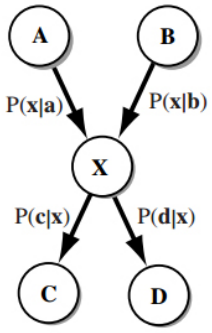


Figura 3 – Exemplo simplificado de uma rede bayesiana (DUDA; HART; STORK, 2012)

Este é também um classificador comumente empregado ao *keystroke dynamics* (BALAGANI et al., 2011; TROJAHN; ORTMEIER, 2013; ANTAL; SZABÓ; LÁSZLÓ, 2015).

## 3.5 C4.5(J48)

É natural e intuitivo classificar um padrão através de uma sequência de perguntas, na qual a próxima pergunta depende da resposta à questão atual, sendo esta uma abordagem particularmente útil quando os dados, sobretudo, não são contínuos, ou seja, quando as perguntas podem ser respondidas com sim ou não (DUDA; HART; STORK, 2012). Tal representação pode ser feita através de uma árvore de decisão, uma das estruturas de dados mais utilização na classificação de padrões (MURTY; DEVI, 2011).

Na Figura 4 é possível ver um exemplo de árvore de decisão. Nela estão presentes os elementos convencionais de uma classificação, como: os atributos ou *features* (*taste, color, shape, size*); os padrões ou valores das características (*sweet, yellow, thin, medium*); e por fim as classes (*Watermelon, Apple, Grape, Grapefruit, Lemon, Banana, Cherry*). Desta forma cada nó representa uma operação de decisão, sendo que os nós do tipo folha correspondem as classes ou ainda aos possíveis resultados finais da decisão, enquanto que o caminho percorrido da raiz da árvore até o nó folha corresponde a uma regra de decisão (MURTY; DEVI, 2011).

Durante a construção de uma árvore de decisão deve-se levar em consideração a sua simplicidade, de tal forma que a estrutura da árvore seja compacta e que através de

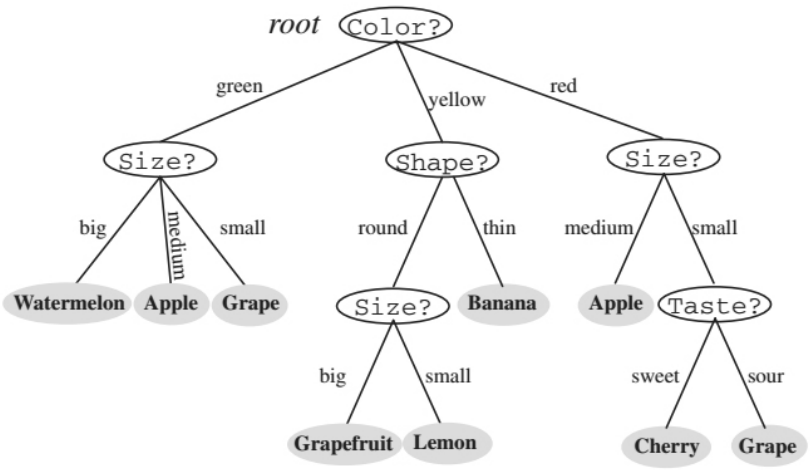


Figura 4 – Exemplo de árvore de decisão (DUDA; HART; STORK, 2012)

poucos nós seja possível modelar o conjunto de dados (DUDA; HART; STORK, 2012). Para isso é adotado o conceito de pureza, ou impureza, dos nós, de forma que quanto menor o valor da impureza, maior o ganho de informação que um determinado nó pode proporcionar à estrutura de decisão (MURTY; DEVI, 2011). Existem vários métodos para aferir a impureza de um nó, dentre eles: *gini impurity, misclassification impurity* e *entropy impurity*, sendo este último um dos mais utilizados. E vários outros aspectos podem ser consideradores durante a criação de uma árvore de decisão, por exemplo: o processo de divisão (*splitting*) da árvore; a escolha de um critério de parada da divisão; além do processo de poda. Mais detalhes sobre esses processos são apresentados por (DUDA; HART; STORK, 2012).

Existem vários algoritmos que implementam classificadores do tipo árvore de decisão, um deles é o C4.5 (QUINLAN, 1993). Também conhecido como J48, é uma das mais populares árvores de decisão (MURTY; DEVI, 2011; ALI et al., 2017).

O classificador C4.5 apresenta algumas vantagens com relação a outras árvores de decisão. Por exemplo, diferentemente do seu antecessor (ID3), o C4.5 consegue tratar tanto de características discretas quanto contínuas (QUINLAN, 1993; DUDA; HART; STORK, 2012).

Além disso o C4.5 fornece algumas formas de tratar uma possível ausência de um atributo em uma amostra dos dados. Assim, ao invés de simplesmente excluir uma determinada amostra que possui algum atributo ausente, o algoritmo pode fazer o aproveitamento apenas das informações existentes, sendo necessário para isso verificar o ganho

de informação que aquela amostra irá proporcionar ao classificador (QUINLAN, 1993; DUDA; HART; STORK, 2012). Mais detalhes sobre o algoritmo C4.5 estão disponíveis em (QUINLAN, 1993).

O C4.5 é uma boa alternativa como classificador de padrões para o *keystroke dynamics*, como já mostrado nos estudos de (ZHAO, 2006; HEMPSTALK; FRANK; WITTEN, 2008; FENG et al., 2012; ANTAL; SZABÓ; LÁSZLÓ, 2015)

## 3.6 *Random Forest*

O *Random Forest* é um classificador do tipo *ensemble* que consiste da combinação de várias árvores de decisão (BREIMAN, 2001) e para a criação das árvores de decisão o *random forest* adota a técnica *bagging* (GISLASON; BENEDIKTSSON; SVEINSSON, 2006).

O método *bagging* (BREIMAN, 1996), nome derivado de *bootstrap aggregation*, permite criar vários classificadores - um agregado de múltiplos classificadores de um mesmo tipo - utilizando dados *bootstrap* (MURTY; DEVI, 2011; DUDA; HART; STORK, 2012). Seguindo uma abordagem *bootstrap*, os novos conjuntos de treinamento são criados a partir da amostragem aleatória com substituição do conjunto de treinamento original (BREIMAN, 2001; DUDA; HART; STORK, 2012). Cada novo conjunto de treinamento tem tamanho inferior ao conjunto original ($n' < n$), devendo-se manter a distribuição dos dados (DUDA; HART; STORK, 2012). Em muitos casos adota-se o tamanho de 2/3 da base de treinamento original, podendo uma mesma amostrar ser selecionada mais de uma vez (MURTY; DEVI, 2011).

Além de contar com a aleatorização dos dados de treinamento, o *random forest* ainda adiciona uma outra camada de aleatoriedade à técnica *bagging* (LIAW; WIENER et al., 2002). No *random forest* cada árvore de decisão tem os seus nós - ou divisões de nós - selecionados a partir de um conjunto aleatório de *features* (BREIMAN, 2001), recomendando-se que a quantidade de características em cada seleção varie entre 1 a $\sqrt{p}$, onde $p$ é a quantidade total de *features* (FRIEDMAN; HASTIE; TIBSHIRANI, 2001).

Os resultados das classificações de cada árvore são consolidadas através de votação, sendo escolhida a classe majoritária (GISLASON; BENEDIKTSSON; SVEINSSON, 2006)

Algumas vantagens podem ser associadas a este classificador, dentre elas:

- Possui poucos parâmetros para ajustes, podendo ser definida a quantidade de árvores a serem geradas e o número de atributos da seleção de *features*. O tornando um classificador de uso amigável (LIAW; WIENER et al., 2002);
- É pouco sensível aos dados (GISLASON; BENEDIKTSSON; SVEINSSON, 2006);
- Ferramente efetiva de predição, podendo ser utilizada tanto para classificação quanto para regressão (BREIMAN, 2001).

Tais qualificações motivaram o uso do *random forest* em diversas áreas, inclusive para o tratamento do *keystroke dynamics* (ANTAL; SZABÓ; LÁSZLÓ, 2015; ZHOU et al., 2016; KOŁAKOWSKA, 2018)

## 3.7 k-NN

Uma outra abordagem de classificação é a de vizinhos próximos (*nearest neighbour*). A regra de vizinho próximo define a classe de uma determinada amostra a partir dos seus vizinhos (COVER; HART, 1967; DUDA; HART; STORK, 2012).

Alguns problemas de classificação podem ser tratados utilizando-se apenas um vizinho próximo (DUDA; HART; STORK, 2012), porém muitas vezes é útil levar em consideração mais de um vizinho (CORD; CUNNINGHAM, 2008), sendo essa técnica de classificação por vários vizinhos mais próximos conhecida como k-NN ou *k-Nearest Neighbour* (COVER; HART, 1967).

Para a execução do k-NN é necessário considerar cada amostra do conjunto de dados como pontos de um espaço de características multidimensional denotado por $D^n = x_1, ..., x_n$, sendo $n$ a quantidade total de *features* e $x_n$ cada amostra no espaço, também conhecido por protótipo (FRIEDMAN; HASTIE; TIBSHIRANI, 2001). Seja $x' \in D^n$ o protótipo mais próximo de um ponto de teste $x$, então, pela regra do vizinho mais próximo, a classe de $x'$ é atribuída ao ponto de teste $x$ (DUDA; HART; STORK, 2012).

No exemplo acima considerou-se apenas um vizinho mais próximo, ou seja, $k = 1$. Nos casos em que $k > 1$, após uma consulta entre os $k$ vizinhos mais próximos, por meio de votação, atribui-se a classe majoritária. Podem ainda ser aplicadas outras forma de

avaliação, por exemplo, aplicando-se um peso maior aos votos dos vizinhos que estiverem mais próximos ao ponto de teste (CORD; CUNNINGHAM, 2008).

Para se conhecer os vizinhos de determinado ponto de teste calcula-se a partir dele a distância até todos os demais pontos do espaço $D^n$, para isso utiliza-se métricas de distância, sendo uma das mais utilizadas a distância Euclidiana (mais detalhes sobre essa métrica encontra-se na Seção 3.1) (FRIEDMAN; HASTIE; TIBSHIRANI, 2001).

O k-NN algumas vezes pode requisitar uma quantidade considerável de memória, pois o conjunto de treinamento é processado em tempo de execução, sendo considerada uma técnica de aprendizagem lenta(*lazy learning*) (CORD; CUNNINGHAM, 2008; FRIEDMAN; HASTIE; TIBSHIRANI, 2001). Por outro lado é um processo transparente, de fácil implementação e *debug*, permitindo que através de uma análise dos vizinhos seja possível explicar a saída do classificador (CORD; CUNNINGHAM, 2008).

Este método de classificação também é bastante empregado ao *keystroke dynamics* (MONROSE; RUBIN, 2000; ANTAL; SZABÓ; LÁSZLÓ, 2015; ZHOU et al., 2016).

## 3.8 *Support Vector Machine*

O *Support Vector Machine* (SVM), ou Máquinha de Vetor de Suporte, é um tipo de algoritmo de aprendizagem que baseia-se na teoria de aprendizado estatístico (VAPNIK, 2013), comportando-se como uma função discriminante linear (CORD; CUNNINGHAM, 2008).

Admitindo-se inicialmente que um determinado conjunto de dados é linearmente separável, o SVM mapeia um vetor de entrada em um espaço de características de alta dimensionalidade e nesse espaço constrói um hiperplano ideal (*optimal hyperplane*) que separa os pontos, ou amostras, de acordo com sua classe (VAPNIK, 2013).

Na Figura 5 é ilustrado de forma simplificada o funcionamento do classificador SVM. Nela estão presentes pontos no espaço euclidiano que correspondem as amostras dos dados, enquanto que as formas geométricas de círculo e quadrado denotam suas respectivas classes. Separando esses pontos existe uma linha mais espessa indicando o hiperplano de decisão, que para a classificação representa o limiar de decisão. Ainda analisando a figura é possível perceber que existem alguns pontos que estão preenchidos, esses são os vetores de

suporte (*support vector*), sendo esses os pontos mais próximos do hiperplano e que por isso influenciam diretamente a localização da superfície de decisão. Por fim tem-se as margens, que mostram a distância entre o hiperplano e os seus pontos mais próximos (HAYKIN, 2007; LORENA; CARVALHO, 2007; DUDA; HART; STORK, 2012; VAPNIK, 2013).

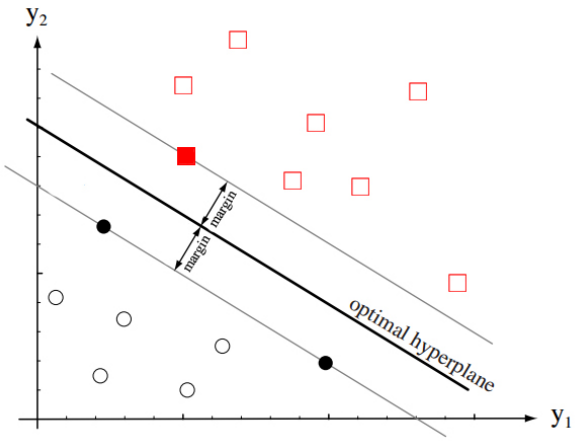


Figura 5 – Espaço de mapeamento SVM. Adaptado de (DUDA; HART; STORK, 2012)

As margens são medidas de confiança da previsão do classificador (LORENA; CARVALHO, 2007), pois infere a distância, ou seja, o nível de discriminação, entre os pontos das duas classes (FRIEDMAN; HASTIE; TIBSHIRANI, 2001). Por isso o treinamento do classificador SVM consiste em encontrar um hiperplano ótimo que proporcione o máximo distanciamento entre os dados e, consequentemente, uma melhor generalização do classificador (DUDA; HART; STORK, 2012). Sendo este um problema de otimização (VAPNIK, 2013).

Até aqui foi considerado que os dados utilizados no SVM são linearmente separáveis, porém em muitos casos do mundo real os dados podem não ser linearmente separáveis. Em casos como esse é possível utilizar margens flexíveis, ou suaves, (*Soft Margin*), proporcionando assim uma melhor acuracidade à classificação quando se tratando de dados não separáveis linearmente (MURTY; DEVI, 2011).

A técnica de margem suave proporciona um relaxamento nas restrições impostas aos problema de otimização do hiperplano. Este procedimento suaviza as margens do classificar permitindo que um determinado ponto permaneça no lado incorreto do hiperplano, ou seja, permite a ocorrência de alguns erros de classificação (LORENA; CARVALHO, 2007).

Uma outra solução para o tratamento dos dados não lineares é o uso de funções de Kernel (*kernel function*). Uma função de *kernel* recebe dois pontos do espaço de entrada e

os transforma através do produto escalar para o espaço de características (HAYKIN, 2007). Dentre os *kernels* mais utilizados estão os polinomiais, gaussianos ou RBF(*Radial Basis Function*) e os signoidais, lembrando que a escolha correta do *kernel* é de fundamental importância, pois afeta diretamente o desempenho da classificação (TAKAHASHI, 2012).

O SVM por ser uma função discriminante linear é ideal para separar padrões pertencentes a duas classes, ou seja, originalmente é um classificador binário (MURTY; DEVI, 2011), mas que através da utilização de algumas técnicas pode ser aplicado a problemas multiclasses (FRIEDMAN; HASTIE; TIBSHIRANI, 2001). Dentre as técnicas mais utilizadas para tratar da abordagem multiclasses está a estratégia decomposicional, que consiste em decompor um problema multiclasse em vários subproblemas binários criando-se múltiplos classificadores de forma que o resultado final da classificação parta da decisão majoritária (FRIEDMAN; GEIGER; GOLDSZMIDT, 1997; LORENA; CARVALHO, 2007; MURTY; DEVI, 2011).

## 3.9 *Multilayer Perceptron*

*Multilayer Perceptron* (MLP) é uma rede neural artificial que representa uma generalização do Perceptron (a forma mais simples de uma rede neural usada para classificação de padrões linearmente separáveis) (HAYKIN, 2007). Uma das principais vantagens do MLP sobre um perceptron simples é a sua capacidade de aprendizagem a partir de dados que não são linearmente separáveis, isso é possível graças as camadas ocultas existentes nessas redes. (MURTY; DEVI, 2011).

Como ilustrado na Figura 6, tipicamente a rede MLP consiste de um conjunto de camadas de neurônios, são elas: camada de entrada, que são compostas por nós de origem por onde a informação é apresentada à rede; uma ou mais camadas ocultas de nós computacionais; e por fim uma camada de saída que produz o resultado final. (FERREIRA, 2006; HAYKIN, 2007).

Quando uma informação, que pode ser um vetor de características, chega à camada de entrada ela logo é encaminhada à camada seguinte. Em um processo sucessivo de propagação de informação, a saída de um determinado neurônio serve de entrada para um neurônio seguinte a ele conectado, até que chegue a camada de saída gerando um resultado final, que na classificação representa uma classe. Essa abordagem de propagação

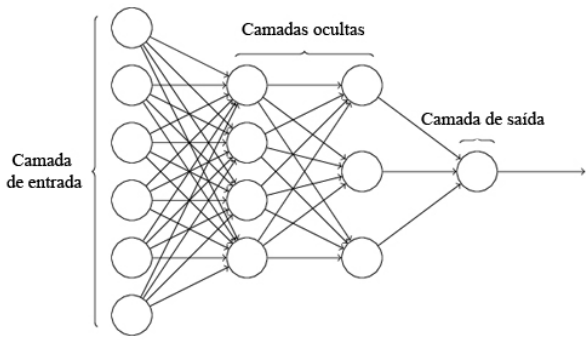


Figura 6 – Rede Multilayer Perceptron.

de informação sempre adiante é conhecida por *feedforward* e é empregada no MLP padrão (HAYKIN, 2007).

Dispostos em múltiplas camadas os neurônios encontram-se interligados por meio de várias conexões (sinapses), criando consequentemente *links* entre as camadas. Essas conexões estão associadas a pesos ajustáveis que armazenam o conhecimento representado no modelo e servem como ponderadores para os sinais de entrada dos neurônios da rede (FERREIRA, 2006; DUDA; HART; STORK, 2012).

Assim como diversos outros tipos de classificadores, antes de utilizar o MLP é necessário realizar o treinamento da rede. Um dos métodos mais populares para treinamento de redes multicamadas é o algoritmo *backpropagation*. Essa técnica que é baseado em gradiente descendente realiza ajustes nos pesos sinápticos como forma a minimizar uma medida de erro global da rede (FERREIRA, 2006; HAYKIN, 2007; DUDA; HART; STORK, 2012). Um exemplo de função de erro é formalizado na Equação 3.6 (DUDA; HART; STORK, 2012; FERREIRA, 2006),

$$E = 1/2 \sum_{i=1}^{N} (target_i - output_i)^2 \tag{3.6}$$

onde $target_i$ é a saída desejada da rede para o padrão $i$, $output_i$ é a saída da rede para o padrão $i$ e $N$ é a quantidade total de atributos. Um maior detalhamento sobre a técnica de aprendizagem por *backpropagation* e como ocorre a atualização dos pesos nas sinapses é apresentado por (HAYKIN, 2007).

O MLP é um algoritmo computacionalmente eficiente com complexidade na ordem

de O(W) (sendo W o número total de pesos sinápticos) e, assim como a técnica de *backpropagation*, é considerado como sendo de fácil implementação (HAYKIN, 2007; DUDA; HART; STORK, 2012), fazendo desse um algoritmo extremamente atrativo, sendo explorado pelas mais diversas áreas de conhecimento. Inclusive sendo bastante explorado para o reconhecimento de padrões de digitação (*keystroke dynamics* (YU; CHO, 2004; TROJAHN; ORTMEIER, 2013; ANTAL; SZABÓ; LÁSZLÓ, 2015).

# 4 Seleção de características

Bases de dados com grandes quantidades de características podem conter recursos irrelevantes ou redundantes à classificação, o que contribui para uma redução no desempenho da classificação (XUE; ZHANG; BROWNE, 2013). Uma forma de se solucionar esse problema, e permitir que somente características relevantes sejam utilizadas, é através da seleção de características (YU; CHO, 2004; XUE; ZHANG; BROWNE, 2013).

A seleção de características é aplicada em dados de dimensionalidade elevada antes de seguir para o processo de classificação (SENATHIPATHI; BATRI, 2014). Este pode ser considerando como um problema de otimização que possui como objetivo a identificação de um conjunto ótimo ou quase ótimo de características que permita maximizar uma determinada medida (SHANMUGAPRIYA; PADMAVATHI, 2011).

A partir da seleção de características é possível se construir um modelo de classificação menos complexo, além de viabilizar a redução do tempo de treinamento dos classificadores (DASH; LIU, 1997; XUE; ZHANG; BROWNE, 2013).

Em alguns casos o processo de seleção de características pode ser custoso ou até mesmo inviável de ser executado (DASH; LIU, 1997), sobretudo quando existem relações complexas entre as *features* (XUE; ZHANG; BROWNE, 2013), chegando a ser um problema NP-completo (GHEYAS; SMITH, 2010).

Os métodos de seleção de características podem ser classificados em dois categorias: abordagem de filtro e abordagem *wrapper* (UNLER; MURAT, 2010; XUE; ZHANG; BROWNE, 2013; KOŁAKOWSKA, 2018).

Nas abordagens de filtro as *features* são selecionada com base em determinados critérios estatísticos, sendo as características escolhidas conforme sua relevância (UNLER; MURAT, 2010; GIOT; EL-ABED; ROSENBERGER, 2011). Os métodos de filtragem são rápidos, mas não possuem robustez contra interações complexas e redundância entre as característica, além disso, é difícil se definir um limiar de corte que determine quando excluir ou não uma característica (DASH; LIU, 1997).

A abordagem *wrapper* inclui um algoritmo de aprendizagem e ou classificação no procedimento de avaliação (XUE; ZHANG; BROWNE, 2013). O algoritmo realiza

uma busca no espaço de possíveis soluções a fim de identificar o melhor subconjunto de características (GIOT; EL-ABED; ROSENBERGER, 2011; KOŁAKOWSKA, 2018), para isso, continuamente, seleciona e avalia a qualidade do subconjunto através da acuracidade da classificação (KOHAVI; JOHN, 1997; GHEYAS; SMITH, 2010).

Apesar da abordagem por filtro ser computacionalmente menos custosa, geralmente a abordagem *wrapper* é mais eficiente, alcançando melhores resultados (DASH; LIU, 1997; XUE; ZHANG; BROWNE, 2013).

Vários trabalhos já foram desenvolvidos com o objetivo de analisar técnicas de seleção de características aplicadas ao *keystroke dynamics*. A seguir alguns desses trabalhos são apresentados.

No trabalho de (SHANMUGAPRIYA; PADMAVATHI, 2011) são propostos 3 formas de seleção de características para o *keystroke dynamics*, sendo eles baseados em *Particle Swarm Optimization* (PSO) (KENNEDY; EBERHART, 1995), *Ant Colony Optimization* (ACO) (DORIGO; MANIEZZO; COLORNI, 1996) e *Genetic algorithm* (GA) (HOLLAND, 1975), enquanto que o *Extreme Learning Machine* (ELM) (HUANG; ZHU; SIEW, 2006) é utilizado como função objetivo para seleção de características.

Os dados utilizados foram obtidos a partir da digitação de 103 indivíduos, cada usuário digitando de 7 a 503 vezes o *password* “drizzle” (com uma média de 26 repetições por usuário). As características obtidas foram a *hold time*, *up-down*, *down-down*, trígrafos e *Virtual Key Force* (VTF) - características que é calculada com base na distância entre as teclas digitadas e velocidade de digitação das características *up-down* - totalizando 43 atributos.

Após a execução do experimentos obteve-se a partir do GA uma redução de 34,88% das *features* (28 características selecionadas), enquanto que o PSO conseguiu uma minimização de 30,23% (30 características selecionadas) e por fim o ACO alcançou uma taxa de redução de 46.51%.

Como já mencionada na Seção 4, é esperado que a seleção de características proporcione uma redução na quantidade total de atributos, o que foi alcançado claramente no trabalho de (SHANMUGAPRIYA; PADMAVATHI, 2011). Porém os autores não deixaram claro o objetivo da função de aptidão, se a otimização seria para apenas minimizar a quantidade de *features* ou se o objetivo é de minimização, ou ainda alguma outra métrica.

Ao se adotar apenas a redução de atributos como função objetivo isso pode gerar problemas à classificação, pois nessas condições os métodos de busca podem minimizar drasticamente as características, gerando uma perda substancial de informações e consequentemente dificultando, ou até mesmo impossibilitando, o reconhecimento de padrões.

Um outro trabalho que trata da seleção de características para o *keystroke dynamics* é apresentado por (SENATHIPATHI; BATRI, 2014). Nele os autores apresentam duas implementações de métodos de seleção, uma baseada em algorítimo genético e a outra um PSO. As seleções de características foram executadas a partir de uma base de dados composta por características do tipo *hold time*, *up-down*, *trígrafos* e *Virtual Key Force* (VTF). Ambos os métodos apresentados no trabalho utilizaram o classificador SVM como função de *fitness*, tomando-se como objetivo a maximização da taxa de redução de características. A taxa de redução de *features* é expresso na Equação 4.1, sendo $F_t$ o número total de *features* e $F_s$ a quantidade de características selecionadas (SENATHIPATHI; BATRI, 2014).

$$RF = \frac{F_t - F_s}{F_t} \tag{4.1}$$

Como resultado do trabalho ambos os métodos conseguiram obter altas taxas de redução de *features*, porém a seleção de características implementada com o PSO conseguiu uma taxa um pouco mais elevada alcançando uma redução de 96,3%, enquanto que o algorítimo genético atingiu uma percentual de 95.6%.

Um outro estudo sobre seleção de características aplicado ao *keystroke dynamics* é desenvolvido por (DARABSEH; NAMIN, 2015). Os seletores de características utilizados foram: PSO, *Best First Forward (BFF) e Backward(BFB)*, GA e algoritmos gulosos (*Greedy Forward(GF) e Greedy Backward(GB)*). Já os classificadores que foram utilizados como funções de *fitness* foram: *Naive Bayes* (NB), k-NN e *Support Vector Machine (SVM)*. Assim todos os métodos de seleção foram utilizados em combinação com cada classificador. A maximização da acuracidade foi adotada como função objetivo nesse trabalho.

Os dados para o *keystroke dynamics* foram obtidos a partir da digitação de 28 indivíduos, cada um digitando 5000 caracteres. As características extraídas da digitação foi a *hold time*, *up-down*, *down-down* e a duração total para digitação de cada palavra, totalizando 80 características (20 de cada tipo).

Após a execução dos experimento foi possível analisar a eficiência dos métodos tanto sob a ótica de redução de características tanto pelo nível de acuracidade da classificação. O algoritmo GF foi o que alcançou maior taxa de redução de *features* com 62,5% (SVM), 73,25% (NB) e 82,5% (k-NN). As demais maiores taxas de diminuição foram BFB com 48,75% (NB), PSO com 57,5% (k-NN), GA com 58,75% (k-NN) e por fim o BFF com de 62,25%(k-NN). Já o algoritmo GB foi o que conseguiu uma menor taxa de redução obtendo valores de 0,8% (SVM), 35% (NB) e por fim 11,25% (k-NN).

Analisando os resultados da acuracidade da classificação o melhor resultado foi obtido pelo PSO com 94,64% seguido pelo GA com 94,04%, ambos utilizando o classificador *Naive* Bayes. Já o algoritmo *greedy forward* foi o que alcançou os resultados mais baixos com 85,77% (SVM), 87,5%(k-NN) e 88,09% (NB), sendo ainda pior que as classificações sem seleção de características. As melhores porcentagens da acuracidade dos demais seletores foram BFF com 92,85% (NB e k-NN), BFB com 93,45% (NB) e GB 93,89% (NB).

# 5 Algoritmos de busca utilizados

## 5.1 Algoritmo Genético

Os algorítimos genéticos (*Genetic algorithm* - GA), inicialmente propostos por Holland (HOLLAND, 1975), são algoritmos de busca baseados na mecânica da seleção natural e genética (GOLDBERG, 1989). Por esse motivo muitos dos termos empregados no GA são provenientes dos estudos da genética (NG; PERERA, 2003).

Esses algoritmos são empregados como um modelo computacional em que uma população - representada conceitualmente por indivíduos, também denominados de cromossomos - corresponde a um conjunto de soluções candidatas para um determinado problema que progridem de forma evolutiva para as melhores soluções (SHANMUGAPRIYA; PADMAVATHI, 2011). De acordo com a terminologia genética, cada cromossomo é composto por um conjunto de genes, sendo que cada gene representa um atributo do modelo a ser ajustado (NG; PERERA, 2003).

A representação de uma solução candidata, ou de um indivíduo, pode ser feita de várias formas, sendo importante escolher a representação correta para o problema que está sendo resolvido (EIBEN; SMITH, 2003). Dentre as principais representações tem-se: a binária, quando o gene pode assumir valor 0 ou 1; inteira, utilizada quando o gene deve receber um valor inteiro; por fim tem-se a representação de ponto flutuante, empregada quando se deseja associar ao gene um número real (EIBEN; SMITH, 2003).

A Figura 7 mostra resumidamente o esquema de execução de um algoritmo genético. A seguir são descritos mais detalhes sobre as etapas do seu funcionamento.

### 5.1.1 Inicialização da população

A princípio para a execução do GA é necessário inicializar todos os indivíduos da população. É possível fazer uso de métodos heurísticos como forma a se inicializar os indivíduos com base em um conhecimento prévio (NG; PERERA, 2003), alguns desses métodos encontram-se disponíveis em (EIBEN; SMITH, 2003). Porém Eiben (2003) recomenda que uma população seja inicializada aleatoriamente, defendendo que geralmente

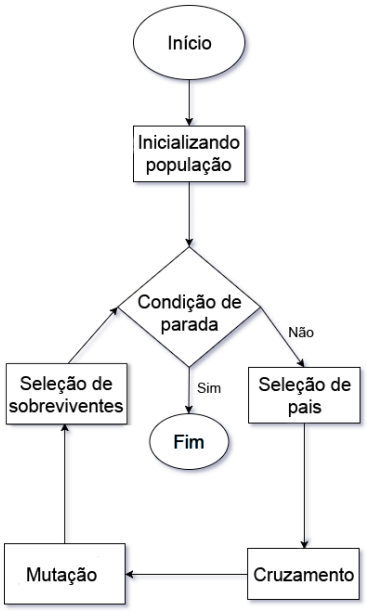


Figura 7 – Fluxo do funcionamento de um algoritmo genético. Adaptado de (DEB, 1998).

os esforços desempenhados pelos métodos heurísticos não justificam possíveis ganhos.

Através da Figura 8 é possível se verificar o motivo pela qual os esforços adicionais dos métodos heurísticos podem não superar suas vantagens. Sabendo-se que o nível $a$ representa uma população inicializada aleatoriamente, enquanto que $b$ indica uma inicialização por heurística, observa-se que apesar do método heurístico proporcionar inicialmente uma melhor população, o seu nível de desempenho é alcançado por uma inicialização aleatória após $k$ interações (EIBEN; SMITH, 2003).

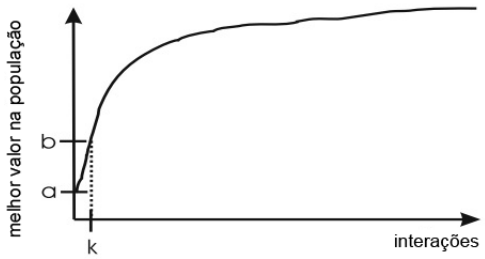


Figura 8 – Comparação do desempenho da população inicializada aleatoriamente e por método heurístico. Adaptado de (EIBEN; SMITH, 2003).

Uma vez inicializada a população segue-se o processo iterativo do GA.

### 5.1.2 Função de *fitness*

A função de *fitness* é tecnicamente o procedimento que atribui um valor de qualidade aos indivíduos (EIBEN; SMITH, 2003), sendo este o principal mecanismo de avaliação

dos cromossomos (TANG et al., 1996). Por isso a função de *fitness* deve ser elaborada de acordo com o problema a qual se deseja tratar (BEASLEY; MARTIN; BULL, 1993; TANG et al., 1996). Esta função após analisar o cromossomo retorna um valor que indica a aptidão do indivíduo (BEASLEY; MARTIN; BULL, 1993).

O termo aptidão é geralmente associado à maximização e isso pode causar uma terminologia pouco intuitiva se o problema remeter a minimização de uma medida, contudo, matematicamente, a mudança entre maximização e minimização é trivial (EIBEN; SMITH, 2003).

As Equações 5.1 e 5.2 descrevem respectivamente em termos algébricos uma abordagem de minimização e maximização (MAVROVOUNIOTIS; LI; YANG, 2017). Para isso considera-se $x$ uma solução candidata e $X$ o conjunto de soluções possíveis, $x^*$ é a melhor solução e $f(x)$ uma função de avaliação de aptidão ou função de *fitness*. Analisando a Equação 5.1 inferi-se que em um problema de minimização a melhor solução possui valor de *fitness* inferior ou igual as demais soluções candidatas, enquanto que no problema de maximização, de forma análoga, a melhor solução possui valor de aptidão maior ou igual as outras soluções candidatas, como representado na Equação 5.2.

$$f(x^*) \leq f(x) \quad \forall\, x \in X \tag{5.1}$$

$$f(x^*) \geq f(x) \quad \forall\, x \in X \tag{5.2}$$

### 5.1.3 Seleção de pais

A seleção de pais é o processo que atribui oportunidade reprodutiva para cada indivíduo (BEASLEY; MARTIN; BULL, 1993), para isso os indivíduos pais geralmente são escolhidos com base na sua qualidade, ou aptidão (EIBEN; SMITH, 2003).

A seleção de pais é tipicamente probabilístico, de tal forma que os indivíduos de alta qualidade tenham maiores chances de se tornarem pais (EIBEN; SMITH, 2003). Vários métodos podem ser aplicados à seleção dos pais, estando alguns deles descritos em (GOLDBERG, 1989; TANG et al., 1996; YU; CHO, 2003).

### 5.1.4 Cruzamento (*Crossover*)

Cruzamento é um operador de recombinação que combina subpartes de dois indivíduos para produzir descendentes que contenham partes do material genético de ambos os pais (TANG et al., 1996). Esse operador é considerado como um dos mecanismos mais importante para os algorítimos evolutivos, sendo o cruzamento um operador que diferencia os algorítimos evolucionários dos demais algorítimos de otimização (EIBEN; SMITH, 2003).

Existem vários métodos de cruzamento, um deles é o cruzamento por 1 ponto, demonstrado na Figura 9. O funcionamento desse método consiste inicialmente da escolha de um número aleatório $r$ dado por um intervalo entre 1 e $l$-1 (sendo $l$ a quantidade total de genes do cromossomo, ou seja, o tamanho do indivíduo), em seguida, é feita a divisão de ambos os pais no ponto $r$ e então geram-se dois filhos a partir das frações dos indivíduos pai (EIBEN; SMITH, 2003). No exemplo da Figura 9 cada cromossomo possui 9 genes, sendo esse o valor de $l$, enquanto que o ponto de corte $r$ é igual a 4.

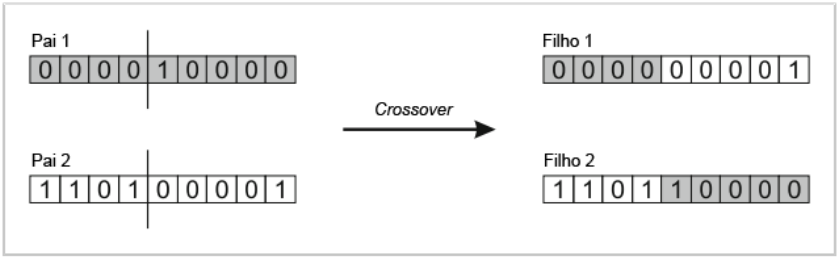


Figura 9 – Cruzamento por 1 ponto.

Outros métodos de cruzamento estão disponíveis em (EIBEN; SMITH, 2003).

### 5.1.5 Mutação

O princípio básico do operador de mutação é alterar os genes dos cromossomos, buscando-se obter melhorias ao indivíduo (KARAKATIČ; PODGORELEC, 2015; TANG et al., 1996). Sob a ótica de otimização computacional, o operador de mutação resulta na ampliação em pequenas proporções da região de busca (KARAKATIČ; PODGORELEC, 2015).

Um exemplo de mutação é mostrado na Figura 10. Considerando uma representação binária o operador de mutação realiza ajustes em todos os genes (*bit* a *bit*), de tal forma que após um teste de probabilidade um bit é alterado (TANG et al., 1996).

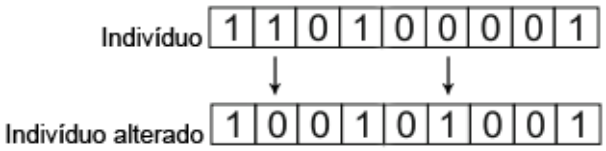


Figura 10 – Operação de mutação bit a bit.

Na natureza a mutação tem poucas chances de acontecer, por isso, geralmente nos algoritmos genéticos baixas taxas de mutação são utilizadas, além de que altas taxas de mudação podem fazer com que informações importantes dos cromossomos sejam perdidas (KARAKATIČ; PODGORELEC, 2015).

### 5.1.6 Seleção de sobreviventes

Na seleção de sobreviventes são escolhidos os indivíduos que irão compor a população da próxima geração, para isso duas estratégias podem ser adotadas, são elas: reprodução estável (*Steady-State*) e substituição de geração (*Generational-Replacement*) (TANG et al., 1996).

Na abordagem *Steady-State* a seleção dos indivíduos sobreviventes é desempenhada com base no valor de aptidão de cada indivíduo, de tal forma que os melhores indivíduos tenham maiores chances de seguir para a geração seguinte. Já na estratégia *Generational-Replacement* a cada geração toda a população é substituída, sendo os indivíduos filhos a população da geração seguinte (TANG et al., 1996; EIBEN; SMITH, 2003).

Este é o último passo no processo de execução de uma geração do GA. Após a seleção de sobreviventes inicia-se novamente todo o ciclo do algoritmo genético até que uma condição de parada seja alcançada. A escolha do critério de parada é arbitrário, podendo a finalização do algoritmo ser definida a partir de um limite máximo de gerações, ou um valor mínimo de qualidade (*fitness*) alcançado pelos indivíduos, ou ainda qualquer outro critério que se deseje (KARAKATIČ; PODGORELEC, 2015).

## 5.2 *Particle Swarm Optimization*

O *Particle Swarm Optimization* (PSO) é um algoritmo inspirado no comportamento social dos pássaros, simulando uma busca por alimentos (KENNEDY; EBERHART, 1995).

No PSO cada indivíduo da população é denominado de partícula e representa uma

solução candidata para um determinado problema. A partícula pode ser descrita através de um vetor $m$-dimensional, onde $m$ é o número de parâmetros a serem otimizados (ABIDO, 2002). Assim, de forma análoga ao voo de um pássaro, cada partícula move-se através de um hiperplano de soluções em busca da melhor solução. E para se orientar cada partícula guarda na memória a sua melhor posição(*pbest*) e a posição do melhor indivíduo da população (*gbest*), sendo essas informações utilizadas na atualização da velocidade e posição da partícula (KENNEDY; EBERHART, 1995; ABIDO, 2002).

Na Equação 5.3 (LEE et al., 2008) é formalizado o cálculo para a atualização da velocidade de cada partícula.

$$v_{i,j}(t+1) = v_{i,j}(t) + c_1 r_1(p_{besti,j} - xi, j(t)) + c_2 r_2(g_{besti,j} - x_{i,j}(t)) \tag{5.3}$$

Sendo que $i$ é o índice da partícula no enxame ($i = i, ..., n$) e $j$ é o índice de uma posição na partícula, ou seja, um atributo ou parâmetro da partícula ($j = 1, ..., m$), $t$ representa o número da interação, $v_{i,j}(t)$ é a velocidade da partícula e $x_{i,j}(t)$ descreve o valor da posição da partícula. Já $r_1$ e $r_2$ são números aleatórios distribuídos uniformemente entre 0 e 1. Por fim $c_1$ e $c_2$ representam os coeficientes de aceleração, onde $c_1$ é um componente cognitivo, que define o nível de influência da melhor posição pessoal para o indivíduo, enquanto que $c_2$ é um componente social, que pondera a influência da melhor posição global de uma partícula em relação à posição de cada indivíduo (LEE et al., 2008; ZHANG et al., 2014).

Na Equação 5.4 é descrito o método para atualização da posição da partícula, onde $x_{i,j}(t+1)$ é a posição resultante da partícula $x_{i,j}$ no instante $t+1$, e $x_{i,j}(t)$ representada a posição da partícula no instante $(t)$, por fim $v_{i,j}(t+1)$ denota a velocidade da partícula no instante $t+1$ (LEE et al., 2008).

$$x_{i,j}(t+1) = x_{i,j}(t) + v_{i,j}(t+1) \tag{5.4}$$

Após a proposta original do PSO vários outras alterações surgiram para melhorar o desempenho desse algoritmo de busca. Uma dessas alterações foi proposta por (SHI; EBERHART, 1998), que consiste na inclusão de um novo parâmetro $w$ no cálculo da velocidade da partícula a fim de que se tenha um equilíbrio entre uma busca global e

uma busca local (exploração e explotação). Dessa forma esse parâmetro, conhecido como peso de inércia, ou parâmetro de inércia, pode proporcionar uma melhoria na forma como o algoritmo PSO converge para uma solução ao suavizar a trajetória das partículas, controlando a influência da velocidade anterior sob a velocidade atual (LEE et al., 2008; ZHANG et al., 2014). O peso inercial pode ser um valor constante positivo ou uma função positiva linear ou não linear em relação ao tempo (SHI; EBERHART, 1998). Ajustando o cálculo da velocidade da partícula com o parâmetro de peso inercial obtem-se a Equação 5.5.

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_1(p_{best,ij} - xi, j(t)) + c_2 r_2(g_{best,i,j} - x_{i,j}(t)) \tag{5.5}$$

No Algoritmo 1 é apresentado o pseudocódigo do PSO (ZHANG; WANG; JI, 2015). Inicialmente as partículas devem ser inicializas com valores aleatórios seguindo uma distribuição uniforme, podendo ainda obedecer um intervalo de Limite Inferior (LI) e Limite Superior (LS) correspondentes às limitações do espaço de busca. Posteriormente os valores de *pbest* de cada partícula recebem o valor 0 e esse mesmo valor é definido para o *gbest* (conhecimento compartilhado entre todas as partículas). Depois disso começa o processo iterativo do PSO, sendo o algoritmo executado até que um critério de parada seja alcançado. Em cada iteração um novo valor é gerador aleatoriamente (distribuição uniforme) para $r_1$ e $r_2$, um número real que varia entre 0 e 1. São atualizadas a velocidade e posição de cada partícula, conforme descrito na Equação 5.5 e 5.4 respectivamente. Após isso cada partícula é avaliada por uma função *fitness* a fim de se mensurar a qualidade da solução, ou da partícula, se a posição atual da partícula é a melhor já registrada então atualiza-se o seu valor de *pbest*. O mesmo é feito para *gbest* comparando a partícula no instante atual $t$ com a melhor solução global, se o valor da partícula for superior à solução global então atualiza-se o valor de *gbest*. Observando o algoritmo percebe-se que na comparação entre os valores das partículas com o *pbest* e *gbest* utiliza-se o operador lógico "menor que", pois a implementação do PSO está considerando um problema de otimização por minimização de valores. Porém a mudança do algoritmo para tratar de maximização da função de *fitness* é trivial, basta utilizar o operador lógico "maior que".

Até aqui o PSO foi apresentado sob uma perspectiva que geram, sobretudo, soluções com valores reais, porém com alguns ajustes é possível direcionar o *particle swarm optimization* para o tratamento de problemas com espaço de características binárias.

**Algorithm 1** Particle Swarm Optimization

1. 1: **function** INICIALIZACAO(P) ▷ P é a enxame de partículas.
   2: **for** $i = 1$ to $N_p$ **do**
   3: $P[i] = U(LI, LS)$ ▷ LS (Limite Superior), LI (Limite Inferior)
   4: $P[i].pbest = 0$
   5: **end for**
   6: $gbest = 0$
   7: **end function**
   8: **function** ITERACAO(P)
   9: **while** criterio de parada **do**
   10: **for** $i = 1$ to $size(P)$ **do**
   11: $r_1, r_2 = U(0, 1)$ ▷ Valor flutuante entre 0 e 1
   12: Atualizar velocidade da partícula ▷ Equação 5.5
   13: Atualizar posição da partícula ▷ Equação 5.4
   14: **if** $f(P[i]_t) < f(P[i].pbest_t)$ **then**
   15: $P[i].pbest_t = P[i]_t$
   16: **end if**
   17: **if** $f(P[i]_t) < f(gbest)$ **then**
   18: $gbest = P[i]_t$
   19: **end if**
   20: **end for**
   21: $t = t + 1$ ▷ Variável para controle das iterações
   22: **end while**
   23: **end function**
   24: A saída do algoritmo é o valor do *gbest*, a melhor solução encontrada

Um PSO binário (BPSO) é proposto por Kennedy e Eberhart (1997). Os autores consideraram que a velocidade da partícula representa a probabilidade de um *bit* ter um valor 0 ou 1. No BPSO o cálculo da velocidade permanece inalterado, conforme Equação 5.5, porém agora a posição é definida de acordo com a Equação 5.6.

$$x_{i,j}(t+1) = \begin{cases} 0 & \text{se rand}() \geq S(v_{i,j}(t+1)) \\ 1 & \text{se rand}() < S(v_{i,j}(t+1)) \end{cases} \tag{5.6}$$

Onde $S(v_{i,j})$ é uma função sigmoide de limitação de transformação (transformando a velocidade em probabilidade), definida conforme Equação 5.7, e $rand()$ é uma função aleatória regida por uma distribuição uniforme que resulta valores entre 0 e 1. Desta forma se o valor de $rand()$ for maior ou igual que o retorno da função $S(v_{i,j})$ a posição da partículaé igual a 1, senão recebe o valor 0 (KENNEDY; EBERHART, 1997; LEE et al.,

2008).

$$S(v_{i,j}(t+1)) = \frac{1}{1+e^{-v_{i,j}(t+1)}} \tag{5.7}$$

O compartilhamento das informações entre as partículas do PSO ocorre de acordo com uma topologia social (LIU et al., 2016). As topologias mais comuns são a de melhor global (*gbest*) e melhor local (*lbest*), também conhecidas, respectivamente, por topologia completamente conectada e topologia anel (KENNEDY; MENDES, 2002; LIU et al., 2016).

Na topologia *gbest* a trajetória de cada partícula é influenciada pela melhor solução global, de forma que todos as partículas estejam totalmente conectadas umas com as outras. Podendo assim essa topologia ser representada por um grafo totalmente conectado. Em termos de desempenho, quando comparada a outras topologias, a topologia *gbest* tende a proporcionar uma convergência mais rápida para uma solução ótima, mas também é mais suscetível a convergir para ótimos locais, porém, ainda assim, é uma das topologias mais utilizadas (KENNEDY; MENDES, 2002; MEDINA; PULIDO; RAMÍREZ-TORRES, 2009).

Enquanto que na topologia *lbest* cada partícula é influenciada apenas pelos seus vizinhos de tal forma que cada partícula esteja apenas conectada aos seus $k$ vizinhos mais próximos, sendo geralmente utilizado um $k = 2$ (LEE et al., 2008; LIU et al., 2016). Essa topologia tende a evitar a convergência prematura das partículas (ZHANG; WANG; JI, 2015).

# 6 Metodologia

## 6.1 Base de dados

A base de dados utilizada nesse trabalho foi criada por (ANTAL; SZABÓ; LÁSZLÓ, 2015) e encontra-se disponível publicamente[1]. Os dados foram coletados a partir da digitação em dispositivos móveis, sendo empregado um total de 42 pessoas (37 usuários utilizaram *tablet* e 5 *smartphone*). As informações foram extraídas a partir da digitação da senha ".tie5Roanl" (sem aspas), sendo repetida 30 vezes pelos usuários em cada seção. Todos as amostrada de digitação que continham deleções (uso de teclas como *backspace* ou *delete*) foram excluídas da base de dados. Ao final foi construída uma base de dados com 51 amostras para cada usuário.

Na Tabela 1 é apresentada a relação de todas as características da base de dados, bem como os seus respectivos quantitativos. Um total de 71 características foram extraídas da digitação, sendo elas: *hold time* (H), *down-down* (DD), *up-down* (UD), *hold pressure* (P) característica que descrimina a pressão exercida na tela do dispositivo, *finger area* (FA) que é a área que o dedo ocupa na tela do dispositivo, além dessas foram utilizadas também como características os valores médios das *features hold time* (MH), *finger area* (MFA) e *hold pressure* (MP).

Tabela 1 – Lista das características da base de dados

| **Tipo de característica** | **Número de características** |
|---|---|
| *Hold time (H)* | 14 |
| *Down-down* (DD) | 13 |
| *Up-down* (UD) | 13 |
| *Hold pressure* (P) | 14 |
| *Finger area* (FA) | 14 |
| Média *hold time* (MH) | 1 |
| Média *finger area* (MFA) | 1 |
| Média *hold pressure* (MP) | 1 |
| **Total** | **71** |

Apesar da senha possuir 10 caracteres, ao longo da digitação são pressionadas um total de 14 teclas nos dispositivos móveis, isso ocorre devido à necessidade do uso das teclas

[1] http://www.ms.sapientia.ro/ manyi/keystroke.html

especiais [123?], [abc] e [Shift] como forma a habilitar a digitação de números, a digitação de letras e o uso de letras maiúsculas e minúsculas, respectivamente. Isso fica evidenciado na Figura 11 que mostra o processo de digitação da senha utilizada na aquisição dos dados.

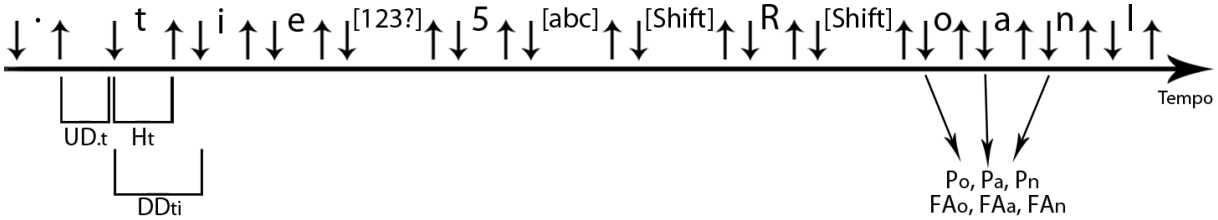


Figura 11 – Processo de digitação da senha ".tie5Roanl" nos dispositivos móveis

Na Figura 11 a seta para baixo indica o início da interação com uma tecla, ou seja, quando ela está sendo pressionada, enquanto que a seta para cima indica o final da interação, ou seja, quando a tecla é solta. Nela é ilustrada também as características baseadas em intervalos de tempos (H, UD e DD), de tal forma que a anotação $H_t$ indicada a *feature hold* da tecla $t$. De forma análoga a mesma anotação é adotada para as características P e FA ($P_t$ pressão na tela ao digitar a tecla $t$ e $FA_t$ área na tela ao pressionar a tecla $t$), essas *features* apesar de não serem calculadas em termos temporais, caracterizam-se por estarem estritamente relacionadas a uma única tecla. Já as características do tipo UD e DD são denotadas com notação semelhante às *features* anteriormente apresentadas, por exemplo, a notação $UD_{.t}$ e $DD_{ti}$ representam, respectivamente, o intervalo de tempo entre soltar a tecla "." e pressionar a tecla $t$, e o tempo decorrido entre pressionar a tecla $t$ e pressionar a tecla $i$.

É necessário salientar que na Figura 11 uma seta para baixo sempre vem sucedida de uma seta para cima e isso foi adotado como forma a facilitar a visualização do processo de digitação, porém, durante a construção da base de dados é possível que alguma tecla tenha sido pressionada antes mesmo que a anterior tenha sido solta, ou seja, ocorrer de uma seta para cima estar seguida de outra seta para cima, como já explicado na Seção 2.4.

## 6.2 *Particle Swarm Optimization* proposto

O PSO proposto neste trabalho para seleção de características é baseados no PSO padrão que implementa fator inercial, cujo o funcionamento já foi explicado na Seção 5.2. Contudo foi definido um limite superior e inferior para as posições das partículas, conforme

descrito na Equação 6.1.

$$x_{i,j}(t+1) = \begin{cases} -2 & \text{se } x_{i,j}(t+1) \leq -2 \\ 2 & \text{se } x_{i,j}(t+1) \geq 2 \end{cases} \tag{6.1}$$

Um limite para a posição da partícula foi implementado como forma a restringir o espaço de busca, isso porque a seleção de características trata-se de um problema com soluções binárias, ou seja, uma determinada características pode ou não estar presente em um vetor de características, não sendo necessário o uso de grandes valores reais.

Apesar de se tratar de uma representação binária ainda assim foi mantido uma delimitação superior e inferior de 2 e -2, respectivamente, do espaço de busca, como forma a proporcionar uma "folga" da navegação da partícula. Dessa forma quanto mais próximos os valores dos atributos estiveram dos extremos mais fortemente será a sua relação com o seu respectivo valor binário.

Na Expressão 6.2 é apresentada uma representação da partícula do PSO proposto. É possível observar que os elementos contidos na partícula possuem valores reais e são utilizados assim ao longo de todo o processo do algoritmo. Contudo no momento da avaliação da partícula - papel desempenhado pelos classificadores - é necessário realizar uma conversão dos valores a fim de se obter uma representação binária da solução.

$$vetor_{solucao} = [0,82; -1,16; 2,72; 1,17; -0,99; 2,61; -0,42; 0,53; 1,96; -1,69] \tag{6.2}$$

Na expressão 6.3 é mostrada a mesma partícula mencionada anteriormente só que agora com valores binários.

$$vetor_{solucao} = [1; 0; 1; 1; 0; 1; 0; 1; 1; 0] \tag{6.3}$$

A conversão dos valores reais para binários ocorre de acordo com a Equação 6.4, onde $x_f$ é um atributo da partícula com valor real, $x'_f$ a sua equivalência binária e $f \in F = (1, 2..., N)$ um atributo da partícula ou uma *feature* do vetor de características $F$ de tamanho $N$, adotando-se um limiar de decisão igual a 0, de tal forma que se $x_f >= 0$

então $x'_f = 1$ senão $x'_f = 0$.

$$x'_f = \begin{cases} 1 & \text{se } x_f \geq 0 \\ 0 & \text{senão} \end{cases} \tag{6.4}$$

## 6.3 Algoritmo Genético proposto

O segundo método para seleção de características do *keystroke dynamics* proposto nesse trabalho é baseado em Algoritmos Genéticos (GA). Como já mencionada na Seção anterior, a seleção de características é tratada sob uma abordagem binária podendo ou não uma característica ser selecionada. Por isso o primeiro passo a ser feito na criação do GA é a representação da solução. Aqui o cromossomo é representado por um vetor de genes que pode assumir o valor 1 ou 0, indicando a presença de uma determinada característica ou não, respectivamente. Porém diferentemente do PSO aqui proposto, no GA essa representação será utilizada ao longo de toda a execução do algoritmo. Cada uma das características da base de dados é então representada por um gene, totalizando assim 71 genes em cada cromossomo.

No GA aqui proposto todos os cromossomos são inicializados aleatoriamente por uma distribuição uniforme. Uma vez inicializada a população de cromossomos segue-se com o processo iterativo do algoritmo, tomando os passos já descritos na Seção 5.1.

Posteriormente é executado o processo de seleção de pais, contudo para se realizar essa etapa é necessário avaliar a qualidade de todos os cromossomos, ou seja, o nível de discriminação dos vetores de características, através dos classificadores apresentados na Seção 6.4. Após a etapa de avaliação é possível então dar prosseguimento à seleção de pais.

O método para a escolha dos pais aqui utilizado foi o de seleção por roleta. Nesse método cada indivíduo é associado a um segmento de uma roleta cujo tamanho da porção é diretamente proporcional ao seu valor de *fitness*, desta forma os cromossomos que tiverem níveis mais altos de aptidão terão maiores chances de serem selecionados. Salientando que durante a execução do operador de roleto as taxas de aptidão dos cromossomos foram padronizadas a fim de se evitar a anomalidade de perda de pressão da seleção.

Através da Equação 6.5 é possível se obter a porção de cada indivíduo na roleta, sendo $p_i$ o segmento da roleta ou a probabilidade de um indivíduo $i$ ser selecionado,

enquanto que $f_i$ corresponde ao valor de *fitness* do indivíduo e por fim o denominador da equação denota o somatório de todos os valores de *fitness* dos indivíduos da população.

$$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{6.5}$$

Uma vez calculada todas as porções dos cromossomos, a roleta é então girada e um indivíduo é selecionado, repetindo-se esse processo até que 100 pais sejam selecionados, podendo um mesmo indivíduo ser selecionado mais de um vez (RAZALI; GERAGHTY et al., 2011; KUMAR, 2012).

Após a seleção dos indivíduos pais é aplicado o operador de cruzamento. Para isso utilizou-se o método de corte por um ponto, cujo seu comportamento já foi descrito na Seção 5.1.4. A cada operação de cruzamento um novo ponto de corte é defino aleatoriamente obedecendo valores entre 1 e $l - 1$, sendo $l$ o número total de características.

Após terem sido gerados os indivíduos filhos, a estes são então aplicados o operador de mutação. O método de mutação utilizado é conhecido por mutação bit a bit, ele permite que cada atributo ou gene do cromossomo possa ter o seu valor alterado com base em uma taxa de mutação, que na prática representa a probabilidade de um gene ser alterado ou não. Como a representação do cromossomo nesse trabalho é binária então o valor de cada gene pode ser alterado de 0 para 1 ou então o inverso. A taxa de mutação adotada no método proposto foi de 0,01, ou seja, cada gene tem a probabilidade de 1% de ter o seu valor alterado.

Feito isso, toda a população, tantos os cromossomos pais quanto os filhos, segue para a etapa de seleção de sobreviventes, sendo aqui adotada a abordagem *Steady-State* (mais detalhes sobre essa abordagem encontra-se na Seção 5.1.6). Para isso, inicialmente foi aplicado um operador de *rank* como forma a se garantir que o melhor cromossomo da população seguisse para a próxima geração. Já para a seleção dos demais sobreviventes - 99 indivíduos/cromossomos - foi utilizado o método de torneio.

O Algoritmo 2 descreve o funcionamento da seleção por torneio. Nesse operador várias iterações, ou torneios, são executados até que uma quantidade ($\lambda$) de indivíduos sejam selecionados. Em cada torneio uma quantidade $k$ de cromossomos são escolhidos aleatoriamente da população, onde são então avaliados e o melhor indivíduo ($K_0$) é selecionado. No presente trabalho foi escolhido um valor de $k = 2$.

**Algorithm 2** Seleção por torneio

1. $quantidadeSelecionada = 0$
   $individuosSelecionados = []$
   **while** $quantidadeSelecionados < \lambda$ **do**
       Escolher aleatoriamente $k$ indivíduos
       Comparar os $k$ indivíduos e selecionar o melhor elemento ($k_0$)
       $individuosSelecionados[quantidadeSelecionada] = k_0$
       $quantidadeSelecionada = quantidadeSelecionada + 1$
   **end while**
   A saída do algoritmo é a variável *individuosSelecionados*

O indivíduo ganhador pode ou não retornar à população do torneio, possibilitando que um cromossomo seja selecionado mais de uma vez (EIBEN; SMITH, 2003).

Os passos de execução do GA são repetidos consecutivamente até que uma condição de parada seja alcançada. Como critério de parada, assim como no PSO aqui proposto, foi utilizada a quantidade máxima de 100 iterações, sendo essa condição escolhida pelos motivos já descritos na Seção 6.2.

Os operadores empregados no algoritmo genético aqui proposto - bem como seus respectivos parâmetros - foram adotados com base nos resultados de alguns experimentos empíricos não exaustivos e também baseando-se em trabalhos disponíveis na literatura que qualificavam positivamente os respectivos operadores.

## 6.4 Avaliação das características

O objetivo da seleção de características é encontrar um subconjunto a partir do conjunto original de características que consiga atingir melhores taxas de acerto (CAVALCANTI, 2005), para isso é necessário se utilizar uma função de *fitness*, também chamada de função de aptidão, que permita mensurar a qualidade de cada subconjunto de características ao longo do processo de seleção de características. Aqui vários métodos de classificação foram utilizados como função de avaliação de aptidão, permitindo assim que os métodos de seleção de características aqui propostos fossem avaliados sob diferentes avaliadores de aptidão.

Os classificadores utilizados nesse trabalho foram: *naive bayes*, redes bayesianas, árvore de decisão (C4.5), *random forest*, k-NN, SVM (*Support Vector Machine*) e *multilayer perceptron* (MLP). Mais detalhes sobre o funcionamento desses classificadores podem ser

vistos no Capítulo 3.

Esses métodos de classificação foram escolhidos baseando-se no bom desempenho deles quando empregados ao *keystroke dynamics*, como mostram alguns trabalhos disponíveis na literatura. Além disso esse mesmo conjunto de classificadores foi utilizado por (ANTAL; SZABÓ; LÁSZLÓ, 2015) para avaliação da base de dados por ele proposta (base que também foi utilizada neste trabalho, conforme descrito na Seção 2.3).

Como ferramenta de apoio foi utilizado o Weka. Essa é uma suíte *open source* que contém vários algoritmos de aprendizagem de máquina implementados, dentre eles ferramentas de classificação, regressão, *clustering* e outros mais[2]. Assim, todos os classificadores utilizados nesse trabalho encontram-se disponíveis no Weka.

Uma das vantagens em se utilizar uma ferramenta como esta é poupar o esforço de reimplementar algoritmos já conhecidos e que, em alguns casos, são de difícil desenvolvimento. Um outro ponto positivo é que o Weka, além de já disponibilizar configurações padrões para muitos algoritmos, também permite que os parâmetros dos métodos sejam facilmente ajustados, além de ser um software comumente adotado em vários trabalhos (BARTLOW; CUKIC, 2006; ANTAL; SZABÓ; LÁSZLÓ, 2015; KOŁAKOWSKA, 2018).

[2] https://www.cs.waikato.ac.nz/ml/weka/

# 7 Experimentos e resultados

Seguindo a metodologia descrita anteriormente foi possível se realizar vários experimentos, de tal forma que cada método de seleção aqui proposto (PSO e GA) pudesse ser executado em conjunto com um determinado classificador, possibilitando assim a análise do desempenho dos seletores de características de acordo com cada método de classificação.

Como já mencionado na Seção 5.2, para a execução do PSO (padrão) é necessário se definir os parâmetros de peso de inércia ($w$) e os coeficientes de aceleração - coeficiente cognitivo ($c_1$) e coeficiente social ($c_2$) - que são utilizados na atualização das velocidades das partículas. Vários estudos mostram que não existe um conjunto de valores para esses parâmetros que seja a melhor opção para todas as aplicações do PSO, porém sabe-se que alguns intervalos de valores podem proporcionar melhores resultados ao PSO e que valores mais baixos ou mais altos podem induzir uma busca por explotação e exploração, respectivamente (ABIDO, 2002; LEE et al., 2008; ZHANG; WANG; JI, 2015). Desta forma através de análises empíricas e tomando base alguns valores de parâmetros já consolidados na literatura, para o PSO aqui proposto utilizou-se o valor de inércia igual a 0,7 e os coeficiente de aceleração (tanto $c_1$ quanto $c_2$) igual a 2, onde esses valores são constantes, ou seja, não variam ao longo das iterações do PSO.

Ainda com relação ao PSO, foi criado um enxame contendo 100 partículas. Já o critério de parada escolhido é baseado na quantidade máxima de iterações, sendo adotado o limite máximo de 100 repetições. Esse valor foi escolhido ao se considerar o alto custo computacional de alguns métodos de avaliação (classificadores) aqui utilizados e também por se perceber, através de análises empíricas, a ausência de melhorias significativas após o limite estabelecido.

Para o algoritmo genético proposto, de forma análoga ao PSO, foi utilizado uma população de 100 indivíduos e como critério de parada também foi utilizada a quantidade máxima de 100 iterações, pelos motivos descritos acima.

Além dos ajustes nos parâmetros dos seletores é necessário também especificar algumas configurações dos classificadores. Assim como no trabalho de (ANTAL; SZABÓ; LÁSZLÓ, 2015), para alguns dos classificadores aqui utilizados foram adotadas as configu-

rações padrão disponibilizadas pela ferramenta Weka.

Uma visão geral dos parâmetros utilizados encontram-se disponíveis na Tabela 2. Nela são apresentados os parâmetros dos classificadores que tiveram os seus valores ajustados e aqueles onde se mantiveram o padrão do Weka. Assim como para os métodos de seleção de características aqui propostos, os ajustes das configurações dos classificadores também foram feitos a partir de experimentos empíricos não exaustivos, tomando-se como base valores já consolidados na literatura. As configurações padrão foram mantidas nos casos em que os resultados dos experimentos empíricos para ajuste dos parâmetros não conseguiram proporcionar melhorias significativas à classificação.

Tabela 2 – Relação de parâmetros dos classificadores

| **Classificadores** | **Método proposto** | **Antal** (ANTAL; SZABÓ; LÁSZLÓ, 2015) |
|---|---|---|
| Bayes Net | (weka) | (weka) |
| C4.5 | confidenceFactor = 0,2 | |
| | minNumObj = 4 | |
| k-NN | (weka) | (weka) |
| MLP | learningRate = 0,05 | (weka) |
| | momentum = 0,3 | |
| | validationThreshold = 20 | |
| | validationSetSize = 30 | |
| | trainingTime = 5000 | |
| *Naive* Bayes | (weka) | (weka) |
| SVM | cost = 7,46 | |
| | gamma = 0,25 | |
| | normalize = true | |
| Random Forest | (weka) | (weka) |

Para o classificador *bayes net* foi mantido os valores padrões dos parâmetros do Weka, assim como no trabalho de (ANTAL; SZABÓ; LÁSZLÓ, 2015).

Já para o classificador de árvore de decisão C4.5 - denominado no Weka de J48 - adotou-se os valores dos parâmetros utilizados em (ANTAL; SZABÓ; LÁSZLÓ, 2015), onde o parâmetro $confidenceFactor$ é igual a 0,2 - fator que está diretamente relacionado ao processo de pós poda, de tal forma que quanto menor o fator de confiança (*confidence factor*) maior será a quantidade de avaliações pós poda (ALI, 2009) - e o parâmetro $minNumObj$, que indicada o número mínimo de instâncias por nós folha, foi igual a 4.

Para os demais parâmetros manteve-se o padrão do Weka.

Para o classificador k-NN (conhecido no Weka como IBk) foram mantidas as configurações padrão do Weka, considerando-se, portanto, $k$ igual a 1.

A partir de análises empíricas alguns parâmetros também foram ajustados para o MLP. As configurações utilizadas foram: a taxa de aprendizagem do algoritmo *backpropagation* ($learningRate$) igual a 0,05; *momentum* com o valor de 0,3; limiar de validação ($validationThreshold$) igual a 20 (número de erros consecutivos permitidos ao longo da fase de teste); tamanho do conjunto de dados para validação ($validationSetSize$) igual a 30, ou seja, 30% dos dados destinado ao treinamento da rede são reservados à validação do modelo; e o número de épocas de treinamento da rede igual a 5000. Os demais parâmetros do classificador MLP foram mantidos conforme o padrão do Weka.

Para o classificador *naive bayes* foram mantidas as configurações padrão do Weka.

Os valores dos parâmetros do SVM foram os mesmos utilizados por (ANTAL; SZABÓ; LÁSZLÓ, 2015), foram eles: custo (*cost*) igual a 7,46; *gamma* recebeu o valor de 0,25; e por fim foi habilitada a normalização dos dados. As demais configurações permaneceram conforme padrão do Weka, sendo por tanto utilizada a *Radial Basis Function - RBF* como função de kernel.

Por fim tem-se o classificador *random forest* a qual também foram mantidas as configurações padrão do Weka.

Todos os classificadores foram executados utilizando o método de validação cruzada (*cross validation*) com 10 *folds*. O *cross validation* é um método estatístico de avaliação que permite que um conjunto de dados seja divido em duas partes, uma utilizada para o treinamento dos algoritmos de aprendizagem e a outra parte sendo empregada nos testes. Esse método possui ainda subcategorias que especificam mais detalhadamente o processo de divisão e avaliação dos dados, e um deles é *k-folds*, abordagem que foi adotada neste trabalho. No *k-folds* particiona-se o conjunto de dados em $k$ segmentos de tamanhos iguais e em seguida se executa $k$ iterações de classificação, de tal forma que $k-1$ *folds* dos dados são utilizados no treinamento e o *fold* restante é utilizado para teste. Após o término das iterações calcula-se o valor médio das métricas obtidas a partir das classificações (taxa de acertos, taxa de falso positivo, taxa de falso negativo, etc). Esse tipo de avaliação proporciona uma generalização dos algoritmos de aprendizado, a fim de

se evitar a especialização em apenas uma porção dos dados (REFAEILZADEH; TANG; LIU, 2009; ZHANG et al., 2014).

Sabendo-se que são 2 os métodos propostos de seleção de características e 7 o número de classificadores utilizados, no total obteve-se 14 combinações de algoritmos para otimização do *keystroke dynamics*, sendo que cada uma dessas combinações foram executadas repetidamente por 30 vezes e assim obtendo os dados necessários para análise estatística do desempenho das soluções propostas. Ao final um total de 420 experimentos foram realizados.

Na Tabela 3 é apresentado um comparativo dos níveis de acuracidade das classificações ao se empregar os métodos de seleção de características propostos nesse trabalho, sendo mostrado também o desempenho das classificações sem o processo de seleção de características. Os dados contidos na referida tabela correspondem aos valores médios das taxas de acertos das classificações seguidos dos seus respectivos desvios padrões, entre parênteses.

Tabela 3 – Taxas de acertos das classificações com os métodos de seleção de características e sem seleção de características (%)

| **Classificadores** | **GA** | **PSO** | **Sem seleção de características** |
|---|---|---|---|
| Bayes Net | 91,82 (0,23) | 92,35 (0,35) | 91,64 (2,04) |
| J48 | 72,72 (0,47) | 73,29 (0,75) | 69,42 (3,35) |
| KNN | 76,08 (0,39) | 76,4 (0,85) | 72,85 (2,85) |
| MLP | 88,21 (1,88) | 88,77 (1,87) | 83,88 (2,56) |
| Naive Bayes | 83,52 (0,37) | 83,97 (0,61) | 78,93 (2,67) |
| SVM | 87,85 (0,31) | 88,46 (0,4) | 87,26 (2,1) |
| Random Forest | 93,57 (0,17) | 94,12 (0,17) | 92,9 (1,7) |

Analisando a acurácia do classificador de redes bayesianas (*bayes net*) percebe-se que os melhores resultados foram alcançados ao se utilizar a seleção de características por PSO, na qual foi possível se obter uma taxa de acertos de 92,35% com desvio padrão de 0,35%. Já a classificação sem a seleção de características demonstrou um desempenho de 91,94% de acertos com dispersão de 1,73%. Por fim tem-se a classificação com seleção de características por GA, que alcançou taxa de 91,82% e taxa de dispersão de 0,23%, sendo esse o menor valor médio obtido nesse método de classificação. Neste caso a classificação sem seleção de características apresenta uma ligeira superioridade em termos percentuais

de acerto em relação à seleção com GA, em contrapartida as classificações com o referido método de seleção proporciona resultados mais estáveis pois sua taxa de desvio padrão é bastante inferior às demais.

Para o classificador C4.5 novamente a aplicação do PSO proporcionou melhores resultados, alcançando assim uma taxa de acerto de 73,29% com desvio padrão de 0,35%, seguido da seleção por GA com 72,72% de acertos e dispersão de 0,47%. Já as classificações sem seleção de características foram as que apresentaram uma menor acurácia, obtendo uma taxa de acerto 72,98% e desvio padrão de 2,25%.

Combinando-se a seleção de características por PSO com o classificador k-NN foi possível de obter uma taxa de acertos de 76,4% e desvio padrão de 0,85%, enquanto que a seleção por GA alcançou um resultado ligeiramente inferior de 76,08% e um nível de dispersão de 0,39%. A classificação sem seleção de característica novamente foi a que apresentou menor acuracidade, ficando com 72,98% de acertos e desvio padrão de 2,25%. Aqui nota-se que as taxas de acerto das classificações aplicando-se o PSO e o GA são bastante próximas, com o PSO proporcionando uma maior valor médio de acuracidade enquanto que o GA garante uma ligeira melhoria na estabilidade dos resultados. Já as classificações sem seleção características além de apresentarem inferioridade na acuracidade de aproximadamente 3% em relação aos outros resultados, também apresentam uma maior variação em sua classificação.

A classificação por MLP quando associada à seleção de características por PSO obteve taxa de acerto de 88,77% e desvio de padrão de 1,87%, sendo esse o seu melhor valor médio de acuracidade, enquanto que a seleção por GA alcançou uma taxa de 88,21% e desvio padrão de 1,88%. Já as classificações sem a aplicação de seleção de características novamente resultaram em níveis mais baixos de acuracidade, apresentando 86,26% de acertos e desvio padrão de 2,19%. Os resultados obtidos através das classificações com MLP, em comparação ao desempenho dos outros classificadores aqui utilizados, foram os que desempenharam maior valor de dispersão, porém ainda assim os seletores de características conseguiram proporcionar uma redução dessas variações.

Na classificação por *naive bayes* novamente os métodos de seleção proporcionaram melhores resultados ficando o PSO com 83,97% de acertos e 1,87% de dispersão e o GA alcançou 83,52% de acuracidade e desvio padrão de 0,37%, enquanto que na ausência de

seleção de características as classificações obtiveram um valor médio de 78,93% de acertos e dispersão de 2,63%.

A seleção de característica por PSO quando aplicada ao classificador SVM também apresentou melhores resultados em comparação ao GA e a classificação sem escolha de características, alcançando uma acuracidade de 88,46% e um baixo nível de dispersão de 0,4%. Em seguida o segundo melhor resultado foi das classificações sem seleção de características com 88,33% de acertos e 1,87% de desvio padrão. Por fim teve-se a seleção por GA com um resultado um pouco mais baixo, obtendo taxa de acertos de 87,85% e desvio padrão de 0,31%. Assim como nos experimentos com *bayes net*, a classificação por SVM sem seleção de características desempenhou um superioridade na taxa acerto em relação à seleção com GA, contudo as classificações com a seleção de característica por algoritmo genético apresentaram resultados mais estáveis.

A classificação por *random forest* quando associada à seleção por PSO proporcionou um dos melhores resultados ao *keystroke dynamics* alcançando taxa de acerto de 94,12% e um nível de dispersão muito baixo de 0,17%. Esse mesmo patamar de acuracidade também foi percebido ao se utilizador a seleção por GA que obteve 93,57% de acerto com um mesmo desvio padrão do PSO, ou seja, 0,17%. Já a classificação sem seleção de características obteve o menor resultado, sendo a taxa de acerto de 93,04% e nível de dispersão de 1,65%.

Na Figura 12 é apresentado graficamente os valores médios das taxas de acertos das classificações com e sem aplicação dos métodos de seleção de características sendo apresentado ainda os seus respectivos intervalos de confiança a um nível de 95%.

O cálculo do intervalo de confiança das taxas de acerto foi feito de acordo com a Equação 7.1, onde $\bar{X}$ e $\sigma$ são respectivamente o valor médio e desvio padrão das taxas e $n$ a quantidade de amostras, que aqui foi igual a 30 (número de repetição dos experimentos), já o número 1,96 corresponde ao valor na tabela de distribuição normal (*Z table*) para um nível de confiança de 95%.

$$\bar{X} \pm 1,96 \frac{\sigma}{\sqrt{n}} \tag{7.1}$$

Ainda analisando a Figura 12 é possível observar facilmente que os resultados das classificações com J48, k-NN, MLP e *naive bayes* que utilizaram os métodos de seleção aqui propostos superam os resultados das classificações sem seleção de características.

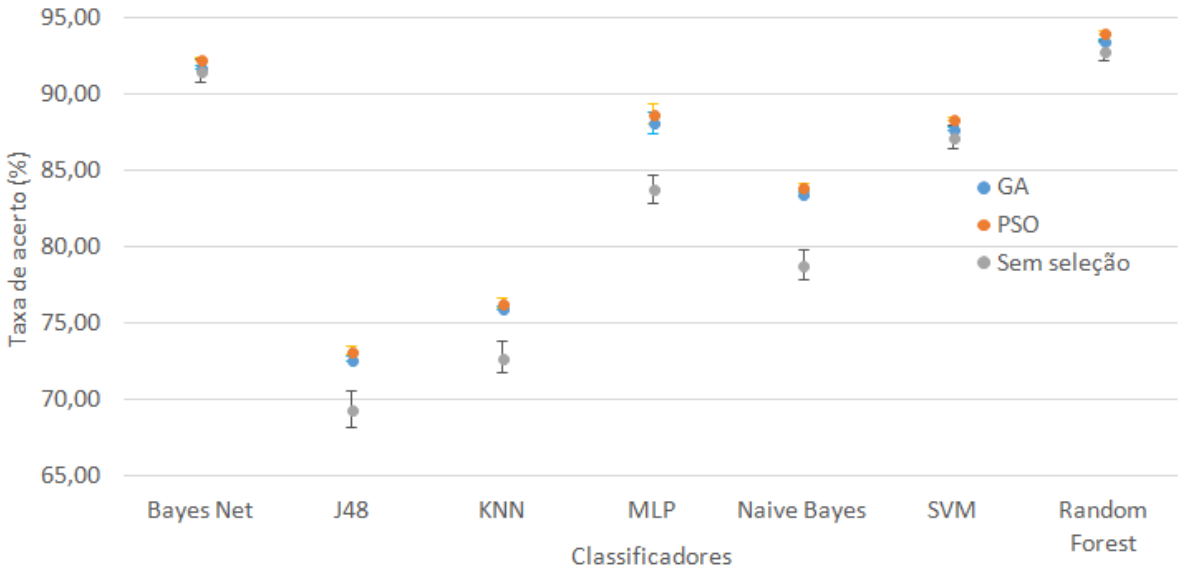


Figura 12 – Visão geral das taxas médias de acerto e seus intervalos de confiança

Porém, para os demais classificadores (*bayes net*, SVM e *random forest*), apesar de o gráfico apontar para uma rápida melhoria nos resultados das classificações com seleção de características, é difícil afirmar apenas com uma análise visual se os valores médios das taxas de acerto são superiores às classificações sem seleção de característica.

Então um teste de hipótese foi aplicado a fim de verificar se existem diferenças estatisticamente significativas entre os resultados (taxas médias de acerto) das classificações com os métodos de seleção de características em relação às classificações sem seleção de características. Anteriormente à aplicação do teste de hipótese foi executado o teste de *Fisher* (ARANGO, 2005) a um nível de significância de 5% ($\alpha = 0,05$) a fim de verificar a similaridade das variâncias dos dados e ao final foi identificado que o dados analisados são homocedásticos, ou seja, possuem variâncias estatisticamente semelhantes. Após esse análise prévia foi aplicado o teste de hipótese (teste t-Student) a um nível de significância de 5% ($\alpha = 0,05$) considerando os dados como sendo independentes e homocedásticos, tomando como hipótese nula ($H_0$) a equivalência estatística dos dados, ou seja, as taxas de acertos não possuem diferenças significativas, enquanto que como hipótese alternativa ($H_1$), de forma análoga, é assumido que existem diferenças significativas entre as taxas de acerto.

Na Tabela 4 são apresentados os resultados dos testes de hipótese. Nela podemos perceber que nos testes de hipótese para o classificador *bayes net* tanto utilizando GA

quanto PSO a hipótese nula ($H_0$) foi aceita, ou seja, as classificações com GA ou PSO desempenharam taxas de acertos estatisticamente equivalentes às classificações sem seleção de característica. Enquanto que para os testes aplicados aos resultados dos classificadores J48, k-NN, MLP, *naive bayes* e *random forest*, todos eles tiveram a hipótese nula rejeitada, ou seja, as taxas de acertos das classificações com seleção de características apresentaram diferenças significativas em relação às classificações sem seleção. Por fim, analisando às classificações com SVM, a hipótese nula apenas foi aceita nos testes com os resultados do GA, enquanto que para o PSO a hipótese nula foi rejeitada.

Tabela 4 – Teste de hipótese de equivalência estatística entre as taxas de acertos das classificações com seleção de características em relação às classificações sem seleção

| **Classificador** | **Método de seleção** | |
|---|---|---|
| | **GA** | **PSO** |
| Bayes Net | Aceita $H_0$ | Aceita $H_0$ |
| J48 | Rejeita $H_0$ | Rejeita $H_0$ |
| KNN | Rejeita $H_0$ | Rejeita $H_0$ |
| MLP | Rejeita $H_0$ | Rejeita $H_0$ |
| Naive Bayes | Rejeita $H_0$ | Rejeita $H_0$ |
| SVM | Aceita $H_0$ | Rejeita $H_0$ |
| Random Forest | Rejeita $H_0$ | Rejeita $H_0$ |

Desta forma podemos inferir que em alguns casos os resultados das classificações com seleção de características são estatisticamente equivalentes às classificações sem seleção, mais precisamente as classificações com *bayes net* (GA e PSO) e as classificações com SVM e GA. Nos demais casos os resultados são estatisticamente diferentes.

Também foram aplicados testes de hipóteses a fim de se verificar o nível de similaridade entre os resultados das classificações com GA e PSO. Na Figura 5 são presentados os resultados dos referidos testes de hipótese, que foram realizados seguindo o mesmo nível de significância e hipóteses (nula e alternativa) descritas anteriormente.

Observando a Figura 5 é possível observar que nos testes com as taxas de acerto das classificações com k-NN e MLP a hipótese nula foi aceita, ou seja, os resultados das classificações com GA e PSO não apresentaram diferenças significativas, enquanto que para os demais classificadores (*bayes net*, J48, *naive bayes*, SVM e *random forest*) a hipótese numa foi rejeitada, ou seja, as taxas apresentam diferenças significativas.

Tabela 5 – Teste de hipótese de equivalência estatística entre as taxas de acertos das classificações com seleção de características por GA em relação às seleções por PSO

| **Classificador** | **Teste t-Student** |
|---|---|
| Bayes Net | Rejeita $H_0$ |
| J48 | Rejeita $H_0$ |
| KNN | Aceita $H_0$ |
| MLP | Aceita $H_0$ |
| Naive Bayes | Rejeita $H_0$ |
| SVM | Rejeita $H_0$ |
| Random Forest | Rejeita $H_0$ |

Desta forma podemos afirmar que apesar de que todas as taxas de acertos das classificações com PSO terem sidos superiores às classificações com GA, os resultados das classificações com k-NN e MLP foram estatisticamente equivalentes.

Até aqui a avaliação de desempenho do *keystroke dynamics* foi feita tomando-se como base as taxas de acerto das classificações do dados dos usuários, porém, conforme descrito anteriormente na Seção 2.2, é possível avaliar o *keystroke dynamics* de diversas formas, uma deles é através das taxas de erro geradas a partir da classificação dos dados.

Na Tabela 6 são mostradas as taxas de erro de falso positivo (FAR) e falso negativo (FRR) das classificações bem como seus respectivos desvios padrões.

Tabela 6 – Taxas de erro das classificações (%)

| **Classificador** | **GA** | | **PSO** | | **Sem seleção** | |
|---|---|---|---|---|---|---|
| | FAR | FRR | FAR | FRR | FAR | FRR |
| Bayes Net | 0,2 (0,01) | 8,18 (0,23) | 0,19 (0,01) | 7,65 (0,35) | 0,2 (0,05) | 8,11 (2,22) |
| J48 | 0,67 (0,01) | 27,28 (0,47) | 0,65 (0,02) | 26,71 (0,75) | 0,75 (0,08) | 30,19 (3,62) |
| KNN | 0,58 (0,01) | 23,92 (0,39) | 0,58 (0,02) | 23,6 (0,85) | 0,66 (0,07) | 26,98 (2,63) |
| MLP | 0,29 (0,05) | 11,32 (2,38) | 0,27 (0,05) | 11,13 (1,95) | 0,39 (0,06) | 16,13 (2,67) |
| Naive Bayes | 0,4 (0,01) | 16,48 (0,37) | 0,39 (0,01) | 16,03 (0,61) | 0,51 (0,06) | 21,07 (2,44) |
| SVM | 0,3 (0,01) | 12,15 (0,31) | 0,28 (0,01) | 11,54 (0,4) | 0,31 (0,05) | 12,21 (1,96) |
| Random Forest | 0,16 (0) | 6,43 (0,17) | 0,14 (0) | 5,88 (0,17) | 0,17 (0,04) | 6,58 (1,6) |

Na classificação por *bayes net* o PSO obteve o erro médio de FAR de 0,19% e desvio padrão de 0,01%, enquanto que as taxas de FRR foram de 7,65% e desvio padrão de 0,35%. Já a seleção por GA teve uma taxa 0,2% de falsos positivos (FAR) e 0,01%

de desvio padrão, e 8,18% de FRR com desvio padrão de 0,23%. Nas classificações sem seleção o FAR foi de 0,2% com desvio padrão de 0,05% enquanto que o FRR foi de 8,11% e desvio de 2,22%.

Já o classificador de árvore de decisão (C4.5) gerou uma taxa de erro um pouco mais elevada. Quando utilizado juntamente com o algoritmo de seleção por PSO alcançou o valor de 0,65% para FAR e desvio padrão de 0,02%, e valor médio de FRR de 26,71% e desvio padrão de 0,75%. O mesmo classificador quando utilizado em conjunto com o GA apresentou um erro médio de falso positivo de 0,67% e nível de dispersão de 0,01%, e também 27,28% de FRR e desvio padrão de 0,47%. Por fim as classificações sem seleção de características tiveram FAR de 0,75% com desvio padrão de 0,08% e FRR igual 30,19% e desvio padrão de 3,62%. Com esses resultados o C4.5, quando comparado com os demais métodos de classificação, foi o classificador que gerou as maiores taxas de erro.

Nas classificações com o k-NN as taxas de FAR foram as mesmas tanto ao se utilizar a seleção por PSO quanto por GA, alcançando-se o valor de 0,58%, os desvios padrões também foram bastante próximos sendo um pouco maior no PSO com 0,02% e o GA um pouco mais estável alcançou 0,01% de variação. O FRR da classificação com o PSO foi de 23,6% e dispersão de 0,85%, já GA ficou com taxa de 23,92% de FRR e taxa de dispersão de 0,39%. Já as classificações sem seleção obtiveram valores de FAR de 0,66% e dispersão de 0,07% e FRR igual a 26,98% e desvio padrão de 2,63%.

O MLP por sua vez quando executado em conjunto com o seletor de características PSO teve FAR igual a 0,27% e desvio padrão de 0,05% e FRR de 11,13% com desvio padrão de 1,95%. O GA obteve 0,29% de falsos positivos e desvio padrão de 0,05% além de 11,32% de FRR com dispersão de 2,38%. As classificações sem aplicação dos métodos de seleção desempenham valores de FAR igual a 0,39% e desvio padrão de 0,06% apresentando ainda FRR de 16,13% com um desvio padrão de 2,67%.

O classificador *naive bayes* quando utilizado em conjunto com o PSO gerou um erro FAR de 0,39% e desvio padrão de 0,01%, já as taxas de FRR foi de 16,03% e desvio padrão de 0,61%. O mesmo classificador quando utilizado em conjunto com a seleção de características por algoritmo genético obteve FAR de 0,4% e nível de dispersão de 0,01% e FRR de 16,48% com desvio padrão de 0,37%. Por fim das classificações sem seleção o FAR foi de 0,51% com desvio de 0,06% e FRR igual a 21,07% e desvio padrão de 2,44%.

A classificação por SVM proporcionou um erro menor que o *naive bayes*, resultando nos experimentos com PSO um FAR de 0,28% e desvio padrão de 0,01%, e as taxas de FRR de 11,54% com a dispersão de 0,4%. Enquanto que ao se empregar o GA teve-se uma taxa de falsos positivos de 0,3% e desvio padrão de 0,01%, já os níveis de FRR foi de 12,15% com valores variando a um desvio padrão de 0,31%. Nos resultados das classificações sem seleção observou-se os valores FAR igual a 0,31% com desvio padrão de 0,05% e taxa de FRR de 12,21% e dispersão de 1,96%.

Por fim o *random forest*, novamente alcançando os melhores resultados, obteve uma taxa de falsos positivos de 0,14% e desvio padrão de 0% para o PSO além de obter taxas de 5,88% de FRR com dispersão de 0,17%. Para os experimentos com o algoritmo genético foi possível se alcançar taxas de 0,16% de FRA e nível de dispersão de 0%, além de ser obtido FRR de 6,43% e desvio padrão de 0,17%. Quando aos experimentos sem seleção de características o valor de FAR foi de 0,17% com taxa de dispersão de 0,04% e FRR igual a 6,58% e desvio padrão de 1,6%.

De forma geral as taxas de FAR foram baixas, todas inferiores a 1%, isso significa que em uma situação prática do *keystroke dynamics* as chances de um usuário impostos violar a autenticação de um foram baixas. Enquanto que as taxas FRR foram um pouco mais elevadas, com valores variando entre 6,43% e 28%.

Além disso, todas as classificações que utilizaram o PSO como seletor de característica conseguiram alcançar taxas de erro menores ou iguais às classificações com seleção de características por GA. Porém, salvo às classificação por MLP, todos os desvios padrões dos erros foram menos nas classificações que empregaram o GA como seletor de *feature*.

Como a seleção de características tem como objetivo escolher as *features* mais relevantes, consequentemente, espera-se que após a execução de um método dessa natureza se tenha a redução do vetor de características. Desta forma na Tabela 7 são apresentados os valores médios das taxas de redução de características em níveis percentuais proporcionadas pelo PSO e pelo GA, lembrando-se que originalmente a base de dados contém 71 características.

Nas classificações com redes bayesianas (*bayes net*) a seleção de característica por PSO conseguiu uma redução de 22,03% das características como desvio padrão de 3,94%. No caso do GA as características foram minimizadas em 28,17% e desvio padrão de 3,95%.

Tabela 7 – Taxas de redução de características (%)

| **Classificador** | **GA** | **PSO** |
|---|---|---|
| Bayes Net | 28,17 (3,95) | 22,03 (3,94) |
| C4.5 | 50 (4,57) | 49,03 (6,46) |
| k-NN | 41,64 (4,89) | 35,53 (7,96) |
| MLP | 38,03 (6,12) | 35,03 (9,03) |
| *Naive* Bayes | 44,32 (4,46) | 38,23 (4,62) |
| SVM | 26,38 (4,7) | 19,93 (4,1) |
| Random Forest | 37,42 (4,77) | 29,56 (4,2) |

Desta forma percebe-se que o GA conseguiu uma maior taxa de redução de características do que o PSO, com valores de desvio padrões semelhante.

Já quando utilizado em conjunto com o classificador C4.5 os seletores de características alcançaram os seus mais altos níveis de redução de *features*. O PSO apresentou uma minimização de 49,03% das características com desvio padrão de 6,46%, enquanto que o GA demonstrou uma pequena superioridade na redução de características ficando com taxa de 50% e com um desvio padrão de 4,57%. Nesse caso a seleção por GA além de conseguir maior redução do vetor de características, também conseguiu taxas de minimização de *features* mais estáveis.

As seleções de característica quando associadas ao classificador k-vizinhos mais próximos, ou k-NN, também geraram níveis elevados de redução de característica, porém um pouco menor que as classificações com o C4.5. Ao final dos experimentos com seleção por PSO as características foram reduzidas em média 35,53% com desvio padrão de 7,96%. Aqui novamente o GA superou o PSO tanto pela taxa de redução de características quanto pelo nível de variação dos resultados, alcançando redução de 41,64% e desvio padrão 4,89%.

Nos experimentos com o classificador MLP as seleções de características também alcançaram uma boa taxa de redução. Com o PSO a minimização ocorreu a uma taxa média de 35,03% com variação de 9,03%, enquanto que na seleção com algoritmo genético as valores foram de 38,03% e 6,12% para taxa de redução e desvio padrão, respectivamente. Aqui o GA conseguiu uma maior redução nos vetores de características e também uma maior homogeneidade no quantitativo das *features*.

Nos experimentos com *nayve bayes* o PSO reduziu em 38,23% as características a uma variação média de 4,62% e quando aplicado o método de seleção por GA a taxa de minimização se elevou para 44,32% e o desvio padrão médio dos resultados foi diminuído para 4,46%.

Já nas classificações com SVM o PSO minimizou os atributos do *keystroke dynamics* em 19,93% com uma flutuação de 4,1% e o GA novamente alcançou um nível superior de redução com 26,38%, porém, nesse caso, o desvio padrão foi de 4,7%, ou seja, a sua variação foi ligeiramente maior que a do PSO.

Por fim é apresentada a análise da redução das características com o classificador *random forest*. Aqui o PSO reduziu em 29,56% as características a uma variação média de 4,2% e o GA minimizou o número de atributos para 37,42% do quantitativo inicial com uma taxa de variação de 4,77%.

De forma geral, para todos os classificadores aqui utilizados, observou-se que as seleções de características com o algoritmo genético proporcionaram uma maior taxa de redução de *features* quando comparado com o desempenho do PSO. Quanto aos níveis de estabilidade dos métodos de seleção aqui propostos, ao se considerar a quantidade de características selecionadas, nota-se que em alguns casos a seleções por algoritmo genético foram mais estáveis e em outras situações o PSO demonstrou maior equilíbrio na escolha dos atributos.

A fim de se consolidar o bom desempenho dos métodos de seleção de características aqui desenvolvidos, na Tabela 8 é apresentado um quadro comparativo que traz os resultados de alguns trabalhos onde também foram aplicados métodos de seleção de características ao *keystroke dynamics*. Nessa tabela são disponibilizados alguns dados estatísticos como média, desvio padrão e coeficiente de variação (CV) tanto para os trabalhos disponíveis na literatura (quando possível) quanto para a presente pesquisa.

Tabela 8 – Comparativo dos trabalhos sobre seleção de características no *keystroke dynamics*

| Trabalho | Seletor | Classificador | Taxa de acerto | | | FAR | | | FRR | | | Taxa de redução | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Média | Desvio | CV | Média | Desvio | CV | Média | Desvio | CV | Média | Desvio | CV |
| Trabalho proposto | GA | Bayes *Net* | 91,818 | 0,228 | 0,002 | 0,200 | 0,006 | 0,028 | 8,182 | 0,228 | 0,028 | 28,169 | 3,950 | 0,140 |
| | | J48 | 72,723 | 0,466 | 0,006 | 0,665 | 0,011 | 0,017 | 27,277 | 0,466 | 0,017 | 50,000 | 4,568 | 0,091 |
| | | k-NN | 76,083 | 0,394 | 0,005 | 0,583 | 0,010 | 0,016 | 23,917 | 0,394 | 0,016 | 41,643 | 4,888 | 0,117 |
| | | MLP | 88,209 | 1,880 | 0,021 | 0,287 | 0,046 | 0,160 | 11,320 | 2,377 | 0,210 | 38,028 | 6,118 | 0,161 |
| | | *Naive* Bayes | 83,523 | 0,369 | 0,004 | 0,402 | 0,009 | 0,022 | 16,477 | 0,369 | 0,022 | 44,319 | 4,465 | 0,101 |
| | | SVM | 87,848 | 0,309 | 0,004 | 0,296 | 0,008 | 0,025 | 12,152 | 0,309 | 0,025 | 26,385 | 4,699 | 0,178 |
| | | *Random Forest* | 93,566 | 0,172 | 0,002 | 0,157 | 0,004 | 0,027 | 6,434 | 0,172 | 0,027 | 37,425 | 4,768 | 0,127 |
| | PSO | Bayes *Net* | 92,350 | 0,349 | 0,004 | 0,187 | 0,009 | 0,046 | 7,650 | 0,349 | 0,046 | 22,033 | 3,937 | 0,179 |
| | | J48 | 73,290 | 0,749 | 0,010 | 0,651 | 0,018 | 0,028 | 26,710 | 0,749 | 0,028 | 49,033 | 6,457 | 0,132 |
| | | k-NN | 76,401 | 0,853 | 0,011 | 0,576 | 0,021 | 0,036 | 23,599 | 0,853 | 0,036 | 35,533 | 7,961 | 0,224 |
| | | MLP | 88,769 | 1,871 | 0,021 | 0,274 | 0,046 | 0,166 | 11,134 | 1,952 | 0,175 | 35,034 | 9,034 | 0,258 |
| | | *Naive* Bayes | 83,974 | 0,608 | 0,007 | 0,391 | 0,015 | 0,038 | 16,026 | 0,608 | 0,038 | 38,233 | 4,624 | 0,121 |
| | | SVM | 88,458 | 0,403 | 0,005 | 0,282 | 0,010 | 0,035 | 11,542 | 0,403 | 0,035 | 19,933 | 4,098 | 0,206 |
| | | *Random Forest* | 94,120 | 0,174 | 0,002 | 0,143 | 0,004 | 0,030 | 5,880 | 0,174 | 0,030 | 29,560 | 4,196 | 0,142 |
| [1] | GA | SVM | | | | 0,370 | 0,710 | 1,919 | 2,500 | 3,670 | 1,468 | 52,540 | 2,400 | 0,046 |
| | PSO | SVM | | | | 0,760 | 1,110 | 1,461 | 0,810 | 1,420 | 1,753 | 77,040 | 3,360 | 0,044 |
| [2] | GA | SVM | 98,130 | | | 0,800 | | | 2,930 | | | | | |
| [3] | PSO | SVM | | | | 1,250 | | | 6,250 | | | 95,600 | | |
| [4] | GA | BPNN | 88,600 | | | | | | | | | | | |
| | PSO | BPNN | 94,800 | | | | | | | | | | | |
| | ACO | BPNN | 88,900 | | | | | | | | | | | |
| [5] | IG | J48 | 99,900 | | | 0,100 | | | | | | | | |
| | FS | *Naive* Bayes | 87,300 | | | 12,300 | | | | | | | | |
| [6] | ACO | SVM | | | | 0,245 | | | 38,400 | | | | | |

No trabalho [1] (AZEVEDO; CAVALCANTI; FILHO, 2007) os autores desenvolveram um método de seleção por PSO e outro do tipo GA e os utilizou em conjunto com o classificador SVM. Observando os valores FAR e FRR de (AZEVEDO; CAVALCANTI; FILHO, 2007) é possível perceber que todos eles são maiores do que aqueles obtidos pelo modelo aqui proposto (considerando apenas os resultados de PSO-SVM e GA-SVM), com uma maior discrepância sobretudo nos valores de desvio padrão e coeficiente de variação. Indicando, de acordo essas duas métricas, que os métodos aqui propostos são mais estáveis, pois apresentam valores menores de desvio padrão. Em contrapartida todas medidas de redução de características são melhores nas seleções de (AZEVEDO; CAVALCANTI; FILHO, 2007), enquanto que a sua melhor taxa média foi de 77,04% (PSO), o maior valor obtido nos experimentos aqui realizados (apenas nas classificações com SVM) foi de 26,38% (GA).

No estudo [2] (SUNG; CHO, 2006) também foi proposto um método de seleção GA-SVM (Algoritmo Genético combinado com SVM). Os destaques dos resultados recaem sobretudo para as taxas médias de acuracidade que é de 98,13% e de FRR 2,93%. Porém já a taxa média de FAR é de 0,8% em (SUNG; CHO, 2006), sendo ela maior do que todas aquelas (do mesmo tipo) alcançadas pelos métodos de seleção aqui elaborados.

Já no trabalho [3] (SENATHIPATHI; BATRI, 2014) o que se mais destaca é a capacidade de redução de atributos do método apresentado. Um modelo de seleção baseado em PSO e SVM foi capaz de reduzir em 95,6% o conjunto de *features*, indo de 2550 para 122 características. Além disso o método de (SENATHIPATHI; BATRI, 2014) ainda alcançou a taxa média de FRR de 6,25%, um quantitativo inferior aos alcançados pelos modelos aqui desenvolvidos (PSO-SVM e GA-SVM). Porém a taxa de FAR de 1,25% foi maior que todas aquelas do mesmo tipo observadas no trabalho aqui desenvolvido.

No trabalho [4] (KARNAN; KRISHNARAJ, 2010) foram criados 3 modelos de seleção de características para o *keystroke dynamics*, um construído com base em algoritmo PSO, outro em GA e por fim um método de seleção por ACO, sendo todos eles utilizados em conjunto com o classificador *Back-Propagation Neural Networks* (BPNN) (DUDA; HART; STORK, 2012). A acuracidade alcançada foi de 88,6% para o GA, 94,8% para o PSO e 88,9% para a seleção com ACO. São desempenhos semelhantes aos alcançados por outros trabalhos disponíveis na literatura e que também são compatíveis com o desempenho dos métodos de seleção de características aqui propostos.

Em [5] (SHABTAI et al., 2012) foram propostos alguns métodos de seleção de características do tipo filtro (mais detalhes sobre esse tipo encontram-se na Seção 4), sendo os melhores resultados obtidos por meio da seleção por *Information Gain* (IG) (DASH; LIU, 1997) e *Fisher Score* (FS) (DUDA; HART; STORK, 2012). Quando utilizado em conjunto com o classificador J48 (C4.5) o IG conseguiu taxa de acerto de 99,9% e FAR de 0,01%, enquanto que a seleção por FS quando combinado com o classificador *Naive* Bayes proporcionou uma taxa média de acertos de 87,3%, valor esse maior do que os resultados alcançados pelos métodos propostos no presente trabalho (PSO-*Naive* Bayes e GA-*Naive* Bayes). Porém a combinação FS-*Naive* Bayes obteve taxa de FAR de 12,3%, enquanto que os métodos de seleção desenvolvidos na presente pesquisa alcançaram FAR de 0,391% (PSO) e 0,402% (GA) quando utilizados em conjunto com o classificador *Naive* Bayes.

Por fim no trabalho [6] (ALSULTAN; WARWICK; WEI, 2017) é apresentado uma método de seleção de características do tipo ACO-SVM. Nesse estudo o desempenho da solução é avaliado através das taxas de falso positivo (FAR) e por meio da taxa de redução de características. O valor médio de FAR é de 0,245%, valor rapidamente melhor aos alcançados pelos métodos propostos pelo presente trabalho (considerando apenas as classificações com SVM). Porém o valor de 38,4% de FRR apresentado por (ALSULTAN; WARWICK; WEI, 2017) é demasiadamente alto, sendo a maior taxa de erro apresentada nessa análise comparativa.

Observando a Tabela 8 e considerando a análise comparativa acima descrita é possível perceber que os métodos de seleção de característica aqui desenvolvidos de forma geral apresentaram resultados compatíveis com aqueles já encontrados na literatura, conseguindo alcançar desempenhos equivalentes ou, em alguns casos, até mesmo superiores a outras técnicas já criadas. Um outro ponto que chama bastante atenção nos dados apresentados são os altos níveis de estabilidade dos modelos aqui propostos, ficando evidenciado através dos valores muito baixos de desvio padrão e também coeficiente de variação a confiabilidade dos seletores de atributos concebido pelo presente trabalho.

Nesse trabalho foi possível ainda se analisar a ocorrência das características, ou seja, a frequência com que cada atributo foi ou não selecionado pelos métodos de seleção aqui proposto. Desta forma pode-se identificar as características que mais foram escolhidas, ou seja, aquelas que de forma geral agregam maior acuracidade ao *keystroke dynamics* e também reconhecer as *features* menos selecionadas, ou seja, as características que tendem

a contribuir de forma negativa à classificação. Para isso foi feito um ranqueamento da ocorrência total das *features* a partir dos vetores de características obtidos ao final de cada experimento.

O processo de análise de ocorrência das características foi divido em duas etapas. Na primeira análise foram identificadas as *features* mais selecionadas - de acordo com cada classificador - a partir de uma limiarização em 25% do *rank*, ou seja, 1/4 das características com maiores frequências foram reconhecidas como sendo as mais selecionadas. Analogamente, o mesmo procedimento foi adotado para classificação das *features* como sendo as menos selecionadas. A seguir são apresentados os resultados obtidos a partir dessa investigação.

Na Tabela 9 são aprestadas as características que foram mais selecionadas ao longo das classificações com o PSO. Iniciando a análise pelos experimentos com o *bayes net* percebe-se que a *feature* mais selecionada foi a MFA, estando esta característica presente em todos os resultados da seleção por PSO ao se utilizar o referido classificador. Nesse caso as características mais selecionadas se resumem em 4 do tipo UD, 3 DD, 3 P, 3 do tipo H, além de estarem presentes as características MFA e MH.

Já nas classificações com C4.5 duas características estiveram presentem em todos os resultados, foram as *features* MH e MFA. Além dessas foram ainda reconhecidas como mais selecionadas 5 características DD e 5 UD, 2 do tipo P e mais 2 H, além de estar presente nesta lista de mais selecionadas a característica MP.

Nas classificações com k-NN, 4 características estiveram presentes nos resultados de todos os experimentos, foram $DD_{ti}$, $UD_{ti}$, $UD_{an}$ e $P_p$. Dentre as características mais selecionadas se tem 4 do tipo H, 3 DD, 3 P, 2 do tipo FA, 2 UD, além de estarem nessa relação as 3 características que representam os valores médios de outras *features*, ou seja, MP, MFA e MH.

Nos experimentos com o classificador *nayve bayes* se identificou em todos os resultados da seleção de característica por PSO a presença da *feature* $P_i$. Ao se verificar as características mais selecionadas percebeu-se que 10 são do tipo H, 4 do tipo P, 1 do tipo DD, 1 FA e 1 UD.

Tabela 9 – Lista das características mais selecionadas pelo PSO

| Bayes Net | | C4.5 | | k-NN | | MLP | | *Naive* Bayes | | SVM | | Random Forest | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MFA | 30 | MH | 30 | $DD_{ti}$ | 30 | MFA | 30 | $P_{i}$ | 30 | $P_{R}$ | 30 | MFA | 30 |
| $DD_{nl}$ | 29 | MFA | 30 | $UD_{ti}$ | 30 | $UD_{ti}$ | 27 | $H_{[Shift]}$ | 29 | $FA_{R}$ | 30 | $UD_{nl}$ | 29 |
| $UD_{ie}$ | 29 | $P_{5}$ | 28 | $UD_{an}$ | 30 | $DD_{an}$ | 26 | $H_{o}$ | 28 | $H_{o}$ | 29 | $P_{5}$ | 29 |
| $H_{.}$ | 28 | MP | 25 | $P_{n}$ | 30 | MH | 26 | $H_{a}$ | 28 | $P_{t}$ | 29 | $P_{n}$ | 29 |
| $UD_{ti}$ | 28 | $UD_{5[abc]}$ | 24 | MP | 29 | $H_{[abc]}$ | 25 | $DD_{ti}$ | 28 | $P_{o}$ | 29 | $FA_{.}$ | 29 |
| $P_{i}$ | 28 | $DD_{nl}$ | 23 | MFA | 29 | $H_{o}$ | 25 | $P_{5}$ | 28 | $P_{n}$ | 29 | MH | 29 |
| $FA_{[123?]}$ | 28 | $H_{[abc]}$ | 22 | $DD_{an}$ | 28 | $P_{[abc]}$ | 25 | $H_{.}$ | 27 | $FA_{.}$ | 29 | $H_{.}$ | 27 |
| $FA_{o}$ | 28 | $DD_{.t}$ | 22 | $FA_{.}$ | 28 | $P_{l}$ | 25 | $H_{i}$ | 27 | $FA_{[123?]}$ | 29 | $H_{[abc]}$ | 27 |
| MH | 28 | $UD_{an}$ | 22 | MH | 28 | $FA_{[Shift]}$ | 25 | $H_{e}$ | 27 | MFA | 29 | $DD_{[123?]5}$ | 27 |
| $H_{t}$ | 27 | $DD_{R[Shift]}$ | 21 | $P_{5}$ | 27 | MP | 25 | $H_{l}$ | 27 | $H_{.}$ | 28 | $P_{[abc]}$ | 27 |
| $H_{o}$ | 27 | $UD_{R[Shift]}$ | 21 | $H_{t}$ | 26 | $H_{l}$ | 24 | $P_{.}$ | 27 | $H_{e}$ | 28 | $DD_{[Shift]R}$ | 26 |
| $DD_{ti}$ | 27 | $P_{n}$ | 21 | $H_{o}$ | 26 | $DD_{[Shift]R}$ | 24 | $P_{[abc]}$ | 27 | $H_{5}$ | 28 | $UD_{ie}$ | 26 |
| $DD_{R[Shift]}$ | 27 | $DD_{ti}$ | 20 | $H_{n}$ | 26 | $DD_{nl}$ | 24 | $FA_{[Shift]}$ | 27 | $H_{[abc]}$ | 28 | $UD_{[123?]5}$ | 26 |
| $UD_{[Shift]R}$ | 27 | $UD_{ti}$ | 20 | $FA_{n}$ | 26 | $FA_{o}$ | 24 | $H_{[123?]}$ | 26 | $H_{[Shift]}$ | 28 | $P_{i}$ | 26 |
| $UD_{nl}$ | 27 | $H_{i}$ | 19 | $H_{5}$ | 25 | $P_{a}$ | 23 | $H_{5}$ | 26 | $H_{a}$ | 28 | $P_{R}$ | 26 |
| $P_{[123?]}$ | 27 | $DD_{an}$ | 19 | $DD_{R[Shift]}$ | 25 | $DD_{ie}$ | 22 | $H_{[Shift]}$ | 26 | $DD_{ti}$ | 28 | $DD_{oa}$ | 25 |
| $P_{5}$ | 27 | $UD_{[Shift]R}$ | 19 | $P_{i}$ | 25 | $UD_{ie}$ | 22 | $UD_{ti}$ | 26 | $DD_{an}$ | 28 | $UD_{[Shift]o}$ | 25 |
| $P_{R}$ | 27 | $UD_{nl}$ | 19 | $P_{[abc]}$ | 25 | $UD_{an}$ | 22 | $P_{a}$ | 26 | $UD_{ti}$ | 28 | $FA_{n}$ | 25 |

Em todos os resultados das seleções de características com PSO e SVM encontraram-se a presença das *features* $P_R$ e $FA_R$. Nestes experimentos muitas das melhores características, ou aquelas mais selecionadas, são do tipo H, mais precisamente 2, seguida de 4 características do tipo P, 3 FA, 2 DD e por fim a característica MFA, que apesar que não ser a mais selecionada teve um bom índice de ocorrência aparecendo nos resultados de 29 experimentos.

Nos experimentos com *random forest* e PSO novamente a *feature* MFA reaparece como sendo a mais selecionada, estando contida em todos os vetores de características resultantes do algoritmo de seleção. Enquanto que no restante da lista das características mais selecionadas foram registrados 5 atributos do tipo P, 4 UD, 3 DD, 2 do tipo H, além de estarem presentes as características MH e uma outra do tipo FA ($FA_n$).

Em contrapartida também foi feita uma análise das características que menos foram selecionadas pelo PSO, elas são apresentadas na Tabela 10. Nos experimentos de classificação com *bayes net* e PSO, a *feature* menos selecionada foi a $P_{[Shift]}$ estando contido em apenas 13 vetores de característica finais. Além desse 4 outros atributos do tipo P foram identificados como sendo um dos menos selecionados. Na lista de *features* menos selecionadas também estão registrados 7 características do tipo DD, 2 UD, 1 do tipo H e outra Fa, e por fim a característica MP.

Nos experimentos com C4.5 a característica que menos esteve presente nos resultados finais da seleção foi a $FA_{[123?]}$, com ocorrência igual a 6. Além dessas características também compuseram o *rank* das menos selecionadas 6 atributos do tipo P, 5 outras características FA, 3 H, 2 UD e por fim uma *feature* DD.

As seleções por PSO quando associadas ao classificador k-NN também evitaram as características do tipo FA, sobretudo a $FA_t$ que por sua vez apareceu somente nos resultados de 3 experimentos. Além dela outras 6 características também do tipo FA foram identificadas como sendo uma das menos selecionadas, seguidas de 4 *features* do tipo UD, mais 2 atributos DD, 2 P e uma característica H.

Nos resultados obtidos através de MLP e PSO percebe-se a ausência sobretudo das características $DD_{R[Shift]}$, $FA_t$, $FA_i$, estando essas *features* presentes em apenas 13 dos 30 experimentos. Além delas também foram menos selecionadas 4 características DD, 4 FA, 4 UD, além de 3 *features* do tipo P e outras 2 do tipo H.

Tabela 10 – Lista das características menos selecionadas pelo PSO

| Bayes Net | | C4.5 | | k-NN | | MLP | | *Naive* Bayes | | SVM | | Random Forest | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_{[Shift]}$ | 13 | $FA_{[123?]}$ | 6 | $FA_{t}$ | 3 | $DD_{R[Shift]}$ | 13 | $DD_{.t}$ | 0 | $DD_{[Shift]o}$ | 12 | $DD_{an}$ | 15 |
| $DD_{.t}$ | 15 | $H_{[Shift]}$ | 7 | $FA_{e}$ | 4 | $FA_{t}$ | 13 | $UD_{R[Shift]}$ | 0 | $P_{[Shift]}$ | 12 | $H_{[123?]}$ | 16 |
| $DD_{[Shift]o}$ | 15 | $P_{[123?]}$ | 7 | $DD_{e[123?]}$ | 6 | $FA_{i}$ | 13 | $DD_{[abc][Shift]}$ | 1 | $UD_{[abc][Shift]}$ | 15 | $H_{a}$ | 17 |
| $P_{l}$ | 15 | $DD_{[abc][Shift]}$ | 8 | $FA_{i}$ | 6 | $DD_{e[123?]}$ | 14 | $DD_{[Shift]o}$ | 1 | $DD_{e[123?]}$ | 16 | $UD_{ti}$ | 17 |
| $FA_{a}$ | 15 | $P_{.}$ | 8 | $H_{[Shift]}$ | 7 | $UD_{5[abc]}$ | 14 | $UD_{e[123?]}$ | 1 | $UD_{[Shift]o}$ | 16 | $H_{i}$ | 18 |
| $P_{[Shift]}$ | 16 | $P_{t}$ | 8 | $UD_{e[123?]}$ | 8 | $FA_{e}$ | 14 | $UD_{[Shift]o}$ | 1 | $FA_{[abc]}$ | 16 | $H_{[Shift]}$ | 18 |
| $P_{a}$ | 16 | $P_{R}$ | 8 | $FA_{l}$ | 8 | $DD_{[abc][Shift]}$ | 15 | $DD_{e[123?]}$ | 2 | $H_{[Shift]}$ | 17 | $DD_{ti}$ | 18 |
| $H_{e}$ | 17 | $FA_{o}$ | 8 | $FA_{[Shift]}$ | 9 | $UD_{[abc][Shift]}$ | 15 | $UD_{[abc][Shift]}$ | 3 | $H_{R}$ | 18 | $DD_{e[123?]}$ | 18 |
| $P_{e}$ | 17 | $FA_{a}$ | 8 | $P_{[Shift]}$ | 10 | $UD_{R[Shift]}$ | 15 | $DD_{[123?]5}$ | 4 | $DD_{oa}$ | 18 | $FA_{t}$ | 18 |
| $DD_{e[123?]}$ | 19 | $H_{.}$ | 9 | $P_{l}$ | 10 | $FA_{5}$ | 15 | $UD_{[123?]5}$ | 4 | $UD_{[123?]5}$ | 18 | $H_{t}$ | 19 |
| $DD_{[123?]5}$ | 19 | $P_{[Shift]}$ | 9 | $DD_{nl}$ | 12 | $UD_{e[123?]}$ | 16 | $UD_{.t}$ | 5 | $FA_{i}$ | 18 | $FA_{[123?]}$ | 19 |
| $DD_{oa}$ | 19 | $FA_{n}$ | 9 | $UD_{nl}$ | 12 | $P_{i}$ | 16 | $FA_{l}$ | 5 | $FA_{e}$ | 18 | $FA_{R}$ | 19 |
| $UD_{e[123?]}$ | 19 | $UD_{e[123?]}$ | 10 | $FA_{[abc]}$ | 13 | $P_{[Shift]}$ | 16 | $DD_{R[Shift]}$ | 8 | $FA_{[Shift]}$ | 18 | $FA_{a}$ | 19 |
| MP | 19 | $FA_{5}$ | 10 | $FA_{[Shift]}$ | 13 | $H_{5}$ | 17 | $P_{l}$ | 8 | $DD_{[abc][Shift]}$ | 19 | $FA_{i}$ | 20 |
| $DD_{5[abc]}$ | 20 | $H_{R}$ | 11 | $FA_{5}$ | 14 | $H_{R}$ | 17 | $UD_{nl}$ | 10 | $UD_{e[123?]}$ | 19 | $FA_{5}$ | 20 |
| $DD_{an}$ | 20 | $UD_{[123?]5}$ | 11 | $DD_{[abc][Shift]}$ | 15 | $DD_{[123?]5}$ | 17 | $UD_{ie}$ | 11 | $DD_{.t}$ | 20 | $FA_{[Shift]}$ | 20 |
| $P_{t}$ | 20 | $P_{e}$ | 11 | $UD_{ie}$ | 15 | $P_{t}$ | 17 | MP | 11 | $P_{l}$ | 20 | $FA_{l}$ | 20 |
| $UD_{5[abc]}$ | 21 | $FA_{l}$ | 12 | $UD_{[abc][Shift]}$ | 15 | $FA_{a}$ | 17 | $DD_{5[abc]}$ | 12 | $FA_{5}$ | 20 | $FA_{o}$ | 20 |

Nos resultados das seleções de características com PSO e *nayve bayes* algo que chama atenção é o fato de existiram características com zero ocorrência. As características em questão são do tipo DD e UD, mais precisamente as características $DD_{.t}$ e $UD_{R[Shift]}$. No total 7 *features* DD estiveram entre as menos selecionadas, 7 UD, além de 1 do tipo P ($P_l$), outra do tipo FA ($FA_l$) e também a característica MP.

Já nas seleções de característica por PSO com SVM percebe-se uma ausência sobretudo das *features* $DD_{[Shift]o}$ e $P_{[Shift]}$, cada uma estando presente nos resultados de 7 experimentos. Analisando as demais características menos selecionadas registraram-se 4 atributos do tipo DD, outros 5 do tipo FA, 4 UD e por fim 2 características do tipo H.

Analisando ainda os resultados da seleção de característica do PSO, mas agora utilizando o *random forest*, foi constatado que a *feature* $DD_{an}$ foi a menos selecionada, de tal forma que essa característica é apresentada apenas nos resultados de 5 experimentos. No total 9 *features* do tipo FA foram identificadas como menos selecionadas, seguidas de 5 *features* do tipo H, além de 2 características DD e uma UD ($UD_{ti}$).

Já na Tabela 11 são apresentadas as características que foram mais selecionadas pelo algoritmo genético. Nos experimentos com o classificador *bayes net* e GA as características $UD_{ti}$, $FA_0$ e MFA tiveram uma frequência de 30, ou seja, estiveram presente nos resultados de todos os experimentos. Quanto as demais características mais selecionadas 4 são do tipo H, outras 4 são do tipo UD, as características DD são 3, uma do tipo P e a característica MH.

Para o classificador C4.5 quando aplicado com o GA as características $P_5$, MH e MFA estiveram presentes em todos os experimentos. Das outras *features* mais selecionadas 6 são do tipo UD, 3 do tipo DD, 3 características FA e 1 *feature* MH.

Nas seleções do algoritmo genético com o k-NN as características mais selecionadas foram a $H_n$, $P_n$ e MFA, tendo sido escolhidas em todos os 30 experimentos. Dentre as *features* mais selecionadas 6 são do tipo H, 5 do tipo P, 2 características FA, 1 DD ($DD_{ti}$), 1 UD ($UD_{ti}$), estando presente ainda a MFA, como já mencionada, e a *feature* MH.

Nas classificações com MLP e GA apenas a característica MFA esteve presente em todos os resultados. Além disso tiveram como *features* mais selecionadas 6 do tipo H, 3 P, 3 do tipo FA, 2 características DD, 1 UD ($UD_{an}$), além de estar presente a *feature* MH.

Tabela 11 – Lista das características mais selecionadas pelo GA

| Bayes Net | | C4.5 | | k-NN | | MLP | | *Naive* Bayes | | SVM | | Random Forest | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $UD_{ti}$ | 30 | $P_5$ | 30 | $H_n$ | 30 | MFA | 30 | $P_i$ | 30 | $H_t$ | 30 | MH | 28 |
| $FA_o$ | 30 | MH | 30 | $P_n$ | 30 | MH | 26 | $P_5$ | 30 | $H_5$ | 30 | MFA | 28 |
| MFA | 30 | MFA | 30 | MFA | 30 | $H_t$ | 25 | $H_{[abc]}$ | 28 | $H_n$ | 30 | $DD_{[Shift]R}$ | 24 |
| $DD_{R[Shift]}$ | 29 | MP | 29 | $DD_{ti}$ | 29 | $H_.$ | 24 | $DD_{ti}$ | 28 | $H_.$ | 29 | $P_5$ | 24 |
| $P_5$ | 29 | $UD_{an}$ | 27 | $UD_{ti}$ | 29 | $DD_{ti}$ | 24 | $H_i$ | 27 | $H_{[abc]}$ | 29 | MP | 24 |
| $FA_{[123?]}$ | 29 | $P_n$ | 26 | MH | 29 | $P_n$ | 24 | $H_{[123?]}$ | 27 | $DD_{ti}$ | 29 | $DD_{[Shift]o}$ | 23 |
| $H_{[123?]}$ | 28 | $DD_{nl}$ | 24 | $H_{[abc]}$ | 28 | $H_e$ | 23 | $H_5$ | 27 | $P_5$ | 29 | $H_.$ | 22 |
| MH | 28 | $UD_{5[abc]}$ | 23 | $P_i$ | 28 | $H_o$ | 23 | $H_l$ | 27 | $P_{[abc]}$ | 29 | $H_{[abc]}$ | 22 |
| $DD_{ti}$ | 27 | $UD_{nl}$ | 23 | $FA_.$ | 28 | $H_a$ | 23 | $H_o$ | 27 | MFA | 29 | $UD_{[Shift]o}$ | 22 |
| $DD_{ie}$ | 27 | $FA_.$ | 22 | $H_o$ | 27 | $DD_{[123?]5}$ | 23 | $H_a$ | 27 | $H_o$ | 28 | $UD_{nl}$ | 22 |
| $UD_{ie}$ | 27 | $FA_{[abc]}$ | 21 | $P_.$ | 27 | $UD_{an}$ | 23 | $FA_{[Shift]}$ | 27 | $H_a$ | 28 | $P_n$ | 22 |
| $H_o$ | 26 | $DD_{an}$ | 20 | $P_5$ | 27 | $P_5$ | 23 | $H_e$ | 26 | $UD_{ti}$ | 28 | $FA_.$ | 22 |
| $H_n$ | 26 | $UD_{R[Shift]}$ | 20 | $H_e$ | 26 | $FA_.$ | 23 | $FA_n$ | 26 | $P_R$ | 28 | $DD_{[123?]5}$ | 21 |
| $UD_{[123?]5}$ | 26 | $P_i$ | 20 | $H_{[Shift]}$ | 26 | $P_.$ | 22 | MFA | 26 | $FA_{[123?]}$ | 28 | $DD_{5[abc]}$ | 21 |
| $UD_{[Shift]R}$ | 26 | $DD_{ti}$ | 19 | $P_{[abc]}$ | 26 | $FA_5$ | 22 | $H_.$ | 25 | $FA_R$ | 28 | $UD_{[Shift]R}$ | 21 |
| $FA_.$ | 26 | $UD_{[Shift]R}$ | 19 | $FA_n$ | 26 | $FA_n$ | 22 | $H_{[Shift]}$ | 25 | $FA_a$ | 28 | $UD_{oa}$ | 21 |
| $H_.$ | 25 | $UD_{ie}$ | 18 | $H_.$ | 25 | $H_i$ | 21 | $FA_5$ | 25 | $H_{[123?]}$ | 27 | $P_e$ | 21 |
| $DD_{[Shift]R}$ | 25 | $DD_{5[abc]}$ | 17 | $DD_{an}$ | 25 | $H_5$ | 21 | $H_t$ | 24 | $UD_{an}$ | 27 | $DD_{nl}$ | 20 |

Já nas classificações com *naive bayes* aplicando-se o GA as *features* que foram selecionadas em todos os experimentos são do tipo P, mais precisamente $P_i$ e $P_5$. Já as outras características mais escolhidas 10 são do tipo H, 3 FA, 2 do tipo P, uma características DD, além de estar presente a MFA.

Em todos os experimentos com o classificador SVM e o GA foram registradas as características $H_t$, $H_5$ e $H_n$, sendo por tanto nesse caso as características mais selecionadas. No total 8 *features* do tipo H estiveram entre as mais selecionadas, 3 *features* do tipo FA, 2 P, 1 DD ($DD_{ti}$), 1 do tipo UD ($DD_{an}$), além de constar nessa lista de mais selecionadas a características MFA.

Por fim, nas classificações com *random forest* nenhuma características esteve presente em todos os resultados. Contudo as *features* MH e MFA foram as mais selecionada, cada uma com uma frequência de 28 ocorrências. Além dessas características foram ainda identificadas com as mais escolhidas 4 do tipo DD, 4 do tipo UD, 3 P, 2 características H e uma do tipo FA (FA.).

Já na Tabela 12 são apresentadas as características que foram menos selecionadas pelo GA. Começando a análise pelos experimentos com o classificador *bayes net* percebe-se que a característica menos selecionadas foi a $P_{[Shift]}$, estando essa característica presente nos resultados de apenas 7 experimentos. Entre as *features* menos selecionadas 3 características são do tipo DD, 3 do tipo P e mais 3 *features* do tipo H, 4 são do tipo FA e 4 UD.

Nas classificações com C4.5, assim como o classificador *bayes net*, porém com um valor de ocorrência ainda mais inferior, a *feature* menos selecionada foi a $P_{[Shift]}$ aparecendo em apenas 4 experimentos. Analisando as características menos selecionadas constatou-se que 6 são do tipo FA, 4 P, 3 *feature* UD, além das características DD e H, cada tipo possuindo 2 representantes.

Nas classificações com k-NN a *feature* menos selecionada foram a $FA_t$ e $FA_l$, cada uma aparecendo em apenas 2 experimentos. No total 7 características do tipo FA foram identificadas como sendo as menos selecionadas, além de estarem na relação das menos escolhidas as características do tipo DD, P e UD, cada uma com 3 representantes, além de 1 característica do tipo H ($H_{[Shift]}$).

Tabela 12 – Lista das características menos selecionadas pelo GA

| Bayes Net | | C4.5 | | k-NN | | MLP | | *Naive* Bayes | | SVM | | Random Forest | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $P_{[Shift]}$ | 7 | $P_{[Shift]}$ | 4 | $FA_{t}$ | 2 | $FA_{o}$ | 9 | $DD_{[abc][Shift]}$ | 0 | $FA_{e}$ | 8 | $H_{[Shift]}$ | 7 |
| $UD_{e[123?]}$ | 9 | $UD_{e[123?]}$ | 6 | $FA_{l}$ | 2 | $FA_{e}$ | 11 | $DD_{[Shift]o}$ | 0 | $DD_{.t}$ | 11 | $H_{R}$ | 9 |
| $P_{a}$ | 10 | $UD_{[123?]5}$ | 6 | $FA_{i}$ | 3 | $FA_{[123?]}$ | 12 | $UD_{e[123?]}$ | 0 | $UD_{e[123?]}$ | 11 | $H_{e}$ | 10 |
| $P_{t}$ | 14 | $H_{[Shift]}$ | 8 | $FA_{e}$ | 3 | $FA_{[Shift]}$ | 12 | $UD_{[abc][Shift]}$ | 0 | $FA_{i}$ | 12 | $DD_{.t}$ | 10 |
| $H_{l}$ | 15 | $H_{a}$ | 8 | $H_{[Shift]}$ | 4 | $UD_{e[123?]}$ | 13 | $DD_{.t}$ | 1 | $DD_{ie}$ | 13 | $DD_{an}$ | 12 |
| $DD_{an}$ | 15 | $DD_{[Shift]R}$ | 8 | $P_{l}$ | 4 | $P_{t}$ | 13 | $DD_{e[123?]}$ | 1 | $H_{[Shift]}$ | 14 | $UD_{[abc][Shift]}$ | 12 |
| $H_{[abc]}$ | 16 | $P_{.}$ | 8 | $P_{[Shift]}$ | 5 | $DD_{.t}$ | 14 | $UD_{.t}$ | 1 | $DD_{nl}$ | 15 | $FA_{i}$ | 12 |
| $FA_{t}$ | 16 | $P_{e}$ | 8 | $FA_{[Shift]}$ | 5 | $DD_{e[123?]}$ | 14 | $UD_{[123?]5}$ | 1 | $UD_{.t}$ | 15 | $FA_{R}$ | 12 |
| $FA_{[Shift]}$ | 16 | $FA_{a}$ | 8 | $DD_{e[123?]}$ | 6 | $DD_{[Shift]o}$ | 14 | $UD_{[Shift]o}$ | 2 | $UD_{oa}$ | 15 | $H_{i}$ | 13 |
| $FA_{a}$ | 16 | $FA_{n}$ | 8 | $FA_{a}$ | 7 | $UD_{.t}$ | 14 | $DD_{[123?]5}$ | 4 | $UD_{nl}$ | 15 | $UD_{.t}$ | 13 |
| $UD_{R[Shift]}$ | 17 | $P_{t}$ | 9 | $UD_{e[123?]}$ | 8 | $P_{[Shift]}$ | 14 | $DD_{R[Shift]}$ | 4 | $FA_{[Shift]}$ | 15 | $FA_{t}$ | 13 |
| $UD_{oa}$ | 17 | $P_{[123?]}$ | 9 | $FA_{o}$ | 8 | $FA_{i}$ | 14 | $UD_{R[Shift]}$ | 4 | $DD_{[abc][Shift]}$ | 16 | $DD_{ti}$ | 14 |
| $P_{e}$ | 17 | $FA_{[Shift]}$ | 9 | $DD_{[abc][Shift]}$ | 9 | $FA_{a}$ | 14 | $FA_{l}$ | 5 | $UD_{[Shift]o}$ | 16 | $UD_{ti}$ | 14 |
| $H_{[Shift]}$ | 18 | $DD_{[Shift]o}$ | 10 | $FA_{[Shift]}$ | 10 | $DD_{nl}$ | 15 | $P_{l}$ | 6 | $DD_{[Shift]o}$ | 17 | $UD_{5[abc]}$ | 14 |
| $DD_{e[123?]}$ | 18 | $UD_{.t}$ | 10 | $DD_{nl}$ | 11 | $P_{i}$ | 15 | $UD_{5[abc]}$ | 9 | $UD_{ie}$ | 17 | $P_{[Shift]}$ | 14 |
| $DD_{[abc][Shift]}$ | 18 | $FA_{[123?]}$ | 10 | $P_{R}$ | 11 | $UD_{[Shift]o}$ | 16 | $UD_{nl}$ | 9 | $UD_{[123?]5}$ | 17 | $FA_{l}$ | 14 |
| $UD_{[Shift]o}$ | 18 | $FA_{5}$ | 11 | $UD_{R[Shift]}$ | 13 | $P_{[123?]}$ | 16 | $DD_{nl}$ | 11 | $UD_{[abc][Shift]}$ | 17 | $FA_{[Shift]}$ | 15 |
| $FA_{l}$ | 18 | $FA_{o}$ | 11 | $UD_{oa}$ | 13 | $FA_{l}$ | 16 | $P_{[Shift]}$ | 11 | $P_{[Shift]}$ | 17 | $FA_{o}$ | 15 |

Os experimentos com MLP, assim como o k-NN, também tiveram como característica menos selecionada uma do tipo FA, porém dessa vez a *feature* menos escolhida foi a $FA_o$. Ao final foram identificadas 6 características do tipo FA como sendo menos selecionadas, além de 4 do tipo P e mais 4 DD, e por fim 3 *features* do tipo UD.

Nos resultados das classificações com *nayve bayes* observou-se a completa ausência das características $DD_{[abc][Shift]}$, $DD_{[Shift]o}$, $UD_{e[123?]}$, $UD_{[abc][Shift]}$. No total foram 7 características do tipo DD e mais 7 UD sendo identificadas como menos selecionadas, além de 2 do tipo P e 1 característica FA ($FA_l$).

Nas classificações com SVM a característica que menos aparece é a $FA_e$, estando presente em apenas 8 experimentos. Assim com nas classificações com o *nayve bayes* aqui o número de características como sendo menos selecionadas foi elevado para os tipos UD e DD, com quantidade de 8 e 5 *features* respectivamente, além disso tiveram 2 características do tipo FA, 1 do tipo P ($P_{[Shift]}$) e outra do tipo H ($H_{[Shift]}$).

Por fim nas classificações com *random forest* a *feature* menos selecionada foi a $H_7$. Foram ainda identificadas como menos selecionadas 6 características do tipo FA, 4 do tipo UD, 3 DD, 3 H e também 1 do tipo P ($P_{[Shift]}$).

Uma vez finalizada a apresentação da primeira análise sobre a frequência das características algumas observações podem ser feitas. Primeiramente podemos observar que as características mais selecionadas podem variar de acordo com os classificadores e métodos de seleção utilizados. O mesmo acontece para as características menos selecionadas, porém nesse caso um fato chama atenção, é a ausência total da ocorrência de algumas características do tipo DD e UD nas classificações com *nayve bayes*, seja utilizando a seleção de atributos por PSO ou GA. Enquanto que a ausência de uma *feature* nos experimentos é algo que ocorre pontualmente, a presença de uma determinada característica em todos os experimentos de um determinado classificador é bastante comum, seja aplicando o processo de seleção de características por PSO ou mesmo o GA - salvo os experimentos do classificador SVM-GA onde a ocorrência máxima foi de 28 para as características MH e MFA.

A segunda análise realizada, ainda se tratando do estudo das frequências das características, tem como objetivo analisar a ocorrência das *features* de acordo com o seu tipo, de tal forma que seja possível se identificar quais os tipos de características que são

mais frequentemente selecionados. Para isso considerou-se como sendo a taxa de frequência de cada tipo a razão entre o registro de ocorrência de todas as *features* de um determinado tipo pela capacidade total de ocorrência das respectivas características.

Desta forma na Figura 13 são apresentadas as taxas de seleção de características de acordo com seus respectivos tipos para as seleções com PSO. Na figura é possível se observar que as características do tipo MFA foram as mais selecionadas, alcançando uma taxa de 85%, evidenciando-se a importância das características MFA para as classificações aqui desempenhadas. Em seguida tem-se a *feature* MH que assim como a MFA também discrimina o valor médio de uma outra característica, apresentando nesse caso a taxa de ocorrência de 67%.

Já as características MP e H aparecem em seguida com taxas de 38% e 28%, respectivamente. Enquanto que as características do tipo P juntas possuem taxa de ocorrência de 22%. Por fim temos as características DD, UD e FA com taxas de frequência de 19%, 17% e 10%, respectivamente.

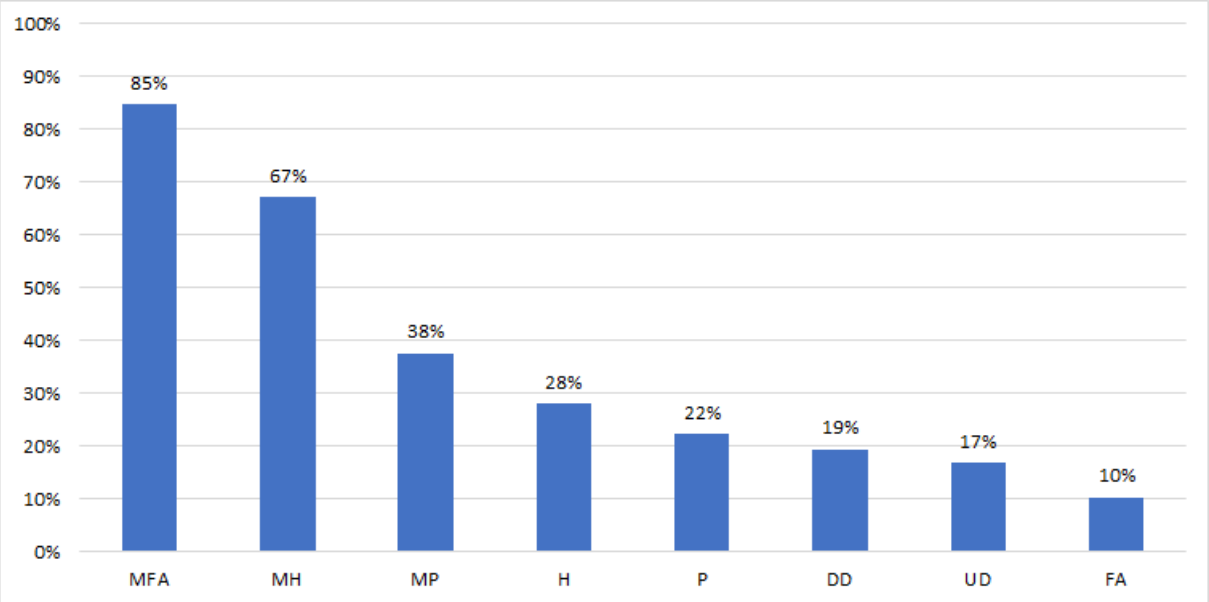


Figura 13 – Visão geral das taxas de ocorrência das características por tipo - Seleção por PSO

A Figura 14, por sua vez, apresenta um gráfico com uma visão geral das taxas de ocorrência de cada tipo de característica para as seleções feitas a partir do algoritmo genético. Nela é apresentada a característica do tipo MFA como sendo a mais selecionada, com taxa de 97% de ocorrência. Em seguida tem-se a *feature* MH com 67%, seguido da características H com 32%, depois vem o tipo MP que obteve taxa de ocorrência de 25%,

18% para a *feature* P, em seguida aparecem as características UD e FA com 15% cada uma e por fim a *feature* DD com 13%, sendo este o tipo de característica menos selecionado.

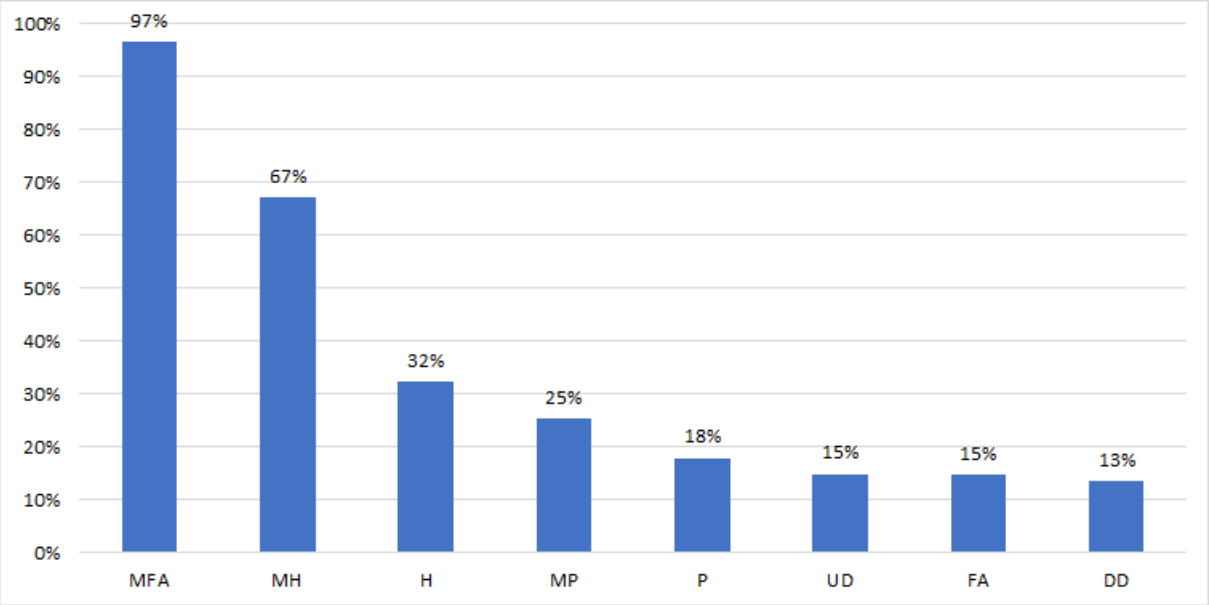


Figura 14 – Visão geral das taxas de ocorrência das características por tipo - Seleção por GA

Observando a característica do tipo MFA nota-se que em ambos os métodos de seleção de característica aqui propostos este foi o tipo de *feature* mais frequente, com uma diferença entre as taxas de frequência entre seleção por PSO e GA de 12%, com mais nível de seleção pelo GA.

Enquanto que MH, tanto no PSO quando no GA, é o segundo tipo de característica com maior frequência de seleção, inclusive com ambos os métodos de seleção apresentando a mesma porcentagem de escolha.

Verificando os tipos MP e H, ainda considerando um ranqueamento das categorias mais selecionadas, esses ocupam respectivamente o 3° e 4° lugar nas seleções por PSO, enquanto que na seleção por GA a ordem é inversa, ou seja, o tipo H é o terceiro mais selecionado seguido tipo MP. Calculando-se a diferença entra as taxas do MP na seleção por PSO e por GA é obtido um valor de 4%, sendo mais selecionado pelo PSO, enquanto que a diferença entre os percentuais da frequência das características do tipo H é de 13%, apresentando maior taxa no GA.

Já a taxa de frequência do tipo P encontra-se em um mesmo patamar nos dois métodos de seleção ocupando o 5° lugar entre o tipo mais selecionado, com uma pequena diferença de 4% entre as taxas do PSO e do GA.

Por último há os tipos DD, UD e FA. Nos dois métodos de seleção esses foram os tipos menos frequentes. A taxa de frequência do tipo DD possui uma diferença 6% entre a seleção por PSO e por GA, sendo que no PSO a sua frequência é maior. Já a categoria FA, que é um pouco mais frequente na seleção por GA, apresenta uma diferença de taxas de 5% ao se comparar com a seleção por PSO. Enquanto que as características do tipo UD também são mais frequentes na seleção por GA e possui uma diferença entre taxas de 3%.

# 8 Conclusão

Nesse trabalho foram propostos dois métodos de seleção de características, são eles: um baseado em PSO (*Particle Swarm Optimization*) e o outro em GA (*Genetic Algorithm*) como forma de otimização ao *keystroke dynamics* nos dispositivos móveis. Ambos os métodos de seleção foram utilizados em conjunto com 7 classificadores - *bayes net*, C4.5 (árvore de decisão), k-NN, MLP (*Multi Layer Perceptron*), SVM (*Support Vector Machine*) e *random forest* - como forma de se avaliar o desempenho dos seletores em diversos cenários, ou seja, através de variadas formas de classificação.

Além disso nesse trabalho foram analisadas as características que foram mais selecionadas e também aquelas que foram menos escolhidas, para fossem identificadas as *features* que possuem maior aptidão, ou seja, aquelas que podem proporcionar um maior nível de acuracidade à dinâmica de digitação. Sendo essa mesma análise feita não apenas para a frequência de seleção de cada característica isoladamente, mas também de acordo com os tipos das características.

Os resultados das taxas de acertos das classificações utilizando os métodos de seleção aqui construídos foram comparados com as taxas de acertos das classificações sem aplicação de seleção de características. Ao final percebeu-se que todas as classificações que utilizaram o PSO tiveram maiores taxas médias de acertos. As taxas de acuracidade dos experimentos com o GA superaram às classificações sem seleção somente quando utilizados alguns classificadores específicos (C4.5, k-NN, MLP, *naive bayes*, *random forest*). De forma geral os resultados obtidos entre as classificações com PSO, com GA e sem seleção de características tiveram valores próximos de acuracidade, sendo a menor diferença entre a melhor taxa e pior taxa de 0,53%, enquanto que a maior diferença entre a melhor acuracidade e a menor acuracidade foi de 5,04%. Aplicado um teste de hipótese percebeu-se que estatisticamente as taxas de acerto das classificações com *naive* bayes foram equivalentes, ou seja, sem diferenças significantes - considerando as classificações com seleção por PSO, GA ou sem seleção - e o mesmo pode ser dito das taxas de acerto do SVM com GA e SVM sem seleção de característica, ou seja, possuem níveis de acuracidade estatisticamente equivalentes.

Além disso os valores dos desvios padrões das taxas de acertos das classificações foram notavelmente menores para os experimentos com os métodos de seleção de características, sobretudo nos experimentos com o GA. Isso mostra que apesar dos resultados da seleção de características serem próximas daquelas obtidas pelas classificações sem uso dos seletores, ainda assim os métodos aqui construídos proporcionam uma classificação mais estável, com baixas taxas de variação de acuracidade, o que é de fundamental importância para o *keystroke dynamics*, pois de um método de autenticação espera-se sobretudo que o mesmo seja estável e coerente, ou seja, que apresentem resultados constantes, com o mínimo de variação.

Uma outra análise que foi feita acerca do desempenho dos métodos de seleção de características, porém desse vez se avaliando as taxas de falso positivo (FAR) e falso negativo (FRR) das classificações. E novamente o PSO demonstrou melhores resultados em todas as suas taxas de erro, tanto de FRR quanto FAR, apresentando valores inferiores aos obtidos pelas classificações utilizando seleção por GA (o que já era de se esperar, uma vez que a taxa de erro é complementar à taxa de acerto). Já os desvios padrões em sua maioria foram ligeiramente menores para os resultados com o GA. Assim conclui-se que o PSO gerou menos erros de classificação, porém o GA apresentou, em alguns casos, uma pequena superioridade em sua estabilidade.

Algo se pode notar é que tanto as taxas de acertos quanto às taxas de erro (FAR e FRR) variaram bastante de acordo com cada classificador utilizado, reforçando assim a ideia demonstrada na literatura de que esses métodos influenciam diretamente o desempenho do *keystroke dynamics*. Nesse trabalho os melhores resultados foram obtidos ao se combinar o classificador *random forest* com o modelo de seleção por PSO. Enquanto que as piores taxas foram obtidas ao se utilizar o classificador C4.5, seja utilizando o PSO ou o GA.

Quanto às taxas de redução de características essas foram maiores para as seleções com algoritmo genético em todas as classificações. Já aos desvios padrões foram menores em alguns casos para o GA e em outros para o PSO, a depender do classificador utilizado.

Quanto as frequências com que as características são selecionadas, tanto na seleção por PSO quanto por GA também, essas variaram de acordo com o classificador, ou seja, não foi identificado um conjunto de características que apresentasse uma mesma tendência de escolha em todas as classificações. Assim podemos sugerir que algumas características

podem ser mais eficientes, ou preferíveis, em alguns classificadores do que em outros.

Inclusive houveram situações em que algumas características estiveram completamente ausentes em todos os experimentos de um determinado classificador. Isso ocorreu nas classificações com *nayve bayes*, que quando utilizado em conjunto com o PSO tiveram uma característica do tipo DD e outra UD com taxas de ocorrência 0, já quando associado à seleção com o GA, 2 características do tipo DD e 2 UD também estiveram ausentes em todos os resultados dos experimentos. Esse comportamento de eliminação total de uma característica foi observado apenas nos resultados do *nayve bayes*, reforçando a ideia de afinidade entre classificador e *feature*.

Algo importante que se pode observar em uma segunda análise foi a frequência das seleções das características de acordo com os tipos das *features*. Fazendo uma comparação entre os resultados da seleção por PSO e os resultados do GA, em ambos os métodos o MFA e o MH foram os mais selecionados, sendo o MFA o tipo mais escolhido, seguido do tipo MH. Os demais tipos também tiveram taxas de frequência bastante próximas, sendo apresentado inclusive valores iguais para o tipo P.

Apesar de não se poder identificar uma preferência generalizada pelas características dos dispositivos móveis em detrimento às *features* convencionais, nos resultados aqui obtidos foi registrado que uma característica exclusiva dos dispositivos móveis (MFA) foi a mais selecionada, sugerindo que as *features* inerentes a estes aparelhos podem proporcionar melhorias ao *keystroke dynamics*.

Um dos desafios enfrentados ao longo do desenvolvimento desse trabalho está relacionado à própria complexidade das soluções propostas, mais precisamente, o desempenho computacional inerentes aos métodos de seleção de características - considerando-se a abordagem de aprendizado de máquina - e às classificações. Apesar do WEKA ser uma ferramenta robusta e versátil, permitindo que diversos recursos sejam adicionados a uma aplicação apenas se utilizando a sua API, durante os experimentos desse trabalho foi possível observar que alguns métodos de classificação são demasiadamente lentos. As classificações sobretudo com MLP são bastante demoradas, de tal forma que um único experimento com o MLP chegou a levar 17 dias para ser concluído. O que inviabiliza inclusive a utilização de alguns parâmetros com valores mais elevados para os algoritmos do PSO e do GA, por exemplo um grande aumento de iterações ou do tamanho da população.

Superadas as limitações e considerando os resultados aqui apresentados podemos concluir que os modelos de características aqui construídos conseguem proporcionar melhorias ao *keystroke dynamics* em dispositivos móveis, que apesar de serem dois métodos com abordagens distintas, de forma geral, apresentaram resultados semelhantes, o que reforça a qualidade e ou coerência das soluções propostas. Além disso pode-se considerar que as características intrínsecas aos dispositivos móveis são objetos de melhorias ao *keystroke dynamics*.

## 8.1 Contribuições deste trabalho

As principais contribuições deste trabalho podem ser enumeradas da seguinte forma:

- Um estudo sobre o *keystroke dynamics* e seus principais aspectos (tipos de validação, métricas de avaliação de desempenho, bases de dados públicas, tipos de *features* e seleção de características);
- Desenvolvimento de dois métodos de seleção de características, um baseado em PSO e outro em GA, que conseguiram proporcionar uma diminuição na dimensionalidade do conjunto de características originais ao mesmo tempo em proporcionaram melhorias nas taxas de acuracidade e nas taxas de erro (FAR e FRR), quando comparados às classificações sem uso de seleção de atributos. Esses métodos não somente apresentaram resultados competitivos com outros já disponíveis na literatura, mas sobretudo se destacaram pelo seu alto nível de estabilidade, o que foi demonstrado através das baixas taxas de variabilidade (desvio padrão e coeficiente de variação) dos seus resultados. Um outro ponto que reforça a integridade dos modelos aqui propostos é a semelhança entre os resultados obtidos por cada um, evidenciando assim a robustez dos processos de busca pelos melhores atributos;
- Através desse trabalho foi possível se observar a oscilação acentuada no desempenho do *keystroke dynamics* ao utilizar vários métodos de classificação;
- Identificação das características que proporcionam maior desempenho ou que impactam negativamente no *keystroke dynamics* quando aplicado em dispositivos móveis;
- Análise das características do *keystroke dynamics* em dispositivos móveis que são mais preferíveis ou rejeitadas por um um determinado classificador. Foi observado

que um atributo pode ser bastante benéfico para o desempenho de um classificador e demonstrar pouca afinidade com outro tipo de método de classificação. Isto pode servir de motivação para a elaboração de um estudo mais aprofundado sobre a relação entre o funcionamento de um classificador e a um terminado tipo de característica do *keystroke dynamics*.

- Verificação da frequência de escolha dos atributos do *keystroke dynamics* de acordo com cada tipo de característica. Através dessa tarefa foi possível observar que o tipo de característica mais escolhido pelos métodos de seleção aqui propostos é um atributo obtidos exclusivamente por dispositivos móveis, servindo de incentivo para o desenvolvimento de novos estudos acerca das características da dinâmica de digitação que são inerentes aos dispositivos móveis.

## 8.2 Trabalhos futuros

Como trabalhos futuros pretende-se inicialmente replicar os experimentos em outras bases de dados, a fim de se avaliar o comportamento dos métodos de seleção de características aqui construídos através de outro conjunto de informações, permitindo assim inclusive se verificar o comportamento dos classificadores ao se utilizar um novo conjunto de dados, procurando-se observar, por exemplo, se o *random forest* ainda será o melhor ou se o *C4.5* será o pior classificador.

Também é desejado desenvolver novos trabalhos ainda se utilizando a base de dados aqui empregada, porém dessa vez suprimindo alguns atributos, por exemplo, removendo-se algumas das *features* que foram menos selecionadas. Uma vez obtido um novo conjunto de dados, aplicar novamente os modelos aqui desenvolvidos a fim de se verificar o comportamento dos métodos de seleção bem como a acuracidade do *keystroke dynamics*.

Além disso é desejado se construir uma base de dados com outras características próprias dos dispositivos móveis (rotação do aparelho, percurso do dedo sobre a tela, dentre outras) e então aplicar os métodos de seleção e verificar se existe uma preferência por alguma nova *feature*, realizando nesse caso experimentos com e sem atributos temporais. O objetivo dessa proposta é explorar características que são geradas através do dinamismo da manipulação dos dispositivos móveis pelo usuários, de forma a não somente verificar aspectos da digitação (batida na tela) mas também observar como o usuário manuseia o

aparelho.

Um outro ponto que se espera abordar futuramente são os classificadores, ou seja, aplicar outros métodos de classificação ainda pouco utilizados no *keystroke dynamics* como por exemplo a classificação por *deep learning* e se verificar as melhorias que esses novos classificadores podem agregar à dinâmica de digitação.

Como trabalho futuro é desejado fazer alterações nos métodos de seleção de características, por exemplo, o uso de outras topologias no PSO, a aplicação de outros operados ao GA e até mesmo explorar a abordagem multiobjetivo nesses algoritmos.

Por fim espera-se utilizar outras ferramentas de apoio à classificação em substituição ao WEKA, como forma a tentar minimizar as dificuldades encontradas durante o uso da ferramenta, conforme descrito no Capítulo 7

# Referências

ABIDO, M. Optimal power flow using particle swarm optimization. *International Journal of Electrical Power & Energy Systems*, Elsevier, v. 24, n. 7, p. 563–571, 2002. Citado 2 vezes nas páginas 55 e 66.

ALI, A. *Dynamic and Advanced Data Mining for Progressing Technological Development: Innovations and Systemic Approaches: Innovations and Systemic Approaches*. [S.l.]: IGI Global, 2009. Citado na página 67.

ALI, M. L. et al. Keystroke biometric systems for user authentication. *Journal of Signal Processing Systems*, Springer, v. 86, n. 2-3, p. 175–190, 2017. Citado 5 vezes nas páginas 8, 22, 23, 24 e 38.

ALSULTAN, A.; WARWICK, K.; WEI, H. Non-conventional keystroke dynamics for user authentication. *Pattern Recognition Letters*, Elsevier, v. 89, p. 53–59, 2017. Citado 5 vezes nas páginas 24, 27, 28, 29 e 81.

ALZUBAIDI, A.; KALITA, J. Authentication of smartphone users using behavioral biometrics. *IEEE Communications Surveys & Tutorials*, IEEE, v. 18, n. 3, p. 1998–2026, 2016. Citado 2 vezes nas páginas 30 e 31.

ANTAL, M.; SZABÓ, L. Z.; LÁSZLÓ, I. Keystroke dynamics on android platform. *Procedia Technology*, Elsevier, v. 19, p. 820–826, 2015. Citado 15 vezes nas páginas 17, 26, 31, 35, 36, 37, 39, 40, 41, 45, 59, 65, 66, 67 e 68.

ARANGO, H. G. Bioestatística: teórica e computacional. In: *Bioestatística: teórica e computacional*. [S.l.: s.n.], 2005. Citado na página 72.

AZEVEDO, G. L.; CAVALCANTI, G. D.; FILHO, E. C. C. An approach to feature selection for keystroke dynamics systems based on pso and feature weighting. In: IEEE. *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*. [S.l.], 2007. p. 3577–3584. Citado na página 80.

BALAGANI, K. S. et al. On the discriminability of keystroke feature vectors used in fixed text keystroke authentication. *Pattern Recognition Letters*, Elsevier, v. 32, n. 7, p. 1070–1080, 2011. Citado 4 vezes nas páginas 21, 35, 36 e 37.

BARTLOW, N.; CUKIC, B. Evaluating the reliability of credential hardening through keystroke dynamics. In: IEEE. *Software Reliability Engineering, 2006. ISSRE'06. 17th International Symposium on*. [S.l.], 2006. p. 117–126. Citado 2 vezes nas páginas 29 e 65.

BEASLEY, D.; MARTIN, R.; BULL, D. An overview of genetic algorithms: Part 1. fundamentals. *University computing*, WHURR PUBLISHERS, v. 15, p. 58–58, 1993. Citado na página 52.

BOURS, P. Continuous keystroke dynamics: A different perspective towards biometric evaluation. *Information Security Technical Report*, Elsevier, v. 17, n. 1, p. 36–43, 2012. Citado 2 vezes nas páginas 21 e 22.

BREIMAN, L. Bagging predictors. *Machine learning*, Springer, v. 24, n. 2, p. 123–140, 1996. Citado na página 39.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001. Citado 2 vezes nas páginas 39 e 40.

CAVALCANTI, G. Darmiton da C. *Composição de biometria para sistemas multimodais de verificação de identidade pessoal*. Tese (Doutorado) — Universidade Federal de Pernambuco, 2005. Citado na página 64.

CHERIFI, F. et al. Performance evaluation of behavioral biometric systems. *Behavioral Biometrics for Human Identification: Intelligent Applications*, p. 57–74, 2009. Citado na página 24.

CLARKE, N. L.; FURNELL, S. M. Authentication of users on mobile telephones–a survey of attitudes and practices. *Computers & Security*, Elsevier, v. 24, n. 7, p. 519–527, 2005. Citado na página 15.

CLARKE, N. L.; FURNELL, S. M. Authenticating mobile phone users using keystroke analysis. *International Journal of Information Security*, Springer, v. 6, n. 1, p. 1–14, 2006. Citado na página 30.

CORD, M.; CUNNINGHAM, P. *Machine learning techniques for multimedia: case studies on organization and retrieval*. [S.l.]: Springer Science & Business Media, 2008. Citado 2 vezes nas páginas 40 e 41.

COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE transactions on information theory*, IEEE, v. 13, n. 1, p. 21–27, 1967. Citado na página 40.

DARABSEH, A.; NAMIN, A. S. Effective user authentications using keystroke dynamics based on feature selections. In: IEEE. *Machine Learning and Applications (ICMLA), 2015 IEEE 14th International Conference on*. [S.l.], 2015. p. 307–312. Citado na página 48.

DASH, M.; LIU, H. Feature selection for classification. *Intelligent data analysis*, Elsevier, v. 1, n. 1-4, p. 131–156, 1997. Citado 3 vezes nas páginas 46, 47 e 81.

DEB, K. Genetic algorithm in search and optimization: the technique and applications. In: CITESEER. *Proceedings of international workshop on soft computing and intelligent systems*. [S.l.], 1998. p. 58–87. Citado 2 vezes nas páginas 8 e 51.

DORIGO, M.; MANIEZZO, V.; COLORNI, A. Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, IEEE, v. 26, n. 1, p. 29–41, 1996. Citado na página 47.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern classification*. [S.l.]: John Wiley & Sons, 2012. Citado 13 vezes nas páginas 8, 34, 35, 36, 37, 38, 39, 40, 42, 44, 45, 80 e 81.

EIBEN, A. E.; SMITH, J. E. *Introduction to evolutionary computing*. [S.l.]: Springer, 2003. v. 53. Citado 7 vezes nas páginas 8, 50, 51, 52, 53, 54 e 64.

EL-ABED, M.; DAFER, M.; KHAYAT, R. E. Rhu keystroke: A mobile-based benchmark for keystroke dynamics systems. In: IEEE. *Security Technology (ICCST), 2014 International Carnahan Conference on*. [S.l.], 2014. p. 1–4. Citado 2 vezes nas páginas 26 e 31.

FENG, T. et al. Continuous mobile authentication using touchscreen gestures. In: IEEE. *Homeland Security (HST), 2012 IEEE Conference on Technologies for*. [S.l.], 2012. p. 451–456. Citado 4 vezes nas páginas 17, 31, 32 e 39.

FERREIRA, T. A. E. *Uma nova metodologia híbrida inteligente para a previsão de séries temporais*. Tese (Doutorado) — Universidade Federal de Pernambuco, 2006. Citado 2 vezes nas páginas 43 e 44.

FILHO, J. R. M.; FREIRE, E. O. On the equalization of keystroke timing histograms. *Pattern Recognition Letters*, Elsevier, v. 27, n. 13, p. 1440–1446, 2006. Citado na página 25.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. *The elements of statistical learning*. [S.l.]: Springer series in statistics New York, 2001. v. 1. Citado 5 vezes nas páginas 39, 40, 41, 42 e 43.

FRIEDMAN, N.; GEIGER, D.; GOLDSZMIDT, M. Bayesian network classifiers. *Machine learning*, Springer, v. 29, n. 2-3, p. 131–163, 1997. Citado 3 vezes nas páginas 33, 36 e 43.

FURNELL, S. M. et al. Applications of keystroke analysis for improved login security and continuous user authentication. In: *Information systems security*. [S.l.]: Springer, 1996. p. 283–294. Citado na página 21.

GAINES, R. S. et al. *Authentication by keystroke timing: Some preliminary results*. [S.l.], 1980. Citado 3 vezes nas páginas 16, 20 e 21.

GHEYAS, I. A.; SMITH, L. S. Feature subset selection in large dimensionality domains. *Pattern recognition*, Elsevier, v. 43, n. 1, p. 5–13, 2010. Citado 2 vezes nas páginas 46 e 47.

GIOT, R.; DORIZZI, B.; ROSENBERGER, C. A review on the public benchmark databases for static keystroke dynamics. *Computers & Security*, Elsevier, v. 55, p. 46–61, 2015. Citado na página 24.

GIOT, R.; EL-ABED, M.; ROSENBERGER, C. Greyc keystroke: a benchmark for keystroke dynamics biometric systems. In: IEEE. *Biometrics: Theory, Applications, and Systems, 2009. BTAS'09. IEEE 3rd International Conference on*. [S.l.], 2009. p. 1–6. Citado 2 vezes nas páginas 24 e 25.

GIOT, R.; EL-ABED, M.; ROSENBERGER, C. *Keystroke dynamics authentication*. [S.l.]: InTech, 2011. Citado 6 vezes nas páginas 17, 23, 24, 25, 46 e 47.

GISLASON, P. O.; BENEDIKTSSON, J. A.; SVEINSSON, J. R. Random forests for land cover classification. *Pattern Recognition Letters*, Elsevier, v. 27, n. 4, p. 294–300, 2006. Citado 2 vezes nas páginas 39 e 40.

GIUFFRIDA, C. et al. I sensed it was you: authenticating mobile users with sensor-enhanced keystroke dynamics. In: SPRINGER. *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. [S.l.], 2014. p. 92–111. Citado 3 vezes nas páginas 17, 31 e 32.

GOLDBERG, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. [S.l.]: Addison-Wesley, 1989. Citado 2 vezes nas páginas 50 e 52.

GONZALEZ, N.; CALOT, E. P. Finite context modeling of keystroke dynamics in free text. In: IEEE. *Biometrics Special Interest Group (BIOSIG), 2015 International Conference of the*. [S.l.], 2015. p. 1–5. Citado na página 29.

HAYKIN, S. *Redes neurais: princípios e prática*. [S.l.]: Bookman Editora, 2007. Citado 4 vezes nas páginas 42, 43, 44 e 45.

HEMPSTALK, K.; FRANK, E.; WITTEN, I. H. One-class classification by combining density and class probability estimation. In: SPRINGER. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. [S.l.], 2008. p. 505–519. Citado 2 vezes nas páginas 28 e 39.

HO, J.; KANG, D.-K. One-class naïve bayes with duration feature ranking for accurate user authentication using keystroke dynamics. *Applied Intelligence*, Springer, p. 1–18, 2017. Citado na página 36.

HOLLAND, J. H. *Adaptation in Natural and Artificial Systems*. Michigan-USA: University of Michigan Press, 1975. Citado 2 vezes nas páginas 47 e 50.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: theory and applications. *Neurocomputing*, Elsevier, v. 70, n. 1, p. 489–501, 2006. Citado na página 47.

IDRUS, S. Z. S. et al. Soft biometrics database: a benchmark for keystroke dynamics biometric systems. In: IEEE. *Biometrics Special Interest Group (BIOSIG), 2013 international conference of the*. [S.l.], 2013. p. 1–8. Citado 2 vezes nas páginas 21 e 26.

JOYCE, R.; GUPTA, G. Identity authentication based on keystroke latencies. *Communications of the ACM*, ACM, v. 33, n. 2, p. 168–176, 1990. Citado 2 vezes nas páginas 16 e 21.

KARAKATIČ, S.; PODGORELEC, V. A survey of genetic algorithms for solving multi depot vehicle routing problem. *Applied Soft Computing*, Elsevier, v. 27, p. 519–532, 2015. Citado 2 vezes nas páginas 53 e 54.

KARNAN, M.; AKILA, M.; KRISHNARAJ, N. Biometric personal authentication using keystroke dynamics: A review. *Applied Soft Computing*, Elsevier, v. 11, n. 2, p. 1565–1573, 2011. Citado 3 vezes nas páginas 14, 15 e 33.

KARNAN, M.; KRISHNARAJ, N. Bio password—keystroke dynamic approach to secure mobile devices. In: IEEE. *Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on*. [S.l.], 2010. p. 1–4. Citado na página 80.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. *IEEE International Conference on Neural Networks*, IEEE, 1995. Citado 3 vezes nas páginas 47, 54 e 55.

KENNEDY, J.; EBERHART, R. C. A discrete binary version of the particle swarm algorithm. In: IEEE. *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*. [S.l.], 1997. v. 5, p. 4104–4108. Citado 2 vezes nas páginas 57 e 58.

KENNEDY, J.; MENDES, R. Population structure and particle swarm performance. In: IEEE. *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*. [S.l.], 2002. v. 2, p. 1671–1676. Citado na página 58.

KILLOURHY, K. S.; MAXION, R. A. Comparing anomaly-detection algorithms for keystroke dynamics. In: IEEE. *Dependable Systems & Networks, 2009. DSN'09. IEEE/IFIP International Conference on*. [S.l.], 2009. p. 125–134. Citado 5 vezes nas páginas 23, 25, 33, 34 e 35.

KOHAVI, R.; JOHN, G. H. Wrappers for feature subset selection. *Artificial intelligence*, Elsevier, v. 97, n. 1-2, p. 273–324, 1997. Citado na página 47.

KOŁAKOWSKA, A. Usefulness of keystroke dynamics features in user authentication and emotion recognition. In: *Human-Computer Systems Interaction*. [S.l.]: Springer, 2018. p. 42–52. Citado 4 vezes nas páginas 40, 46, 47 e 65.

KUMAR, R. Blending roulette wheel selection & rank selection in genetic algorithms. *International Journal of Machine Learning and Computing*, IACSIT Press, v. 2, n. 4, p. 365, 2012. Citado na página 63.

LEBERKNIGHT, C. S.; WIDMEYER, G. R.; RECCE, M. L. An investigation into the efficacy of keystroke analysis for perimeter defense and facility access. In: IEEE. *Technologies for Homeland Security, 2008 IEEE Conference on*. [S.l.], 2008. p. 345–350. Citado na página 35.

LEE, S. et al. Modified binary particle swarm optimization. *Progress in Natural Science*, Elsevier, v. 18, n. 9, p. 1161–1166, 2008. Citado 5 vezes nas páginas 55, 56, 57, 58 e 66.

LEE, S.-H. et al. A study on feature of keystroke dynamics for improving accuracy in mobile environment. In: SPRINGER. *International Workshop on Information Security Applications*. [S.l.], 2016. p. 366–375. Citado 5 vezes nas páginas 16, 17, 27, 30 e 31.

LEGGETT, J.; WILLIAMS, G. Verifying identity via keystroke characterstics. *International Journal of Man-Machine Studies*, Elsevier, v. 28, n. 1, p. 67–76, 1988. Citado na página 21.

LEGGETT, J. et al. Dynamic identity verification via keystroke characteristics. *International Journal of Man-Machine Studies*, Elsevier, v. 35, n. 6, p. 859–870, 1991. Citado 2 vezes nas páginas 20 e 21.

LI, Y. et al. Study on the beihang keystroke dynamics database. In: IEEE. *Biometrics (IJCB), 2011 International Joint Conference on*. [S.l.], 2011. p. 1–5. Citado na página 26.

LIAW, A.; WIENER, M. et al. Classification and regression by randomforest. *R news*, v. 2, n. 3, p. 18–22, 2002. Citado 2 vezes nas páginas 39 e 40.

LIU, Q. et al. Topology selection for particle swarm optimization. *Information Sciences*, Elsevier, v. 363, p. 154–173, 2016. Citado na página 58.

LORENA, A. C.; CARVALHO, A. C. de. Uma introdução às support vector machines. *Revista de Informática Teórica e Aplicada*, v. 14, n. 2, p. 43–67, 2007. Citado 2 vezes nas páginas 42 e 43.

MAVROVOUNIOTIS, M.; LI, C.; YANG, S. A survey of swarm intelligence for dynamic optimization: algorithms and applications. *Swarm and Evolutionary Computation*, Elsevier, 2017. Citado na página 52.

MEDINA, A. J. R.; PULIDO, G. T.; RAMÍREZ-TORRES, J. G. A comparative study of neighborhood topologies for particle swarm optimizers. In: *IJCCI*. [S.l.: s.n.], 2009. p. 152–159. Citado na página 58.

MILLER, B. Vital signs of identity [biometrics]. *IEEE spectrum*, IEEE, v. 31, n. 2, p. 22–30, 1994. Citado na página 20.

MONROSE, F.; RUBIN, A. D. Keystroke dynamics as a biometric for authentication. *Future Generation computer systems*, Elsevier, v. 16, n. 4, p. 351–359, 2000. Citado 5 vezes nas páginas 16, 21, 33, 34 e 41.

MONTALVAO, J.; ALMEIDA, C. A. S.; FREIRE, E. O. Equalization of keystroke timing histograms for improved identification performance. In: IEEE. *Telecommunications Symposium, 2006 International*. [S.l.], 2006. p. 560–565. Citado na página 25.

MOSKOVITCH, R. et al. Identity theft, computers and behavioral biometrics. In: IEEE. *Intelligence and Security Informatics, 2009. ISI'09. IEEE International Conference on*. [S.l.], 2009. p. 155–160. Citado 2 vezes nas páginas 16 e 28.

MURTY, M. N.; DEVI, V. S. *Pattern recognition: An algorithmic approach*. [S.l.]: Springer Science & Business Media, 2011. Citado 7 vezes nas páginas 35, 36, 37, 38, 39, 42 e 43.

NG, A.; PERERA, B. Selection of genetic algorithm operators for river water quality model calibration. *Engineering Applications of Artificial Intelligence*, Elsevier, v. 16, n. 5, p. 529–541, 2003. Citado na página 50.

NGUYEN, T.; LE, T.; LE, B. Keystroke dynamics extraction by independent component analysis and bio-matrix for user authentication. *PRICAI 2010: Trends in Artificial Intelligence*, Springer, p. 477–486, 2010. Citado na página 30.

OBAIDAT, M. S. A verification methodology for computer systems users. In: ACM. *Proceedings of the 1995 ACM symposium on Applied computing*. [S.l.], 1995. p. 258–262. Citado na página 23.

PISANI, P. H.; LORENA, A. C. A systematic review on keystroke dynamics. *Journal of the Brazilian Computer Society*, Springer, v. 19, n. 4, p. 573–587, 2013. Citado 3 vezes nas páginas 23, 27 e 28.

QUINLAN, J. R. *C4. 5: programs for machine learning*. [S.l.]: Elsevier, 1993. Citado 2 vezes nas páginas 38 e 39.

RAVINDRAN, S.; GAUTAM, C.; TIWARI, A. Keystroke user recognition through extreme learning machine and evolving cluster method. In: IEEE. *Computational Intelligence and Computing Research (ICCIC), 2015 IEEE International Conference on*. [S.l.], 2015. p. 1–5. Citado na página 21.

RAZALI, N. M.; GERAGHTY, J. et al. Genetic algorithm performance with different selection strategies in solving tsp. In: *Proceedings of the world congress on engineering*. [S.l.: s.n.], 2011. v. 2, p. 1134–1139. Citado na página 63.

REFAEILZADEH, P.; TANG, L.; LIU, H. Cross-validation. In: *Encyclopedia of database systems*. [S.l.]: Springer, 2009. p. 532–538. Citado na página 69.

REID, P. *Biometrics for network security*. [S.l.]: Prentice Hall Professional, 2004. Citado 2 vezes nas páginas 8 e 24.

ROH, J.-h.; LEE, S.-H.; KIM, S. Keystroke dynamics for authentication in smartphone. In: IEEE. *Information and Communication Technology Convergence (ICTC), 2016 International Conference on*. [S.l.], 2016. p. 1155–1159. Citado na página 31.

ROTH, J. et al. Biometric authentication via keystroke sound. In: IEEE. *Biometrics (ICB), 2013 International Conference on*. [S.l.], 2013. p. 1–8. Citado na página 30.

ROTH, J. et al. Investigating the discriminative power of keystroke sound. *IEEE Transactions on Information Forensics and Security*, IEEE, v. 10, n. 2, p. 333–345, 2015. Citado na página 30.

RYBNIK, M.; TABEDZKI, M.; SAEED, K. A keystroke dynamics based system for user identification. In: IEEE. *Computer Information Systems and Industrial Management Applications, 2008. CISIM'08. 7th*. [S.l.], 2008. p. 225–230. Citado na página 35.

SAEVANEE, H.; BHATARAKOSOL, P. User authentication using combination of behavioral biometrics over the touchpad acting like touch screen of mobile device. In: IEEE. *Computer and Electrical Engineering, 2008. ICCEE 2008. International Conference on*. [S.l.], 2008. p. 82–86. Citado 2 vezes nas páginas 16 e 30.

SAINI, B. S.; KAUR, N.; BHATIA, K. S. Keystroke dynamics for mobile phones: A survey. *Indian Journal of Science and Technology*, v. 9, n. 6, 2016. Citado 4 vezes nas páginas 16, 22, 26 e 33.

SENATHIPATHI, K.; BATRI, K. An analysis of particle swarm optimization and genetic algorithm with respect to keystroke dynamics. In: IEEE. *Green Computing Communication and Eflectrical Engineering (ICGCCEE), 2014 International Conference on*. [S.l.], 2014. p. 1–11. Citado 3 vezes nas páginas 46, 48 e 80.

SHABTAI, A. et al. "andromaly": a behavioral malware detection framework for android devices. *Journal of Intelligent Information Systems*, Springer, v. 38, n. 1, p. 161–190, 2012. Citado na página 81.

SHANMUGAPRIYA, D.; GANAPATHI, P. A wrapper-based classification approach for personal identification through keystroke dynamics using soft computing techniques. *Developing Next-Generation Countermeasures for Homeland Security Threat Prevention*, IGI Global, p. 330, 2016. Citado 2 vezes nas páginas 15 e 16.

SHANMUGAPRIYA, D.; PADMAVATHI, G. A survey of biometric keystroke dynamics: Approaches, security and challenges. *International Journal of Computer Science and Information Security*, 2009. Citado 5 vezes nas páginas 14, 15, 22, 23 e 24.

SHANMUGAPRIYA, D.; PADMAVATHI, G. An efficient feature selection technique for user authentication using keystroke dynamics. *IJCSNS International Journal of Computer Science and Network Security*, v. 11, n. 10, p. 191–195, 2011. Citado 3 vezes nas páginas 46, 47 e 50.

SHI, Y.; EBERHART, R. A modified particle swarm optimizer. In: IEEE. *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. [S.l.], 1998. p. 69–73. Citado 2 vezes nas páginas 55 e 56.

SHINDE, P.; SHETTY, S.; MEHRA, M. Survey of keystroke dynamics as a biometric for static authentication. *International Journal of Computer Science and Information Security*, LJS Publishing, v. 14, n. 4, p. 203, 2016. Citado 4 vezes nas páginas 20, 27, 33 e 34.

SUNG, K.-s.; CHO, S. Ga svm wrapper ensemble for keystroke dynamics authentication. In: SPRINGER. *International conference on Biometrics*. [S.l.], 2006. p. 654–660. Citado na página 80.

TAKAHASHI, A. *Máquina de vetores-suporte intervalar*. Tese (Doutorado) — Universidade Federal do Rio Grande do Norte, 2012. Citado na página 43.

TANG, K.-S. et al. Genetic algorithms and their applications. *IEEE signal processing magazine*, IEEE, v. 13, n. 6, p. 22–37, 1996. Citado 3 vezes nas páginas 52, 53 e 54.

TEH, P. S.; TEOH, A. B. J.; YUE, S. A survey of keystroke dynamics biometrics. *The Scientific World Journal*, Hindawi Publishing Corporation, v. 2013, 2013. Citado 8 vezes nas páginas 14, 16, 21, 22, 24, 27, 28 e 30.

TROJAHN, M.; ORTMEIER, F. Biometric authentication through a virtual keyboard for smartphones. *International Journal of Computer Science & Information Technology*, Academy & Industry Research Collaboration Center (AIRCC), v. 4, n. 5, p. 1, 2012. Citado 2 vezes nas páginas 31 e 33.

TROJAHN, M.; ORTMEIER, F. Toward mobile authentication with keystroke dynamics on mobile phones and tablets. In: IEEE. *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*. [S.l.], 2013. p. 697–702. Citado 4 vezes nas páginas 17, 31, 37 e 45.

UNLER, A.; MURAT, A. A discrete particle swarm optimization method for feature selection in binary classification problems. *European Journal of Operational Research*, Elsevier, v. 206, n. 3, p. 528–539, 2010. Citado na página 46.

VAPNIK, V. *The nature of statistical learning theory*. [S.l.]: Springer science & business media, 2013. Citado 2 vezes nas páginas 41 e 42.

VILLANI, M. et al. Keystroke biometric recognition studies on long-text input under ideal and application-oriented conditions. In: IEEE. *Computer Vision and Pattern Recognition Workshop, 2006. CVPRW'06. Conference on*. [S.l.], 2006. p. 39–39. Citado na página 29.

VURAL, E. et al. Shared research dataset to support development of keystroke authentication. In: IEEE. *Biometrics (IJCB), 2014 IEEE International Joint Conference on*. [S.l.], 2014. p. 1–8. Citado na página 26.

WANG, J.; NESKOVIC, P.; COOPER, L. N. Improving nearest neighbor rule with a simple adaptive distance measure. *Pattern Recognition Letters*, Elsevier, v. 28, n. 2, p. 207–213, 2007. Citado 2 vezes nas páginas 34 e 35.

XUE, B.; ZHANG, M.; BROWNE, W. N. Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE transactions on cybernetics*, IEEE, v. 43, n. 6, p. 1656–1671, 2013. Citado 2 vezes nas páginas 46 e 47.

YU, E.; CHO, S. Ga-svm wrapper approach for feature subset selection in keystroke dynamics identity verification. In: IEEE. *Neural Networks, 2003. Proceedings of the International Joint Conference on*. [S.l.], 2003. v. 3, p. 2253–2257. Citado na página 52.

YU, E.; CHO, S. Keystroke dynamics identity verification—its problems and practical solutions. *Computers & Security*, Elsevier, v. 23, n. 5, p. 428–440, 2004. Citado 3 vezes nas páginas 17, 45 e 46.

ZHANG, Y.; WANG, S.; JI, G. A comprehensive survey on particle swarm optimization algorithm and its applications. *Mathematical Problems in Engineering*, Hindawi Publishing Corporation, v. 2015, 2015. Citado 3 vezes nas páginas 56, 58 e 66.

ZHANG, Y. et al. Binary pso with mutation operator for feature selection using decision tree applied to spam detection. *Knowledge-Based Systems*, Elsevier, v. 64, p. 22–31, 2014. Citado 3 vezes nas páginas 55, 56 e 69.

ZHAO, Y. Learning user keystroke patterns for authentication. In: *Proceeding of World Academy of Science, Engineering and Technology*. [S.l.: s.n.], 2006. v. 14, p. 65–70. Citado na página 39.

ZHONG, Y.; DENG, Y. A survey on keystroke dynamics biometrics: approaches, advances, and evaluations. *Recent Advances in User Authentication Using Keystroke Dynamics Biometrics. Science Gate Publishing*, p. 1–22, 2015. Citado 4 vezes nas páginas 16, 22, 23 e 24.

ZHONG, Y.; DENG, Y.; JAIN, A. K. Keystroke dynamics for user authentication. In: IEEE. *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. [S.l.], 2012. p. 117–123. Citado 2 vezes nas páginas 34 e 35.

ZHOU, L. et al. Harmonized authentication based on thumbstroke dynamics on touch screen mobile phones. *Decision Support Systems*, Elsevier, v. 92, p. 14–24, 2016. Citado 4 vezes nas páginas 27, 31, 40 e 41.