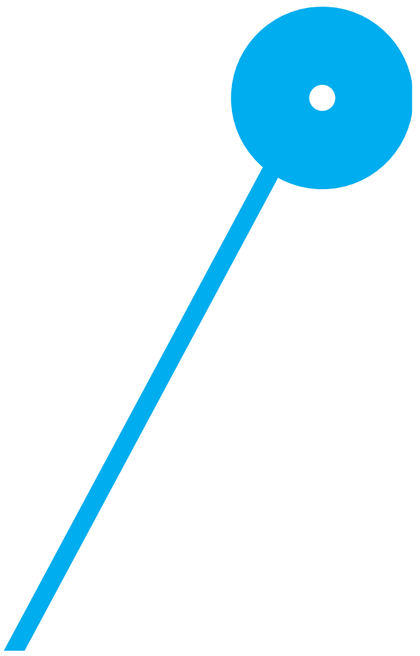
ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO
POLITÉCNICO
DO PORTO

P.PORTO

M

MESTRADO
ENGENHARIA INFORMÁTICA

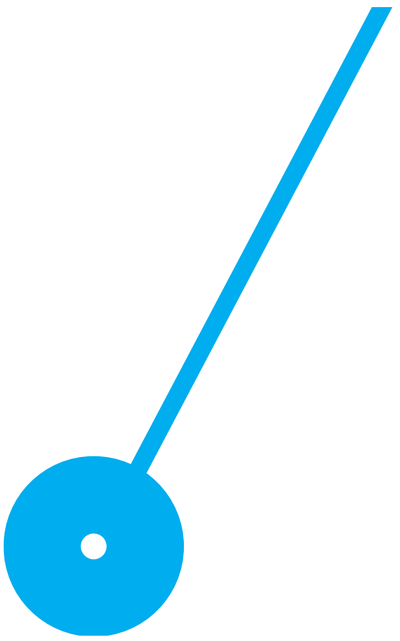


Deteção de fraude em *e-Sports*
Pedro Miguel Ferreira Xavier

12/2019

Pedro Miguel Ferreira Xavier. Deteção de fraude em *e-Sports*

P.PORTO



Deteção de fraude em *e-Sports*
Pedro Miguel Ferreira Xavier
12/2019
M
MESTRADO
ENGENHARIA INFORMÁTICA
ESCOLA
SUPERIOR
DE TECNOLOGIA
E GESTÃO
POLITÉCNICO
DO PORTO
P.PORTO

Agradecimentos

Gostaria de agradecer a todas as pessoas que de forma direta ou indireta tornaram este momento possível. Agradecer de forma muito particular ao professor Davide Carneiro por toda a ajuda e trabalho de orientação prestado ao longo desta enorme etapa, sem a sua preciosa ajuda este momento não era de todo possível, a ele o meu grande obrigado!

Agradecer também a toda a equipa da Performetric por todo o apoio e disponibilidade prestada ao longo desta etapa.

Gostava também de agradecer a todos os professores do Mestrado em Engenharia Informática da ESTG por toda a ajuda e por me tornarem um melhor profissional.

Agradecer igualmente à minha família e amigos por todo o apoio prestado nos momentos mais difíceis e por me incentivarem a continuar.

A todos, um grande obrigado!

Resumo

O mundo dos *e-Sports* está em constante mudança e crescimento, atualmente são vistos como uma tendência nos desportos cativando milhões de pessoas. Com o aumento da sua popularidade, dos jogadores e inclusive do interesse económico, é muito provável haver algum tipo de fraude (*cheating*) por causa do aumento de competitividade. Existem vários tipos de fraude e podem ser aplicados a vários níveis, o objetivo deste projeto consiste em desenvolver uma solução que permita detetar/evitar qualquer tipo de fraude em jogos online, através de uma identificação contínua do jogador. Neste sentido, foram desenvolvidas duas abordagens, uma através de meios estatísticos e outra através de *machine learning* para cumprir o objetivo proposto. As duas abordagens focam-se na interação do jogador, recorrendo à biometria comportamental para assim criar um modelo de jogador. Posteriormente estes modelos vão determinar se o jogador está a praticar algum tipo de fraude. Esta solução para além de não interferir com o fluxo do jogo, não necessita de qualquer alteração no *hardware* e não se foca em aspetos específicos do jogo. Com os resultados obtidos verificou-se que a abordagem de *machine learning* obteve bons resultados, revelando o potencial da aplicação e contribuindo assim para um ecossistema de jogo mais equilibrado.

Palavras-chave: *e-Sports*, Fraude, *Machine Learning*, Inteligência Artificial

Abstract

The e-Sports world is constantly changing and growing and is nowadays seen as a major online trend. Due to the rise of its popularity, the number of players involved and the economic impact it has, it is highly likely that some kind of fraudulent techniques will be used by players to gain a competitive advantage over their opponents. There are several types of fraud and they can be applied in several ways and this project aims to develop a solution to detect/avoid these frauds using players' continuous identification and validation to detect fraudulent behaviour. Two approaches were developed, one based on statistical methods and another based on machine learning, and both focus on the player interaction with the system using behavioural biometrics to detect patterns of fraudulent behaviour. These approaches do not interfere with the game flow and do not require any hardware changes to be implemented. They are also independent from the game logic being able to be used on any kind of sport. The tests made with the machine learning approach showed very positive results and we believe that it has the potential to make the online gaming ecosystem safer and more fair for everyone involved.

Keywords: e-Sports, Cheating, Machine Learning, Artificial Intelligence

Conteúdo

1 Introdução **9**

1.1 Solução Proposta 13

1.2 Objetivos 13

1.3 Performetric 14

1.4 Estrutura do documento 14

2 Contextualização **16**

2.1 Biometria comportamental 16

2.1.1 Casos de Uso 18

2.2 Interação Homem-Computador 19

2.2.1 Arquiteturas Típicas de HCI 21

2.2.2 Sistemas Multimodais 23

2.2.3 Aplicações de HCI 24

2.3 Inteligência ambiente 25

2.4 Computação Consciente do Contexto 27

2.5 Inteligência Artificial Explicável 28

2.5.1 Interpretabilidade 29

2.5.2 Escala de Interpretabilidade 29

2.6 Propriedades das explicações 30

2.7 *Real-time Analytics* 32

2.7.1 Bases de Dados NoSQL 34

3 Estado da Arte **37**

3.1 *Cheating* em *e-Sports* 37

3.2 Trabalho Relacionado 40

3.3 Análise Crítica 42

4 Arquitetura **44**

5 Identificação de Jogadores em *e-Sports* **47**

5.1 Dados 47

5.1.1 Eventos Base 49

5.1.2 *Feature Extraction* 50

5.2 Metodologias 55

5.2.1 IQR 55

5.2.2 *Machine learning* 61

5.3 API 67

5.4 Análise Crítica 70

6 Considerações Finais **72**

6.1 Conclusão 72

6.2 Trabalho futuro 73

Lista de Tabelas

4.1 Projeção do crescimento dos dados 46

5.1 Resultados para abordagem IQR 59

5.2 Resultados para abordagem de *machine learning* 66

Lista de Figuras

1 *Prize pool e-Sports* vs Desportos Tradicionais 11

2 *e-Sports* audiência **NAM** - *North America* **APAC** - *Asia-Pacific* **EU** - *Europe* Fonte: [52] 12

3 Características biométricas 17

4 Conceitos presentes na Inteligência Ambiente 26

5 Arquitetura da solução proposta 45

6 Exemplo *raw data* exportado da base de dados MongoDB 48

7 Excerto de um registo de interação de um jogador com o rato. 50

8 Distância em linha recta entre dois pontos no ecrã (s_{dist}) versus distância realmente percorrida pelo ponteiro (r_{dist}). 52

9 A distância média à linha recta é dada pela média do somatório das distâncias dos eventos MOVE à linha recta. A linha tracejada a vermelho representa a distância média a que o ponteiro viajou da linha recta. 53

10 Processo de cálculo da distância real percorrida pelo rato, calculada através do somatório da distância entre cada dois eventos MOVE consecutivos, denotados na imagem pelos pontos vermelhos. 54

11 Processo de cálculo da curva da trajetória do rato, calculada através do somatório do ângulo entre cada dois eventos MOVE consecutivos, denotados na imagem pelos pontos vermelhos. 54

12 Excerto do modelo do jogador *player1* 57

13 Interação do jogador 2. As linhas verde e vermelha representam os limites inferior e superior do seu modelo de interação, respetivamente, e a linha tracejada uma instância de interação aleatória. 58

14 *Key Down Time* por jogador (apenas 10 jogadores) 60

15 *Average Excess of Distance* por jogador (apenas 10 jogadores) 60

16 *Mouse velocity* por jogador (apenas 10 jogadores) 61

17 Interação do jogador *Player* 2 com o modelo do jogador *Player* 1 62

18 Exemplo do método K-Fold Cross Validation, com K=5 64

19 Importância das variáveis para o jogador 1 65

20 Importância das variáveis para o jogador 2 66

21 Importância das variáveis para o jogador 3 66

22 Resposta da API para o método *api/predict* 68

Pedido de Confidencialidade

Este documento, que descreve o trabalho realizado durante o Mestrado em Engenharia Informática pelo estudante Pedro Miguel Ferreira Xavier, desenvolvido na empresa Performetric, deve ser tratado com confidencialidade em virtude do sigilo do *know-how* tecnológico envolvido.

Glossário

SQL - *Structured Query Language*

NoSQL - *Non Structured Query Language*

RTP - Rádio e Televisão de Portugal

EUA - Estados Unidos da América

NBA - *National Basketball Association*

NAM - América do Norte, *North America*

APAC - *Asia-Pacific*

EU - Europa, *Europe*

HCI - Interação Homem-Computador, *Human Computer Interaction*

MMHCI - Interação Homem-Computador Multimodal, *Multimodal Human-Computer Interaction*

AmI - Inteligência Ambiente, *Ambient Intelligence*

AI - Inteligência Artificial, *Artificial Intelligence*

XAI - Inteligência Artificial Explicável, *Explainable Artificial Intelligence*

RDBMS - Sistema de Gestão de Base de Dados Relacional, *Relational Database Management System*

CRUD - Criar, Consultar, Atualizar e Apagar, *Create, Read, Update and Delete*

RAM - Memória de acesso aleatório, *Random Access Memory*

API - Interface de programação de aplicações, *Application Programming Interface*

IQR - Intervalo Interquartil, *Interquartile Range*

DRF - Floresta aleatória distribuída, *Distributed Random Forest*

KPI - Indicador-chave de desempenho, *Key Performance Indicator*

OLAP - Processamento analítico online, *Online Analytical Processing*

ETL - Extrair Transformar Carregar, *Extract Transform Load*

BIOS - Sistema Básico de Entrada/Saída, *Basic Input/Output System*

Capítulo 1

Introdução

Os *e-Sports* surgiram nos últimos anos como um novo desporto no qual os jogadores competem online, usando computadores, consolas ou outros dispositivos. Essas competições não ocorrem necessariamente num estádio ou em algum edifício (ex: pavilhão) e são vistas através de *streaming* de vídeo (ex: *Twitch*). Com o recente regulamento/normas, existem ligas profissionais, nas quais os jogadores ou equipas são pagos por patrocinadores e podem receber prémios significativos.

A indústria dos *e-Sports* tem registado um enorme crescimento ao longo dos anos, tanto em termos de audiência como de receitas. O crescimento da audiência é o que mais contribui para o crescimento das receitas. Com este aumento brevemente será um negócio de 1 mil milhão de dólares (aproximadamente: 871.785.000.000 €) e com uma audiência global a rondar os 400 milhões de espectadores. É uma área com um grande foco na atualidade tendo em conta a publicidade que é feita em torno dos principais torneios tanto a nível individual, dentro os quais se destacam:

- Fortnite World Cup Finals 2019 - Solo;
- Hearthstone World Championship 2019;
- FIFA eWorld Cup 2019 - Grand Final.

Como a nível colectivo (equipas):

- The International 2018;

- Fortnite World Cup Finals 2019 - Duo;
- LoL 2018 World Championship.

Se tivermos em conta os valores pagos aos jogadores (*prize pool*) nos torneios mencionados anteriormente (individual: $16.787.500 e colectivo: $47.082.177) podemos chegar à conclusão do porquê de estar tão na moda e ter cada vez mais jogadores. Se por outro lado, olharmos para a audiência, o torneio *The International 2018* contou com cerca de 55 milhões de espectadores espalhados por todo mundo.

Entretanto se recuarmos até ao momento do primeiro torneio de *e-Sports* (*The National Space Invaders Championship*) com o famoso jogo *Space Invaders* realizado em 1980, e que contou com mais de 10.000 participantes e com o vencedor a receber como prémio uma consola, vemos que os *e-Sports* cresceram imenso e com isto os respectivos *prize pools*. Hoje, os *e-Sports* estão a marcar uma nova era no desporto, apesar de estarem virados para uma vertente mais moderna da que normalmente é interpretada quando ouvimos a palavra desporto.

Podemos ver que cada vez mais equipas de desportos tradicionais tanto a nível nacional como exemplo o Sporting Clube de Portugal ou Vitória Sport Clube que criaram equipas de *e-Sports*, ou mesmo a RTP a criar uma plataforma online dedicada à transmissão e divulgação de campeonatos nacionais e internacionais denominado de RTP Arena. Olhando a nível internacional, equipas bem conhecidas da maior parte da população mundial na vertente de futebol como o Barcelona ou o Manchester United a formarem também as suas equipas de *e-Sports* para competirem.

O momento de explosão dos *e-Sports* acontece aquando do aparecimento das primeiras plataformas de *streaming* de jogos, como a *Twitch* (antiga *Justin.tv* que depois foi comprada pela Amazon em 2014) e que deu a liberdade a qualquer pessoa para fazer transmissões dos seus próprios jogos. Em 2013 os utilizadores desta plataforma tinham assistido a umas impressionantes 200 mil horas em vídeos/transmissões relacionados com jogos.

Outro grande momento foi em 2014, quando os EUA (Estados Unidos da América) reconheceram os jogadores de *e-Sports* como atletas profissionais. Este momento ajudou a indústria a crescer e tornou-se mais fácil operar pelos vários torneios organizados.

Posto isto, cada vez mais as grandes empresas (ex: Coca-Cola, Adidas, Vodafone) têm tendência a ligar-se a esta indústria por todas as vantagens que esta oferece a nível de divulgação de

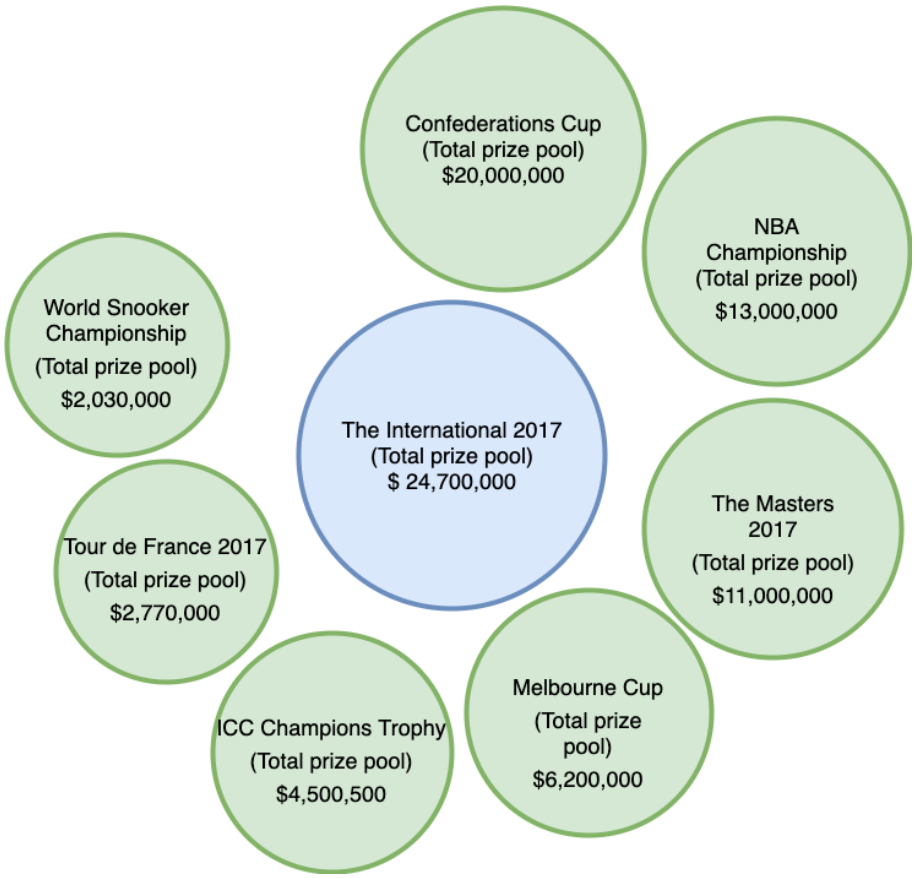


Figura 1: *Prize pool e-Sports* vs Desportos Tradicionais

marcas.

Com um aumento de todo um ambiente em redor dos *e-Sports* é com naturalidade que o *prize pool* dos torneios/competições também aumente, e em muitos casos é superior aos desportos tradicionais de grande renome que já existem à alguns anos, como é possível verificar na figura 1. Focando apenas no ano de 2017, um dos torneios mais rentáveis dos *e-Sports* apresenta um *prize pool* superior às competições mais conhecidas, como é o caso das finais da NBA (*NBA Championship*) que é um evento de grande renome à escala mundial, ou mesmo a Taça das Confederações (*Confederations Cup*) que tem sempre grande impacto onde quer que se realize.

Como acontece nos desportos tradicionais, a crescente notoriedade incentiva os jogadores e equipas a serem cada vez mais competitivos, às vezes recorrendo a comportamentos antidesportivos para ganhar alguma vantagem na competição. Nos desportos tradicionais, o tipo mais comum de fraude é o uso de substâncias para melhorar o desempenho. Nos *e-Sports*, existem novos meios para melhorar o desempenho, muitos dos quais são suportados ou possibilitados pelos dispositivos usados para competir. Por uma questão de justiça, é imperativo que os *e-*

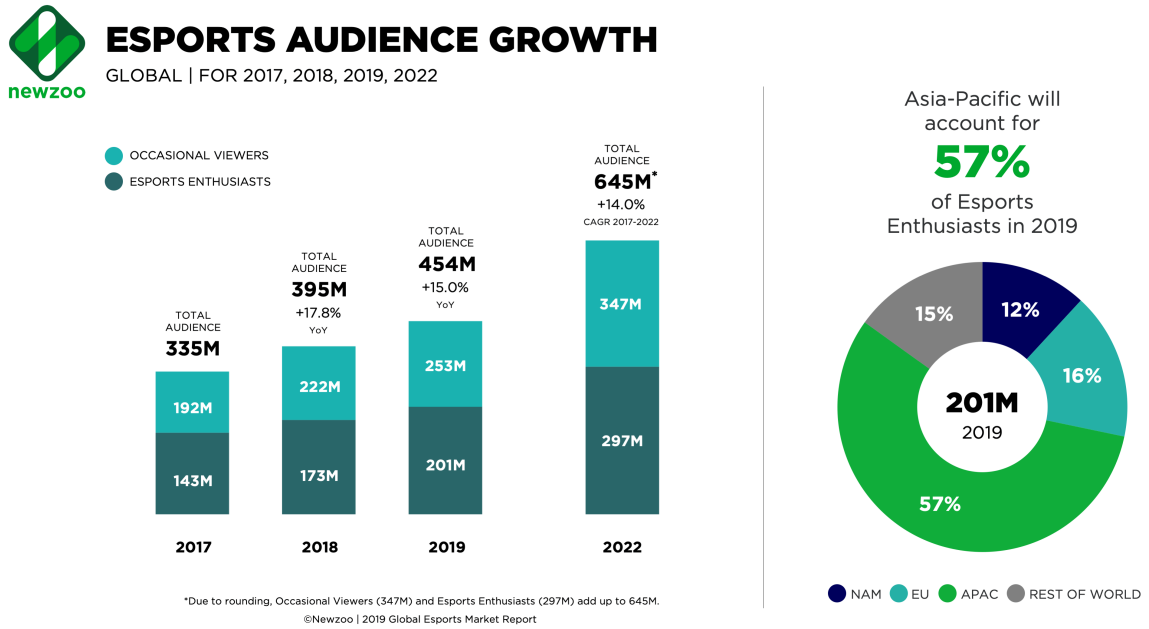


Figura 2: *e-Sports* audiência
NAM - *North America*
APAC - *Asia-Pacific*
EU - *Europe*
Fonte: [52]

Sports sejam inspeccionados como acontece com os desportos tradicionais, para assim evitar este tipo de comportamento. No entanto, as abordagens existentes não são adequadas, pois os *e-Sports* têm características muito distintas.

Cheating diminui o prazer do jogador participar num ambiente de jogo. Por exemplo, um jogador em particular que está a jogar um jogo online sem nenhum auxilio externo ilícito (ex: código alterado, *hacks*, etc) está em desvantagem em comparação com um jogador que tem auxilio. O jogador que não está a praticar *cheating* pode ser totalmente dominado, independentemente das suas habilidades individuais. Se o jogador que tem o comportamento correto for constantemente derrotado por jogadores que praticam *cheating*, e geralmente acontece de forma rápida e decisiva, o jogador que pratica um comportamento correto pode assim perder o interesse num jogo específico, numa rede de jogos específicos ou em empresas específicas. Esta perda de interesse afeta diretamente a comunidade responsável pelo jogo, fazendo com que cada vez menos jogadores utilizem os seus serviços. Como tal, existe um interesse inerente à comunidade, fornecedores de serviços e jogadores com comportamentos corretos, para identificar e reduzir tudo o que posso estar relacionado com *cheating* no ambiente de jogo.

1.1 Solução Proposta

Existem muitos tipos diferentes de *cheating* nos jogos online, incluindo *aimbots* (*bots* que disparam automaticamente mesmo que o adversário não esteja no raio de ação do jogador) ou *triggerbots* (um *bot* que dispara automaticamente quando um adversário está no seu raio de ação). Outro tipo de *cheating* acontece quando um jogador adultera a sua identidade, fazendo com que seja outro jogador a jogar por si. Isto é possível mesmo em competições oficiais, pois certas eliminatórias de qualificação não ocorrem num formato físico. Com este projeto procuramos evitar os vários tipos de *cheating* através do desenvolvimento de um método que tem como objetivo a identificação contínua dos jogadores de *e-Sports* e que o mesmo possa ser usado nos jogos e/ou competições, para que a fraude possa ser detectada e/ou evitada.

O método tem de ser tão abrangente como genérico. Abrangente no sentido de focar vários jogos e/ou competições e genérico para não se focar e não estar dependente de informações do jogo, o que iria tornar o método bastante complexo e quase impossível de gerir.

O método tem como foco o comportamento do jogador, recorrendo à biometria comportamental. Assim é possível criar um perfil/modelo para cada jogador sem recorrer a informações do jogo, usando apenas a interação do jogador com os periféricos e continuamente verificar se existe algum desvio em relação ao padrão do jogador, e caso exista, tomar as decisões necessárias.

1.2 Objetivos

O âmbito deste trabalho é, desenvolver um método para detetar/evitar fraude nos *e-Sports* e eventualmente ser utilizado nas competições profissionais, contribuindo para um ecossistema mais transparente e justo.

Para tornar esta solução possível, o método tira proveito de um aspeto que os métodos de *cheating* discutidos no capítulo seguinte têm em comum: a maneira como o jogador se comporta/interage no jogo. Ou seja, do ponto de vista de um observador externo, o *cheater* comporta-se de maneiras não naturais/humanas. Às vezes, essas mudanças são significativas ou evidentes o suficiente para serem detectadas por um observador humano. No entanto, existem técnicas mais subtis e mais difíceis de detetar. O método proposto usa *Big Data*,

estatística e *machine learning*.

O uso do método proposto na área dos *e-Sports*, principalmente em competições profissionais, pode aumentar significativamente a justiça, a transparência e a credibilidade de todo o processo. Consequentemente, aumentará a disposição dos patrocinadores e marcas de participarem e se associarem às equipas e jogadores de *e-Sports*, contribuindo para a evolução ética e sustentável da indústria.

1.3 Performetric

O projeto aqui apresentado foi desenvolvido para a empresa Performetric. A Performetric atua na área da fadiga mental e na análise em *real-time* da mesma. A fadiga mental é inevitável, e é muito difícil de reconhecer, determinar e gerir os nossos próprios níveis de fadiga. A fadiga é um fator importante, mas muitas vezes negligenciado, o que faz haver um acréscimo de erros e um decréscimo do desempenho. Muitos estudos demonstram que a fadiga mental leva à perda de eficiência, diminuição da produtividade no local de trabalho, deterioração das funções cognitivas e ao aparecimento de erros críticos no trabalho. As respostas das pessoas tornam-se mais lentas, mais variáveis e mais sujeitas a erros após o cansaço mental [65, 69, 21, 75].

A Performetric fornece duas aplicações, a primeira direcionada para os utilizadores comuns, pessoas que utilizam diariamente o computador para trabalho, com o intuito de perceber a fadiga de um utilizador e assim alertar quando os níveis de fadiga começam a aumentar. A segunda aplicação tem como foco os *e-Sports*, com o objetivo de gerir a fadiga para evitar um impacto negativo no desempenho e melhorar os resultados e a *perfomance* da equipa.

1.4 Estrutura do documento

A presente secção tem como objetivo explicar de uma forma clara qual é proposta, os seus objetivos e como o documento se encontra estruturado. No capítulo 2 é feito uma contextualização sobre as áreas importantes para a realização deste projeto. Em 2.1 é feito uma explicação do que é a biometria comportamental, como é importante para o projeto e quais as técnicas utilizadas. Seguido, na secção 2.2 é explicado em que consiste o HCI e como pode ajudar os utilizadores e consequentemente melhorar os ambientes envolventes. Na secção 2.3 é des-

crito o que é a Inteligência ambiente, quais os conceitos principais e quais as áreas a que se aplica. Seguido, na secção 2.4 é referido a importância do contexto e como os sistemas podem-se adaptar às actividades dos utilizadores. A secção 2.5 tem como foco a explicação dos modelos de *machine learning*. O capítulo é concluído com a secção 2.7 que tem em conta os dois lados da análise, dando maior foco à análise em tempo real, analisando alguns problemas e apresentando algumas ferramentas para análise em tempo real.

O capítulo 3 começa por abordar na secção 3.1 os tipos de *cheating* e em 3.2 é feito uma análise das soluções ou propostas de soluções existentes até à data. Termina em 3.3, com uma comparação entre as abordagens.

O capítulo 4 aborda a arquitetura implementada e como a mesma funciona nesta solução.

Posteriormente, no capítulo 5 é explicado como os dados de interação do jogador têm origem e quais são importantes para a solução, contendo ainda uma descrição dos mesmos. Aborda ainda as metodologias usadas e apresenta os resultados para as mesmas. É detalhada a API e os seus métodos, termina com uma análise entre as duas metodologias.

O capítulo que encerra este documento é o 6, aqui é feito uma revisão geral do projeto, se os objetivos foram cumpridos, o que podia ser melhorado e o trabalho futuro.

Capítulo 2

Contextualização

2.1 Biometria comportamental

Os sistemas biométricos reconhecem um humano com base nas suas características. Estas características dividem-se em dois grupos: físicas e comportamentais (figura 3).
A biometria física inclui impressão digital, reconhecimento facial, íris, entre outras características. Estas características biométricas fazem parte da pessoa e são únicas [19].
A biometria comportamental é uma características humana única que uma pessoa possui. Existem características que podem ser usadas para autenticar uma pessoa no mundo digital [67].

Biometria comportamental define um campo que extrai os recursos comportamentais de um utilizador [83]. A biometria tradicional usa características físicas ou fisiológicas humanas que são virtualmente únicas para cada indivíduo, incluindo impressões digitais, íris ou reconhecimento facial, impressão da palma, entre outras [9]. Estas características são usadas principalmente para fins de identificação. Por sua vez a biometria comportamental, conta com as características comportamentais do indivíduo. Embora a biometria comportamental também possa ser usada para fins de identificação, as características estão sujeitas a alterações de acordo com o estado da pessoa em determinado momento. Por exemplo, um indivíduo stressado pode mostrar diferenças significativas na fala, reduzindo a precisão da identificação.
Saber como uma pessoa geralmente se comporta permite detetar mudanças comportamentais significativas, que podem por sua vez indicar mudanças no estado da pessoa, indicando por exemplo, se a pessoa se encontra mais stressada. Algumas destas características estão

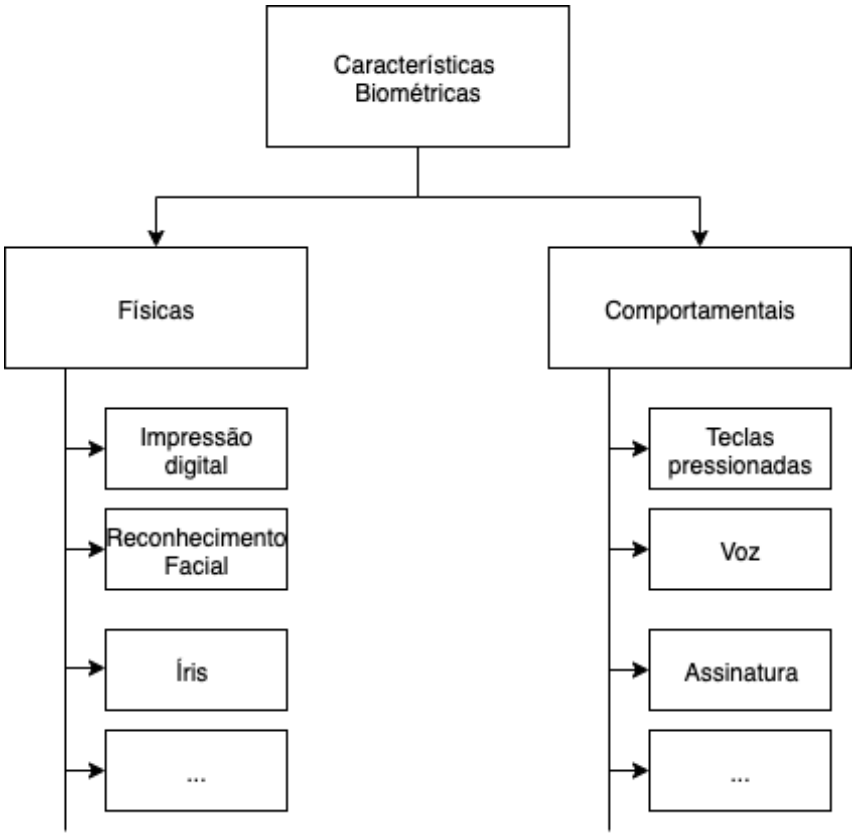


Figura 3: Características biométricas

presentes na figura 3.

A biometria comportamental tem sido usada essencialmente para fins de autenticação contínua de utilizadores. Existe uma variedade de subdivisões no domínio da biometria comportamental. Este capítulo vai focar-se em apenas duas delas, que foram usadas como única fonte de dados para a realização deste projeto:

- Dinâmicas de teclado;
- Dinâmicas de rato.

A dinâmica de teclado consiste em capturar e examinar o estilo e ritmo de escrita de um utilizador. O estilo de escrita de um utilizador pode ser utilizado como uma forma de ajudar a reconhecer o utilizador por detrás de determinada ação. Um estudo [59] define a dinâmica do teclado como uma importante técnica que pode ser usada para medir o comportamento e as dinâmicas habituais de escrita de um utilizador. A análise destas dinâmicas pode ser efetuada de duas maneiras [7]:

- Texto estático/fixo;
- Texto dinâmico/livre.

Muitos métodos foram usados para a autenticação baseada nas dinâmicas do teclado, com variação no uso de estatísticas a algoritmos mais complexos. Em muitos casos, uma combinação desses métodos também foi utilizada [5].

Outra pesquisa recente sobre autenticação baseada na dinâmica do teclado é realizada por [59], onde todo o processo de autenticação é divido em duas fases, fase de registo e fase de autenticação. Todo o processo é monitorizado por uma *webcam* e é classificado através do algoritmo de classificação *Bayesian*.

Esta dinâmica tem certas vantagens sobre os outros métodos de autenticação contínua [6]:

- Não há exigência de *hardware* adicional;
- Altamente aceitável pelos utilizadores, já que a interferência é mínima;
- Preserva a privacidade e segurança com base no comportamento;
- Não pode ser esquecido ou roubado.

A dinâmica do rato é uma abordagem biométrica projetada para capturar os movimentos estáticos e dinâmicos do uso do rato. Informações sobre o movimento do rato, como a mudança de posição do ponteiro ao longo do tempo e no espaço, tempo entre cliques, são registados, fornecendo uma base para construir um modelo de referência para o utilizador. Portanto a dinâmica do rato é baseada nos recursos de interação Homem-Computador (HCI) - como um utilizador interage com uma aplicação pode ser usado para o autenticar.

2.1.1 Casos de Uso

Neste campo, o rato e o teclado foram usados nos últimos anos como fonte de informações valiosas para a análise de padrões comportamentais, conhecidos respectivamente como dinâmicas de teclado/rato. Estas duas abordagens têm sido consistentemente usadas nos últimos anos para uma ampla gama de propósitos diferentes.

A biometria comportamental também tem sido usada para fins de identificação do utilizador. Existem vários estudos na área da investigação que podem ser apontados como dinâmica do rato para este fim específico. Recursos holísticos (estatísticas de clique único, estatísticas de clique duplo, tempo do movimento, etc) e recursos procedimentais (curva de velocidade, curva de aceleração, etc) para caracterizar o movimento do rato são usados em vários sistemas [71].

Os autores realizaram um estudo com 37 participantes, no qual foi obtido uma taxa de aceitação satisfatória com apenas 11,8 segundos de interação. Da mesma forma, [79] usaram 25 participantes e 5 recursos com o objetivo de modelar o ritmo do clique, que quantificam diferentes tempos entre cliques e durante cliques. Enquanto o estudo anterior usava principalmente o movimento do rato, este usa apenas o clique do rato. Estas duas abordagens poderiam ser usadas em conjunto, na tentativa de desenvolver uma abordagem mais precisa.

Outras abordagens foram analisadas por [35], com foco na autenticação através de abordagens baseadas na dinâmica do rato. Os autores também apresentam os resultados de vários testes realizados por os mesmos para ilustrar as observações e sugerir directrizes para avaliar futuras abordagens de autenticação com base na dinâmica do rato. Características biométricas também foram usadas para avaliar os níveis de stress e fadiga mental das pessoas. Nos últimos anos, vários estudos têm-se focado em descobrir padrões de interação das pessoas com os computadores e *smartphones*, para construir modelos que podem ser usados em tempo real para avaliar o stress e a fadiga do utilizador [62, 13].

2.2 Interação Homem-Computador

A interação Homem-Computador (*Human-Computer Interaction* - HCI) é um campo na área da investigação que estuda a interação entre os Homens e os computadores, de todas as formas e está particularmente envolvida com a compreensão do relacionamento entre os humanos e a tecnologia emergente [16]. O campo multidisciplinar [42, 8], originário do trabalho sobre os factores humanos em sistemas de computação na década de 1970, que hoje utiliza diversos métodos e práticas da Ciência da Computação, Psicologia, Sociologia, Design e Artes. Os conceitos centrais também foram formulados, tendo por base a noção de controlo e o design centrado no utilizador [72, 55].

A utilização de computadores sempre solicitou a questão da interface. Os métodos pelos quais

os humanos interagem com os computadores percorreram um longo caminho, caminho este que ainda continua e novos projetos e sistemas aparecem cada vez mais e as pesquisas nesta área têm crescido rapidamente nas últimas décadas.

O crescimento no campo HCI não está apenas na qualidade da interação, também sofreu modificações diferentes ao longo do tempo. Em vez de projetar interfaces regulares, as diferentes áreas de pesquisa tiveram focos diferentes nos conceitos aplicados, caso da multimodalidade em vez da unimodalidade, interfaces ativas em vez de passivas.

O conceito inicial era automaticamente representado com o aparecimento do computador, ou de uma forma mais geral, da máquina. A razão de facto é clara: a maioria das máquinas sofisticadas não serve de nada, a menos que possam ser usadas de forma adequada pelos Homens. Este argumento básico apresenta os principais termos que devem ser considerados no design do HCI: funcionalidade e usabilidade [84].

A funcionalidade de um sistema é definido pelo conjunto de ações ou serviços que este fornece aos utilizadores. No entanto, o valor da funcionalidade é visível apenas quando é possível ser utilizado com eficiência pelo utilizador [73]. Por sua vez a usabilidade com uma certa funcionalidade é o alcance pelo qual o sistema pode ser usado de forma eficiente e adequada para realizar certas metas para certos utilizadores. A eficácia real de um sistema é alcançada quando existe um equilíbrio adequado entre a funcionalidade e a usabilidade de um sistema [54].

HCI é atualmente uma parte importante do design de sistemas. A qualidade do sistema depende de como é representado e utilizado. Por essa razão, esta secção é dedicada a projetos de HCI. De facto, as nova directrizes de investigação e desenvolvimento apontam para a substituição dos métodos comuns de interação por métodos inteligentes, adaptáveis, multimodais e naturais, focando áreas como a Inteligência ambiente, realidade virtual, entre outras.

Tendo estes conceitos em mente, e que os termos computador, máquina e sistema são frequentemente usados de forma intercambiável neste contexto, o HCI deve-se ajustar entre o utilizador, o computador e os serviços necessários, a fim de alcançar determinado desempenho tanto na qualidade como na organização dos serviços [78].

2.2.1 Arquiteturas Típicas de HCI

O fator mais importante é a configuração. De facto, qualquer interface é definida pelo número de *inputs* e *outputs*. A arquitetura de um sistema de HCI mostra quais são estes inputs e outputs e como funcionam juntos. As próximas secções explicam as diferentes configurações e *designs* nos quais uma interface é baseada.

2.2.1.1 Sistemas Unimodais

Como mencionado anteriormente, uma interface depende principalmente do número e diversidade de *inputs* e *outputs*, que são canais de comunicação que permitem ao utilizador interagir com o computador. Cada um dos diferentes canais é denominado de modalidade [33]. Um sistema baseado apenas numa modalidade é chamado de unimodal. Com base na natureza das diferentes modalidades, elas podem ser divididas em três categorias:

- *Visual-based*;
- *Audio-based*;
- *Sensor-based*.

As próximas subsecções descrevem cada categoria e fornecem exemplos e referências à modalidade.

Visual-based

A interação visual com *human-computer* é provavelmente a área mais difundida nos sistemas HCI. Considerando a variedade das aplicações, problemas e abordagens em aberto, os investigadores tentaram abordar diferentes aspetos das respostas humanas que podem ser reconhecidos como um sinal visual. Algumas das principais áreas de pesquisa são:

- Análise da expressão facial;
- Rastrear o movimento corporal;

- Reconhecimento de gestos;
- Deteção do olhar (rastrear o movimento dos olhos).

Embora o objetivo de cada área seja diferente devido às aplicações, pode concluir-se uma concepção geral para cada área. A análise da expressão facial geralmente lida com o reconhecimento facial [20, 26, 58]. O rastreamento dos movimentos corporais [28, 3] e o reconhecimento de gestos [37, 82, 43] são o principal foco desta área e podem ter objetivos diferentes, mas na maioria das vezes são usados para a interação direta de humanos e computadores. A deteção de olhar [74] é uma forma de indireta de interação entre o utilizador e as máquinas, usada principalmente para uma melhor compreensão de atenção, intenção ou foco do utilizador em contextos sensíveis.

Audio-based

A interação baseada em áudio entre um computador e um humano é outra área importante dos sistemas HCI. Esta área lida com informações adquiridas por diferentes sinais de áudio. Embora a natureza dos sinais de áudio possa não ser tão variável quanto os sinais visuais, as informações recolhidas nos sinais de áudio podem ser mais confiáveis, úteis e em alguns casos indicadores exclusivos de informações. As áreas de pesquisa nesta secção podem ser divididas nas seguintes partes:

- Reconhecimento de voz;
- Análise da emoção auditiva;
- Deteção de ruído;
- Interação musical.

Historicamente, o reconhecimento de voz [63, 12] tem sido o principal foco dos investigadores. Um dos grandes objetivos é integrar emoções através do áudio [61, 18]. A geração e interação da música é uma área muito nova nos domínio HCI, com aplicações na indústria da arte, que são estudadas em sistemas HCI baseados em áudio e visual [47].

Sensor-based

Esta secção é uma combinação de diversas áreas com uma ampla gama de aplicações. O ponto comum destas diferentes áreas é que pelo menos um sensor físico é usado entre o utilizador e a máquina para existir a interação.

- Teclado e rato;
- Sensores hápticos;
- Sensores de movimento;
- Sensores de pressão.

Alguns dos sensores descritos em cima existem há algum tempo, outros são tecnologias novas. Os sensores de movimento são uma tecnologia de ponta que revolucionou a indústria cinematográfica, artes e jogos. Os sensores hápticos e de pressão são de especial interesse para aplicações no ramo da robótica e da realidade virtual [66, 31, 32]. Novos robôs, "humanóides", incluem centenas de sensores hápticos que tornam os mesmos sensíveis ao toque [29, 38]. Estes tipos de sensores também são utilizados em cirurgias médicas [11].

2.2.2 Sistemas Multimodais

O termo multimodal refere-se à combinação de múltiplas modalidades. Nos sistemas MMHCI (*Multimodal Human-Computer Interaction*), estas modalidades referem-se principalmente às formas como o sistema responde às entradas, ou seja, canais de comunicação [33].

As definições destes canais são na prática os sentidos humanos: visão, audição, paladar, olfato e tato. Portanto, uma interface multimodal atua de forma a simplificar a interação Homem-Computador, através de dois ou mais modos de entradas que vão além do teclado e do rato. As interfaces multimodais incorporam diferentes combinações de fala, gesto, olhar, expressões faciais, entre outros. Uma das combinações frequentes é o gesto e a fala [57].

Embora o sistema ideal de MMHCI deva contar com uma combinação de modalidades únicas que interajam de forma correlativa, o mesmo é difícil de realizar devido aos limites práticos

e aos problemas que possam existir em cada modalidade que não permitem a fusão das diferentes modalidades. Apesar de todos os progressos realizados na maioria dos sistemas existentes, as modalidades ainda são tratadas separadamente e, somente no final, os resultados das diferentes modalidades são finalmente juntos com os restantes.

A razão é que os problemas em aberto em cada área ainda precisam de ser resolvidos o que significa que ainda há trabalho a ser feito para adquirir uma ferramenta confiável para cada sub-área. Além disso, os papéis das diferentes modalidades e a sua participação na interação não são cientificamente conhecidos. "No entanto, as pessoas transmitem sinais comunicativos multimodais de maneira complementar e redundante. Portanto, para realizar uma análise multimodal com múltiplos sinais de entrada e adquiridos por diferentes sensores, os sinais não podem ser considerados mutuamente independentes e não podem ser combinados de maneira livre de contexto no final da análise pretendida, mas pelo contrário, os dados de entrada devem ser processados num espaço de recursos conjuntos e de acordo com um modelo dependente do contexto. No entanto, além dos problemas de deteção de contexto e desenvolvimento de modelos dependentes de contexto para combinar informações multissensoriais, deve-se lidar com o tamanho do espaço necessário para os recursos conjuntos. Os problemas incluem grande dimensão, diferentes formatos de recursos e alinhamento de tempo" [33].

Um aspeto interessante da multimodalidade é a colaboração de diferentes modalidades para auxiliar os reconhecimentos. Por exemplo, o rastreamento do movimento labial (*visual-based*) pode ajudar a melhorar os métodos de reconhecimento de fala (*audio-based*).

2.2.3 Aplicações de HCI

Um exemplo clássico de um sistema multimodal é o sistema de demonstração "*Put That There*" [10]. Este sistema permite mover um objeto para um novo local num mapa dizendo apenas "*put that there*" enquanto apontamos para o objeto que queremos trocar e depois para o local desejado. Interfaces multimodais têm sido usadas em diversas aplicações, como quiosque de informação, como o *MATCHKiosk* da *AT&T* [34] e sistemas de autenticação biométrica [57].

As interfaces multimodais podem oferecer várias vantagens sobre as interfaces tradicionais. Por um lado, as interfaces multimodais oferecem uma experiência mais natural e mais fácil de usar. Por exemplo, um sistema imobiliário denominado *Real Hunter* [15] tem várias funcionalidades, uma das quais a utilização de gestos e fala, para escolher as casas e fazer perguntas

sobre uma casa em particular. A utilização de gestos para selecionar um objeto, no caso uma casa e utilizar a fala para retirar mais informações da mesma permite uma experiência mais natural. Outro exemplo, é o caso da *MATCHKiosk* que também permite o uso da fala e da escrita. Assim, num ambiente barulhento, pode-se fornecer informações por meio escrito para especificar a nossa procura. Outros exemplos de sistemas multimodais estão listados abaixo:

- *Smart Video Conferencing* [48];
- *Intelligent Homes/Offices* [49];
- *Intelligent Games* [68].

2.3 Inteligência ambiente

A inteligência ambiente (*Ambient Intelligence* - AmI) é uma área multidisciplinar com o objetivo de trazer inteligência aos nossos ambientes quotidianos e tornar os mesmos sensíveis a nós, desenvolvendo espaços inteligentes que se adaptam aos interesses, necessidades e desejos das pessoas que utilizam estes ambientes, ajudando-os a executar tarefas referentes ao seu quotidiano [23].

O conceito de inteligência ambiente fornece uma visão onde as pessoas estão rodeadas de tecnologia e assim identificar as pessoas presentes no meio, através da recolha de informação sobre as suas ações, emoções, entre outras, de modo tornar a interação cada vez melhor.

A palavra **ambiente** refere-se à necessidade de envolver a tecnologia de uma forma não invasiva nos objetos e ambientes quotidianos [1].

Por sua vez, a palavra **inteligência** reflete que os ambientes devem poder reconhecer pessoas, personalizar as preferências das mesmas e adaptar-se às pessoas (utilizadores) [1].

Para melhor a compreensão do que a inteligência ambiente prioriza os conceitos presentes na figura 4: **Computação Ubíqua**, **Computação Pervasiva**, **Computação do Contexto** e **Sistemas Embutidos** são fundamentais.

Computação Ubíqua significa que temos acesso a dispositivos de computação em qualquer lugar de maneira integrada e coerente.

Figura 4: Conceitos presentes na Inteligência Ambiente

Computação Pervasiva corresponde à parte física, como telemóveis, computadores ou mesmo relógios inteligentes.

Computação do Contexto significa que o sistema tem consciência da situação presente e é capaz de tomar uma decisão de acordo com a situação atual. A indústria automóvel tem investido nestes sistemas para a deteção da probabilidade de acidentes. É um dos conceitos mais desejados de incluir no AmI, a identificação do contexto é muito importante para decidir/agir de forma mais inteligente.

Sistemas Embutidos são dispositivos eletrônicos e computacionais que são incorporados dentro de objetos ou bens. Atualmente os bens como os carros, electrodomésticos são equipados com microprocessadores. AmI e sistemas embutidos são facilmente distinguidos visto que em cenários de ambiente inteligente os dispositivos podem estar visíveis.

Começa a ficar claro que a inteligência deve ser fornecida aos ambientes usados diariamente. Em áreas como casas, veículos, fábricas e até mesmo cidades inteligentes é necessário estar presente um agente inteligente. Um agente que seja capaz de recolher e processar dados de forma a melhorar a tomada de decisão com base em experiências passadas e sem esquecer

o contexto atual.

Com isto podemos afirmar que a inteligência ambiente não é possível sem a inteligência artificial. Por outro lado os investigadores da área da inteligência artificial devem estar cientes da necessidade de integrar com as mais diversas áreas (automação, computação gráfica, comunicação) [51].

2.4 Computação Consciente do Contexto

O termo **contexto** tem despertado um enorme interesse nas mais diversas áreas das ciências da computação. As circunstâncias em que um determinado evento acontece é conhecido como contexto. O contexto permite que um indivíduo entenda e interprete corretamente um acontecimento. A importância do contexto vai mesmo ao ponto de moldar quem nós somos enquanto indivíduos. Por estas razões não é estranho um indivíduo exibir diferentes ações em diferentes contextos [25].

Os sistemas com computação consciente têm um grande propósito em áreas como **Inteligência Ambiente**, **Computação Persuasiva**, **Computação Ubíqua**, **Inteligência Artificial**. Estes sistemas recolhem informações do contexto, podendo este ser físico, computacional ou mesmo as tarefas de um utilizador para então conseguir adaptar cada vez mais o sistema ao utilizador.

Muitos investigadores definiram o **contexto** de acordo com as suas ideias, com um esforço para tornar o conceito o mais geral possível. Schilit e Theimer descreveram o contexto em 1994 como identidades, locais, objetos e pessoas próximas [70]. Em 1996, Brown definiu o contexto através dos elementos que rodeiam um utilizador e que um computador consegue identificar [2]. Uma definição de contexto muitas vezes citada e bastante genérica é a de Dey e Abowd: “Contexto é qualquer informação que possa ser usada para caracterizar a situação de uma entidade. Uma entidade é uma pessoa, local ou objeto, que é considerado relevante para a interação entre um utilizador e uma aplicação, incluindo o utilizador e a aplicação” [70, 41].

O termo contexto foi classificado em duas categorias:

- Físico;

- Lógico.

O contexto físico pode ser determinado por sensores ou *hardware*. Contexto lógico é fornecido através da interação do utilizador com os serviços disponíveis [4, 77].

Os sistemas sensíveis ao contexto são capazes de se adaptar às atividades dos utilizadores com base no seu contexto atual, sendo este um local, objeto, etc. Estes sistemas observam constantemente o ambiente e propõem sugestões adequadas aos utilizadores para que eles possam tomar as ações necessárias.

A tendência futura destes sistemas será baseada no contexto social. Este contexto não se deve focar apenas nas interacções entre utilizadores, mas também entre as máquinas e entre pessoas as e máquinas (IoT e IoE) [27].

2.5 Inteligência Artificial Explicável

Nos últimos anos, a inteligência artificial (AI) alcançou um momento notável que, se aproveitado adequadamente, pode oferecer melhores expectativas em vários setores. Para que este passo seja dado em breve é preciso passar a barreira da explicabilidade. Os sistemas de *machine learning* são usados para tomar decisões importantes sobre a vida das pessoas, como o emprego, finanças, empréstimos [45]. Estas mesmas áreas têm de perceber o domínio, o modelo, e a forma como ele funciona, para assim perceber as ações que o modelo toma e assim conseguir explicar as mesmas. Por exemplo, uma pessoa vê o seu pedido de empréstimo rejeitado, a organização deve ser capaz de explicar o porquê de o mesmo ter sido rejeitado. Caso contrário é impensável recorrer a estes sistemas pois não conseguimos entender as ações tomadas.
Atualmente a maioria dos sistemas de inteligência artifical e *machine learning* limitam-se a produzir uma decisão sem realmente explicar o porquê. Isto é um problema que abrange várias áreas e que se enquadram na Inteligência Artificial Explicável (*Explainable Artificial Intelligence - XAI*), que é amplamente reconhecido como um recurso crucial para a implementação prática de inteligência artifical, com foco principal no público alvo para qual a explicabilidade é procurada. Cada vez mais os sistemas de inteligência artificial recebem mais responsabilidades e tarefas mais complexas e é necessário entender quando podemos confiar nos resultados. Estes sistemas não são perfeitos e consequentemente têm falhas, o importante é saber porque

falhou para assim perceber melhor o domínio.

2.5.1 Interpretabilidade

Não há definição matemática de interpretabilidade. Uma definição não matemática, que pode ser utilizada é: interpretabilidade é o grau em que o ser humano pode entender a causa da decisão [50], outra definição: interpretabilidade é o grau em que um humano pode prever consistentemente o resultado do modelo [40]. Quanto maior a interpretabilidade de um modelo de *machine learning*, mais fácil é para alguém compreender o porquê de determinadas decisões e previsões serem feitas. Um modelo é mais interpretável que outro, se as decisões de um modelo forem mais fáceis de entender para um humano do que as decisões do outro modelo.

Se um modelo de *machine learning* tem um bom desempenho, porque não confiamos no modelo e ignoramos o facto de ele tomar determinada decisão? "O problema é que uma métrica única, como a previsão da classificação, é uma descrição incompleta da maioria das tarefas do mundo real." [22].

Mas "porquê" que a interpretabilidade é tão importante? Quando se trata de uma abordagem preditiva, é preciso inverter os papéis: queremos saber o que é previsto? Ou queremos saber o porquê da previsão feita?
Em alguns casos não precisamos de saber o porquê de determinada decisão ser tomada, basta saber que o desempenho foi bom num conjunto de testes. Mas, em outros casos, conhecer o "porquê" pode ajudar a conhecer melhor o problema, os dados, e o motivo pelo qual o modelo pode falhar. Alguns modelos podem não exigir explicações porque são usados em ambientes de baixo risco (exemplo de um sistema de recomendações). A necessidade da interpretabilidade surge quando existe uma dificuldade na formalização de problemas [22], o que significa que, para certos problemas, não é suficiente obter a previsão (*the what*). O modelo também deve explicar como chegou à previsão (*the why*) porque uma previsão correcta só resolve parcialmente o problema.

2.5.2 Escala de Interpretabilidade

A complexidade de um modelo de *machine learning* está diretamente relacionado com a sua interpretabilidade. Geralmente, quanto mais complexo o modelo mais difícil é interpretar e

explicar. O número de pesos e a dimensão, são boas formas de quantificar a complexidade de um modelo. No entanto, analisar a forma funcional de um modelo é particularmente útil em algumas aplicações, como *credit scoring*.

De acordo com [22], podemos avaliar a interpretabilidade de um modelo a três níveis:

Avaliação ao nível da aplicação (*real task*): A explicação é colocada no produto e o teste final é feito com um utilizador.

Avaliação ao nível humano (*simple task*): É uma avaliação simplificada ao nível da aplicação. A grande diferença é que os testes não são realizados por especialistas na área mas sim com leigos. Isto torna os testes mais baratos porque já não é necessário encontrar especialistas e é mais fácil encontrar pessoas para testar.

Avaliação ao nível de função (*proxy task*): Não requer humanos, tem um melhor aproveitamento quando a classe usada pelo modelo já foi avaliada por outra pessoa numa avaliação ao nível humano.

2.6 Propriedades das explicações

Propriedades dos métodos de explicação

Translucidez: descreve quando um método de explicação se baseia na análise do modelo de *machine learning* (exemplo: parâmetros). Por exemplo, métodos de explicação baseados em modelos intrinsecamente interpretáveis, como o modelo de regressão linear (específico do modelo), são altamente translúcidos. Os métodos que dependem apenas da manipulação de entradas e da observação das previsões têm zero translucidez. Dependendo do cenário, diferentes níveis de translucidez podem ser desejáveis. A vantagem da alta translucidez é que o método pode contar com mais informações para gerar explicações. A vantagem da baixa translucidez é que o método de explicação é mais portátil.

Portabilidade: descreve a variedade de modelos de *machine learning* com os quais o método de explicação pode ser usado. Métodos com baixa translucidez têm maior portabilidade porque tratam o modelo de *machine learning* como uma caixa preta. Modelos substitutos podem ser o método de explicação com maior portabilidade.

Complexidade Algorítmica: descreve a complexidade computacional do método que gera a explicação. É importante considerar esta propriedade quando o tempo de computação é um *bottleneck* na geração de explicações.

Propriedades das explicações individuais

Precisão: até que ponto uma explicação prevê dados que não foram vistos? Uma precisão elevada é especialmente importante se a explicação for usada para previsões no lugar do modelo de *machine learning*. A baixa precisão pode ser boa se o modelo se basear em explicar o que um modelo de caixa preta faz. Neste caso, apenas fidelidade é importante.

Fidelidade: até que ponto a explicação se aproxima da previsão do modelo de caixa preta? Alta fidelidade é uma das propriedades mais importantes de uma explicação, porque uma explicação com baixa fidelidade é inútil para explicar o modelo. Precisão e fidelidade estão profundamente relacionados. Se o modelo de caixa preta tiver alta precisão e a explicação tiver alta fidelidade, a explicação também terá alta precisão.

Consistência: quanto uma explicação difere entre os modelos que foram treinados na mesma tarefa e que têm previsões semelhantes? Se as explicações são semelhantes, as explicações são altamente consistentes.

Estabilidade: quão semelhantes são as explicações para instâncias semelhantes? Enquanto a consistência compara explicações entre modelos, a estabilidade compara explicações entre instâncias semelhantes para um modelo. Alta estabilidade significa que pequenas variações nos atributos de uma instância não alteram substancialmente a explicação (a menos que essas pequenas variações alterem drasticamente a precisão). Uma falta de estabilidade pode ser o resultado de uma alta variação do método de explicação. Alta estabilidade é sempre desejável.

Compreensibilidade: até que ponto os humanos entendem as explicações? Difícil de definir e medir, mas extremamente importante acertar. Muitas pessoas concordam que a compreensibilidade depende da audiência. A compreensibilidade dos recursos usados na explicação também deve ser considerada. Uma transformação complexa de atributos pode ser menos compreensível que os atributos originais.

Novidade: a explicação reflete se uma instância de dados vem de uma ”nova”região, fora da

distribuição dos dados de treino do modelo? Nestes casos, o modelo pode ser impreciso e a explicação inútil. O conceito de novidade está relacionado ao conceito de certeza. Quanto maior a novidade, maior a probabilidade de o modelo ter baixa certeza devido à falta de dados.

2.7 *Real-time Analytics*

No espectro da análise, existem dois extremos que podem ser definidos. Numa extremidade, existem aplicações analíticas em *batch*, usadas para análises mais complexas e de longa duração. Por norma, estas aplicações têm tempos de reposta mais lentos (horas ou dias) e requisitos mais baixos de disponibilidade. Uma *framework* muito utilizada para a execução de aplicações analíticas em *batch* é o Hadoop.

No outro extremo do espectro, estão aplicações analíticas em tempo real. O tempo real pode ser considerado a nível dos dados ou a nível do utilizador final. O primeiro refere-se à capacidade de processar dados com baixa latência (processar uma grande quantidade de dados com os resultados disponíveis para o utilizador final quase em tempo real), possibilitando, por exemplo, fornecer recomendações para um utilizador com base no histórico de pesquisas. O segundo traduz-se na capacidade de processar dados à medida que chegam, possibilitando a agregação dos dados e a extração de tendências sobre a situação atual do sistema (*streaming analytics*).

Em relação ao fluxo de processamento de dados, os principais problemas estão relacionados com:

- *Sampling Filtering*;
- *Correlation*;
- *Estimating Cardinality*;
- *Estimating Quantiles*;
- *Estimating Moments*;
- *Finding Frequent Elements*;
- *Counting Inversions*;

- *Finding Subsequences*;
- *Path Analysis*;
- *Anomaly Detection Temporal Pattern Analysis*;
- *Data Prediction*;
- *Clustering*;
- *Graph analysis*;
- *Basic Counting*;
- *Significant Counting*.

As principais aplicações são testes A/B, deteção de fraude, análise de rede, análise de tráfego, redes de sensores e análise de imagens médicas [36].

De acordo com [36], as ferramentas apresentadas abaixo são as mais conhecidos para *streaming* e são *open source*:

S4 Análise em tempo real com um modelo baseado em *key-value*;

Storm A plataforma de análise em tempo real mais popular e desenvolvida no Twitter;

Millwheel Estrutura de análise em tempo real do Google, que fornece uma semântica única;

Samza Estrutura para análise em tempo real sem topologia, que realça a partilha entre grupos;

Akka *Toolkit* para desenvolver aplicações distruídas, concorrentes e tolerantes a falhas;

Spark Faz análises offline e online usando o mesmo código e o mesmo sistema;

Flink Faz a junção da análise online e offline usando técnicas de RDBMS(*Relational Database Management System*)

Pulsar Faz análises em tempo real usando SQL (*Structured Query Language*)

Heron Storm modernizado com maior foco na escalabilidade e *debugging*

As análise online, por outro lado, são desenhadas para fornecer análises de menor peso muito rapidamente. Os requisitos deste tipo de análise são a baixa latência e alta disponibilidade. Na era do *Big Data*, os processos *OLAP* (*Online analytical processing* [17]) e *ETL* (*Extract, Transform, Load*) são muito caros. Particularmente, a heterogeneidade das fontes de dados dificulta a definição de um *schema*, dificultando a visão direcionada ao modelo. Neste paradigma, as análises são necessárias quase em tempo real para dar suporte as aplicações e aos seus utilizadores.

2.7.1 Bases de Dados NoSQL

Um elemento essencial num ecossistema de processamento de dados, seja ele em *real-time* ou em *batch*, é a base de dados em que os dados são armazenados.

As base de dados relacionais provaram ser altamente eficientes, confiáveis e consistentes em termos de armazenamento e processamento de dados estruturados [39]. Contudo, em relação aos 3V's (Volume, Velocidade e Variedade), o modelo relacional apresenta várias falhas. Empresas como a Amazon, Facebook e Google começaram a trabalhar nos seus próprios mecanismos de dados para lidar com o *Big Data*, e esta tendência inspirou outros a fazerem o mesmo, mas em formato *open-source* e para outros casos de uso. Os principais motivos para a mudança para uma base de dados NoSQL são problemas de desempenho e flexibilidade [76].

Pode ser estabelecido um mapeamento entre as características de *Big Data* (os 3 V's) e os recursos NoSQL. As base de dados NoSQL podem gerir grande volumes de dados, permitindo o particionamento de dados em muitos nós de armazenamento e estruturas virtuais, superando as restrições de infraestructura tradicional. Ao comprometer as propriedades do ACID (Atomicidade, Consistência, Isolamento, Durabilidade garantida pelo RDBMS nas transações da base de dados) as propriedades NoSQL abrem caminho para menos bloqueios entre as consultas de dados dos utilizadores.

De acordo com [14], as principais características que fazem parte dos sistemas NoSQL são: a capacidade de dimensionar horizontalmente as operações CRUD em vários servidores, capacidade de replicar e distribuir dados por vários servidores, interface ou protocolo simples a nível de chamada (em contraste com uma ligação SQL), uso eficiente de índices distribuídos e RAM para armazenamento de dados e capacidade de dinamicamente adicionar novos atributos ao

dados.

No entanto, os sistemas diferem em muitos pontos. As base de dados NoSQL permitem acomodar uma variedade de dados presentes em problemas reais.

As base de dados NoSQL podem-se dividir em 4 categorias:

Key-value Stores Uma base de dados chave-valor apenas pode executar duas operações: armazenar pares de chaves e valores e recuperar os valor armazenados que recebem essa chave. Estes tipos de sistema são adequados para aplicações com modelos de dados simples que exigem um armazenamento de dados eficiente. Redis e Memcached são exemplos deste tipo populares deste tipo de base de dados.

Document-oriented Databases Este tipo de armazenamento de dados é projetado para armazenar e gerir documentos. Permite documentos ou listas e como valores e os nomes dos atributos são definidos dinamicamente para cada documento no tempo de execução. Uma única coluna pode conter centenas de atributos (em analogia ao modelo relacional), e o número e o tipo de atributos podem variar de uma lista para a outra, desde que o *schema* seja livre. Este tipo de armazenamento permite pesquisas de chaves e valores. MongoDB, CouchDB pertencem a este tipo de armazenamento.

Column-oriented Databases As bases de dados orientadas a colunas são o tipo de armazenamento de dados que mais se assemelha ao modelo relacional em um nível conceptual. Existe os conceitos de tabelas, linhas e colunas, criando assim a noção de um *schema*. Nesta abordagem, as linhas são divididas entre nós por meio de *sharding* na chave primária. Geralmente são divididas através de um *range* e não de uma *hash*, ou seja, em consultas sobre intervalo de valores não é preciso percorrer todos os nós. As linhas são agrupadas em coleções (tabelas) e os atributos de uma linha individual podem ser de qualquer tipo. Apache Cassandra e Apache HBase são exemplos desse tipo de armazenamento de dados.

Graph-oriented Databases As base de dados orientada a grafos possuem nós e arestas. Os nós são entidades de domínio de dados e as arestas são o relacionamento entre duas entidades. Os nós podem ter propriedades ou atributos. O Neo4J é o banco de dados orientado a gráficos mais popular.

Como foi apresentado, existem várias opções na hora de escolher uma base de dados NoSQL,

e as diferentes categorias e arquiteturas servem a propósitos também eles diferentes. Em relação as consultas complexas, os sistemas de armazenamento de dados orientados a colunas e orientados a documentos são mais adequados do que os armazenamentos chaves-valores e base de dados orientada a grafos. Considerando este último facto, é apresentada uma comparação em [46], onde vários sistemas são classificados numa escala de 5 pontos (ótimo, bom, médio, medíocre, péssimo) em relação a um conjunto de atributos.

Pelas características mencionadas anteriormente, neste trabalho é utilizada uma base de dados não relacional, na qual são armazenados os dados recolhidos da aplicação cliente.

Capítulo 3

Estado da Arte

3.1 *Cheating* em *e-Sports*

Existem vários tipos diferentes de *cheating* nos *e-Sports*, que podem ser resumidos em dois grupos:

- *Cheating* assistido por *software*;
- Jogo antidesportivo.

Cheating assistido por *software* inclui métodos que usam *software* específico no dispositivo do jogador para melhorar uma certa habilidade relevante para o desempenho do jogador no jogo. Os parágrafos a seguir descrevem os métodos mais comuns de *cheating* assistido por *software*.

Aimbot é um método que se baseia nas informações recebidas pelo aparelho do jogador sobre os locais dos adversários. Embora essas informações sejam necessárias para o jogo e todos os seus componentes, também podem ser exploradas por um *software* para atingir continuamente um determinado adversário de uma maneira tão precisa que seria impossível um jogador humano conseguir. Isto pode ser realizado quando o adversário está atrás de qualquer obstáculo (como uma parede), permitindo que o jogador atinja o adversário independentemente da distância e dos obstáculos que possam estar entre o jogador e o adversário.

Triggerbot é um método para disparar automaticamente quando um adversário aparece na

mira do jogador. Isso leva o jogador a exibir tempos de reação que seriam impossíveis para um humano.

O atraso artificial é um método que consiste em reduzir ou interromper temporariamente o fluxo de dados entre o jogador e o servidor. O principal objetivo desse método é fazer com que o avatar do jogador se comporte de maneira irregular e imprevisível. O jogador é, portanto, capaz de alterar o fluxo de saída dos seus dados, enquanto ainda recebe os dados dos seus adversários. Isto dá ao jogador uma vantagem sobre os seus adversários. Este método pode ser implementado tanto a nível de *hardware* como de *software*.

O *look-ahead* é um método no qual o jogador obtém uma vantagem sobre os adversários adiando as suas próprias ações para que as ações dos adversários possam ser analisadas antes das suas próprias ações serem enviadas. É obtido por meio de *software* especial que modifica os pacotes de dados enviados anexando a data e hora anterior à atual, o que leva o servidor a acreditar que o pacote foi enviado no horário certo, mas com atraso durante o transporte.

O jogo antidesportivo inclui abordagens que não dependem necessariamente de *software* ou *hardware* específico, mas sim de comportamentos antiéticos do jogador.

Boosting, ou *stat padding* é um método pelo qual um jogador usa várias contas e joga contra essas contas, com o objetivo de aumentar (ou seja, melhorar significativamente) a conta principal. Isto também pode ser realizado por vários jogadores, nos quais um ou vários jogadores concordam em permitir que um jogador ganhe, permitindo assim aumentar o número de pontos.

O *script* é uma técnica que permite ao *cheater* automatizar as tarefas ou executá-las com uma velocidade que não é normal, aumentando o desempenho do jogador. Estas técnicas são geralmente implementadas usando *software* específico ou explorando os recursos do jogo. Um exemplo é o recarregamento automático de uma arma quando certas condições são atendidas. Uma aplicação usada é *Rapid fire*, descrito abaixo.

As modificações *Rapid fire*, são métodos que dependem de modificações dos periféricos (exemplo: comandos, ratos, teclados) que o *cheater* usa para jogar. Estas modificações podem ser realizadas através de *hardware* ou *software* e permitem ao jogador disparar ou executar outras ações no jogo a um ritmo muito mais elevado, ficando assim com uma vantagem

para os seus adversários.

Character sharing é um método pelo qual várias pessoas jogam com a mesma conta. Isto dá uma maior vantagem pois a conta ficará online mais tempo, permitindo assim uma maior recolha de recursos, a obtenção de mais pontos e/ou a evolução mais rápida das suas habilidades e equipamentos pertencentes ao jogo.

Twinking é a ação de um jogador transferir equipamentos/recursos (do jogo) de uma conta mais evoluída para uma conta de um nível inferior, ganhando assim uma vantagem sobre os jogadores do mesmo nível. Esta transferência pode ocorrer entre contas do mesmo jogador ou de jogadores diferentes e muitas vezes envolve dinheiro.

Ghosting é um método pelo qual um jogador observa a jogabilidade de outro jogador usando um dispositivo (computador, telemóvel, consola) e uma conta. O *cheater* explora um mecanismo existente na maioria dos jogos online, que permite que as pessoas assistam aos jogos. Quando um jogador tem mais que um dispositivo e conta, ele pode observar o adversário, obtendo assim informações valiosas de maneira injusta, como a localização do adversário.

Os métodos de *cheating* descritos acima podem ser implementados de três maneiras diferentes:

- Modificar o código do jogo;
- Modificar o *software*;
- Alterar os pacotes.

Os métodos de *cheating* podem ser implementados com modificações no código do jogo. Embora os fornecedores de jogos distribuam apenas as versões binárias (ou seja, versões executáveis nas quais o código fonte não está legível), existem técnicas que revelam partes do código ou em alguns casos o código completo permitindo assim a sua alteração. Além de constituir um comportamento antidesportivo este método também é estritamente proibido pelo contrato da licença de jogos.

Os métodos de *cheating* também podem ser implementados através de modificações no *software* do dispositivo. Nestes métodos, não é o jogo que é alterado, mas sim os componentes do sistema onde o jogo é executado (por exemplo: modificação de drivers). Estes métodos podem ser usados, por exemplo, para desenhar todo o ecossistema do jogo sem os obstáculos,

permitindo assim que o *cheater* veja um adversário que não deve ser visível nas condições atuais.

Finalmente, alguns métodos de *cheating* podem ser implementados por meio de interceptação de pacotes, alteração e/ou manipulação. Este é um tipo de ataque *man-in-the-middle* e, permite que o *cheater* intercepte e, eventualmente, mude os dados dos pacotes recebidos/enviados, a fim de obter uma vantagem sobre os adversários (por exemplo, mudança do local ou ações do jogador).

3.2 Trabalho Relacionado

A partir da análise realizada, pode-se concluir que os métodos de *cheating* nos *e-Sports* podem ser muito diversos tanto na implementação como nos objetivos. Além disso, também está claro que os métodos de *cheating* nos *e-Sports* são muito diferentes dos praticados nos desportos tradicionais. Enquanto nos desportos tradicionais são utilizados substâncias para melhorar o desempenho do jogador, nos *e-Sports* alterações de *software/hardware* são implementadas para melhorar o desempenho do jogador. A partir disto, os métodos tradicionais ao combate de fraude não são adequados para este novo tipo de desporto.

A análise realizada mostra ainda que os métodos de *cheating* podem ser muito diversos, tanto no método de implementação como nos objetivos. Alguns métodos e equipamentos para prevenir ou detetar fraude nos *e-Sports* já foram apresentados, embora num número limitado ou com aplicações muito limitadas. Isto pode explicar a falta de uso de tais métodos em competições. Nesta secção, é realizada uma análise crítica a estes métodos.

Na secção anterior, foi detalhado que os métodos de *cheating* nos *e-Sports* podem ser organizados em três categorias principais:

- Modificar o código do jogo;
- Modificar o *software*;
- Alterar os pacotes.

Kenneth D, em [64] descreve um método para impedir fraudes em *e-Sports*. O método proposto depende de uma autoridade que certifica o *hardware* e *software* presentes numa de-

terminada plataforma de jogo, num determinado momento, incluindo os *drivers* da plataforma. Esta mesma autoridade também deve certificar todas as alterações executadas na plataforma (como alterações de *drivers* gráficos ou alterações de *hardware*). Por exemplo, os *drivers* em uso não foram alterados ou o código de jogo usado pelo jogador é o original. O principal objetivo deste método é evitar *cheating* através de modificações no sistema.

Uma abordagem semelhante é proposta por Pearson et al em [60], embora apenas ao nível de *software*. Esta abordagem começa por verificar a autenticidade do *software* relevante, como o *BIOS* e o sistema operacional, para garantir que o jogador está a jogar numa ”divisão”confiável e que nenhum componente de *software* esteja presente na mesma “divisão” que possa alterar a interação com o jogo. Se o sistema passar nesta primeira “auditoria”, o método verificará a identidade do jogo, para garantir que o código original não foi alterado, nomeadamente a aplicação de um *patch* ou alguma modificação não autorizada ao jogo. Essas verificações podem incluir verificações ao nível de assinaturas de *patches* conhecidos, verificações feitas nos nomes das rotinas usadas no jogo e verificações de alguns componentes do jogo. Dependendo da implementação do jogo, os componentes podem incluir arquivos executáveis, arquivos binários, arquivos de biblioteca ou *applets*. Também é proposto o uso de um agente de monitorização com foco no desempenho do jogador a ser executado na plataforma do jogador. No entanto, não esclarecem como o agente tem acesso às informações do jogo.

Eisen [24], propõe um método para lidar com *cheating* de adulteração de pacotes e, mais especificamente, ataques *man-in-the-middle*. A abordagem recolhe um conjunto de impressões digitais de dispositivos associadas ao identificador de sessão do utilizador. Ao recolher as impressões digitais do dispositivo e informações da sessão em vários locais, entre o conteúdo entregue pelo servidor durante uma sessão, o método proposto pode identificar a violação da sessão, ou seja, quando o conteúdo é o acedido a partir de diferentes endereços IP na mesma sessão. O método proposto é genérico e pode ser aplicado em qualquer domínio em que exista conteúdo de sessão, o que inclui os *e-Sports*.

Overton, em [56] propõe uma abordagem para a deteção de *cheating* que usa duas instâncias do jogo em execução em duas máquinas separadas, nas quais uma monitoriza a outra. Especificamente, as instâncias do jogo efetuam uma monitorização às ações dos outros jogadores, apontando ações que estão fora dos recursos do jogo (por exemplo, realizar um salto com uma distância maior que a máxima permitida pelo jogo). O método proposto indica quais os *cheaters* ao sistema do jogo para que estes possam assim ser removidos do jogo. O documento

também não esclarece como o método teria acesso às regras do jogo ou como poderia ser adaptado para outros jogos, uma vez que cada jogo tem as suas próprias regras.

Bijian Guo [30], propõe um método contra *cheating* para jogos online, especialmente desenvolvido para jogos de guerra, baseados em lançamento de projéteis. A ideia principal do método é enviar as informações de cada lançamento do projétil para o servidor, para o mesmo calcular a trajectória. O servidor retorna a trajectória (para o cliente), que apenas precisa de ser desenhada. O objetivo é, mover toda a lógica do jogo de um cliente não confiável para um servidor certificado. Este método, no entanto, só se aplica a este tipo específico de jogo. Além disso, o autor não esclarece como se protege de ataques *man-in-the-middle*, que podem alterar os pacotes recebidos/enviados para o servidor, obtendo assim vantagem sobre o adversário.

3.3 Análise Crítica

As abordagens apresentadas anteriormente de prevenção/deteção de fraude na sua maioria focam apenas um dos três tipos de *cheating*.

Podem ser encontrados outros métodos ou sistemas semelhantes/relacionados com os descritos anteriormente [80, 81, 53].

No entanto, a partir da análise dos métodos apresentados anteriormente, é possível retirar algumas breves conclusões. Cada método tem como foco apenas um tipo de *cheating*. Existem métodos que dependem de informações específicas do jogo, mas não é descrito como essas informações seriam obtidas ou mesmo como o método pode ser adaptado a diferentes jogos.

A abordagem apresentada tem mais vantagens e resolve as falhas referidas para as abordagens apresentadas anteriormente. A forma como foi desenvolvida permite uma rápida e fácil utilização não sendo necessário qualquer alteração a nível de *hardware*, basta simplesmente instalar e os dados da interação do jogador com os periféricos começam a ser recolhidos e o seu comportamento a ser analisado.

Outra vantagem é o facto de não se focar em nenhum jogo específico, ou seja, não depende de informação específicas do jogo para analisar o comportamento do jogador e assim fica disponível para qualquer tipo de jogo, não interferindo com o fluxo normal do jogo.

Por último, e mais importante a abordagem não se foca num tipo específico de *cheating*, mas em todos de uma forma geral, procurando encontrar desvios que diferem do comportamento do jogador.

Capítulo 4

Arquitetura

Como já foi referido anteriormente, a solução proposta consiste em desenvolver um método completamente independente do jogo e do tipo de fraude, para identificação contínua dos jogadores de *e-Sports* e assim detetar e/ou prevenir fraude.

Foram também já abordados os diferentes sistemas de HCI existentes, nomeadamente sistemas unimodais que são baseados em apenas uma modalidade e sistemas multimodais que agrupam várias modalidades. Dentro dos sistemas unimodais ainda podemos fazer um fazer uma divisão em três categorias: *visual-based*, *audio-based* e *sensor-based*.

É neste último, *sensor-based*, que se foca a arquitetura desenvolvida para a solução proposta. Os periféricos utilizados, neste caso teclado e rato são as únicas fontes de dados utilizadas e nesse sentido funcionam como sensores virtuais ou *soft sensors* [44].

A arquitetura desenvolvida encontra-se representada na figura 5 e foca-se em 3 componentes:

- Utilizador;
- Servidor;
- API.

A componente denominada por Utilizador é essencial na arquitetura, pois é o ponto de entradas das características biométricas (comportamentais) do utilizador e posteriormente armazenados numa base de dados NoSQL, MongoDB, orientada a documentos. Esta componente

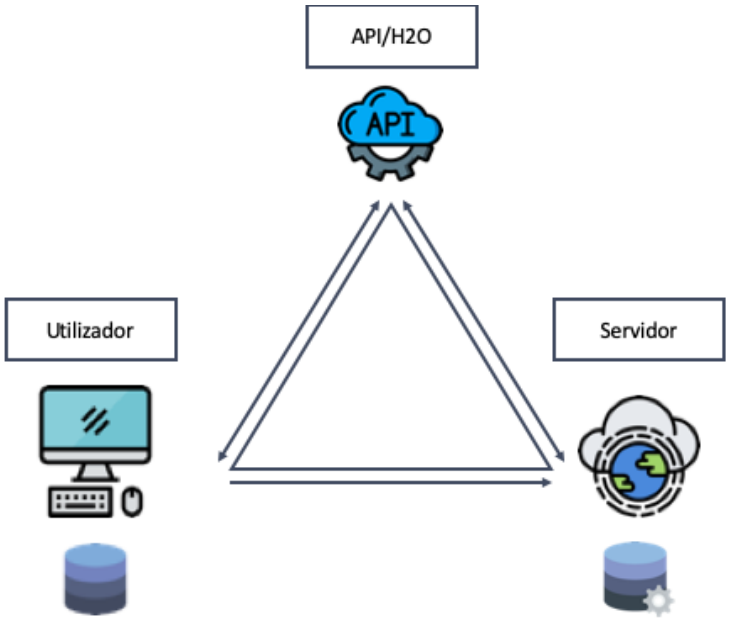


Figura 5: Arquitetura da solução proposta

comunica com as outras duas componentes (Servidor e API).

Para o Servidor são enviados os dados dos utilizadores que serão tratados de forma a criar as características do utilizador através dos dados recolhidos da interação com o rato e teclado. Após este tratamento os dados serão persistidos numa base de dados central, MongoDB, sendo esta responsável pelo armazenamento dos dados de todos os utilizadores. A base de dados é ainda utilizada para o armazenamento de meta-dados, nomeadamente os dados de treino dos modelos de *machine learning* (e.g. data de treino, métricas de *performance*). Por último, no servidor são ainda persistidos, no sistema de ficheiros, os modelos treinados para cada utilizador, que são depois colocados em memória pela API e utilizados para fazer previsões.

A API tem como única fonte de dados a base de dados central, sendo estes dados utilizados para criar e treinar modelos para cada utilizador tendo em conta as suas características individuais. Os modelos são treinados com o algoritmo DRF (*Distributed Random Forest*). O algoritmo gera um grupo de árvores de classificação, denominadas de *forest*, em vez de uma única árvore. De uma forma separada cada uma destas árvores tem uma aprendizagem fraca, mas quando combinadas permite melhores resultados do que o uso de uma única árvore. O resultado final da classificação é a previsão média de todas as árvores. Na categorização dos dados os modelos produzem um valor entre 0 e 1.

Por fim a aplicação do jogador comunica com a API para enviar dados de interação e obter previsões relativas à identidade do jogador. Dependendo do domínio de aplicação, a aplicação pode depois notificar o servidor do jogo em caso de inconformidade.

Em termos de armazenamento de dados no servidor, como foi referido anteriormente, foi utilizada a base de dados MongoDB. MongoDB é na verdade mais que base de dados, pois também fornece ferramentas nativas de processamento de dados, das quais se destacam *MapReduce* e *pipelene* de agregação. Estas podem operar numa coleção de dados fragmentada (particionada em várias máquinas, ou seja, dimensionamento horizontal). Estas ferramentas são poderosas para executar análises estatísticas em tempo real, que é um dos objetivos para solução.

Uma vez que os dados recolhidos e armazenados crescem linearmente com o número de utilizadores e o tempo de utilização, e que isto pode representar um *bottleneck* em termos de escala, foram analisados os requisitos computacionais associados a esta abordagem.

A tabela 4.1 apresenta uma projeção do crescimento dos dados consoante o número de utilizadores/jogadores num intervalo de tempo. A projeção teve em conta os atributos que cada documento tem, bem como o tamanho que estes ocupam na base de dados escolhida e a recorrência dos dados enviados para o servidor.

	1 utilizador	**100 utilizadores**	**10000 utilizadores**	**1000000 utilizadores**
5 minutos	136 bytes	13.28 Kbs	1.297 Mbs	129.7 Mbs
1 dia	12.75 Kbs	1.245 Mbs	124.5 Mbs	12.159 Gbs
1 semana	89.25 Kbs	8.716 Mbs	871.6 Mbs	85.115
1 mês	382.5 Kbs	37.354 Mbs	3.648 Gbs	364.8 Gbs
1 ano	4.545 Mbs	454.5 Mbs	44.382 Gbs	4.438 Tbs

Tabela 4.1: Projeção do crescimento dos dados

Capítulo 5

Identificação de Jogadores em *e-Sports*

5.1 Dados

O *dataset* utilizado para a realização deste projeto foi cedido pela empresa Performetric, na qual este trabalho se enquadra. Este *dataset* contém cerca de tf mil instâncias que representam os dados de interação de 308 jogadores diferentes, em 7 jogos diferentes.

Cada instância contém os dados agregados de 5 minutos de um jogador/jogo específico. A agregação dos dados com este intervalo é feita para minimizar a variância natural que existe neste tipo de dados. Isto é, os padrões que procuramos neste tipo de abordagem são mais difíceis de encontrar se analisarmos os eventos de interação um a um. Assim, esta agregação permite diluir a sua variação natural, e a emergência de comportamentos bem definidos.

Cada instância tem 41 variáveis ou colunas, que representam o jogador, o jogo, e os respectivos dados de interação. Este *dataset* tem a estrutura presente na Figura 6. As variáveis encontram-se detalhadas na Secção 5.1.2.

```
{
    "_id" : ObjectId("5ce9768901d89b7b38935fcc"),
    "User" : "player1",
    "Task" : "game",
    "KDTVar" : 187722.228,
    "KDTMean" : 427.868,
    "MAMean" : 0.883,
    "MAVar" : 0.621,
    "MVMean" : 1.55,
    "MVVar" : 1.184,
    "TBCMean" : 3708.0,
    "TBCVar" : 30471438.848,
    "DDCMean" : 1874.955,
    "DDCVar" : 3447825.744,
    "DMSLean" : 578025893.531,
    "DMSLVar" : 70797260686.498,
    "AEDMean" : 4.819,
    "AEDVar" : 36.536,
    "ADMSLMean" : 485002.5,
    "ADMSLVar" : 439994921034.379,
    "TBKVar" : 630938.031,
    "TBKMean" : 809.575,
    "LeftClicks" : 37,
    "RightClicks" : 8,
    "KeysPressed" : 1569,
    "WVMean" : 312.8,
    "MouseDistance" : 64.028,
    "MouseExcessDistance" : 6.343,
    "MousePrecision" : 94.237,
    "ErrorPerKey" : 0,
    "CDMean" : 1540.444,
    "CDVar" : 2537755.683,
    "DBCMean" : 1252.472,
    "DBCVar" : 2635021.883,
    "EDBCMean" : 3687.115,
    "EDBCVar" : 27820370.264,
    "SSDBCMean" : 19.187,
    "SSDBCVar" : 17070.231,
    "ASDBCMean" : 104737.051,
    "ASDBCVar" : 17376004330.752,
    "TDCMean" : 154.5,
    "TDCVar" : 12.5,
    "Game" : fortnite
}
```


Figura 6: Exemplo *raw data* exportado da base de dados MongoDB

5.1.1 Eventos Base

O processo de aquisição e extração de dados começa com a aquisição dos eventos de interação relevantes que o sistema operativo disponibiliza. Isto é feito atualmente por uma aplicação desenvolvida especificamente para este efeito e que é instalada em cada um dos computadores dos jogadores.

Esta aplicação é executada em segundo plano e não requer qualquer interação do jogador. No limite, o jogador nem precisa de saber que a aplicação está a corre. Os seguintes eventos são adquiridos pela aplicação e armazenados temporariamente no computador do jogador, sendo enviados posteriormente para o servidor:

MOUSE_DOWN (timestamp, [left — right], posX, posY): descreve "a primeira metade do clique (quando pressionamos o rato)", num determinado momento. Também indica qual o botão que foi pressionado (esquerdo ou direito).

MOUSE_UP (timestamp, [left — right], posX, posY): é similar ao anterior, mas descreve "a segunda parte do clique (quando soltamos o rato)".

MOVE (timestamp, posX, posY): descreve o movimento do rato, num determinado momento, para as coordenadas (posX, posY) no ecrã.

KEY_DOWN (timestamp, key): identifica uma determinada tecla (do teclado) que está a ser pressionada num determinado momento.

KEY_UP (timestamp, key): é similar ao anterior, mas para o movimento contrário, ou seja, quando a tecla deixa de ser pressionada.

A figura 7 detalha um pequeno excerto de um log da interação de um jogador com o rato. Nas duas primeiras linhas é possível ver que existiu um pequeno movimento do rato, seguindo-se de um clique com um pequeno arrasto nas linhas 3 a 5, terminando com outro movimento do rato.

A este tipo de registos são posteriormente acrescentados os dados relativos ao jogador e ao jogo, aquando do seu envio para o servidor.

```
MOV,635296941683402953,451,195
MOV,635296941684123025,451,197
MOUSEDOWN,635296941684443057, Left,451,199
MOV,635296941685273140 ,452 ,200
MOUSE UP,635296941685283141, Left,452,200
MOV,635296941685723185,452,203
MOV,635296941685803193,454,205
```

Figura 7: Excerto de um registo de interação de um jogador com o rato.

5.1.2 *Feature Extraction*

Os registos de interação criados pela aplicação mencionada anteriormente são processados para retirar informações que podem caracterizar o comportamento do jogador enquanto interage com o computador. As variáveis extraídas dos logs de interação encontram-se descritas abaixo.

A relação entre cada uma das variáveis e a *perfomance* do jogador, é na maior parte dos casos inversa. Um valor crescente reflete um desempenho decrescente. Por exemplo, maior distância percorrida entre cliques ou cliques mais longos mostram uma menor *performance*. As únicas duas exceções são a *mouse acceleration* e *mouse velocity*.

Estas duas variáveis não podem ser analisadas de forma tão linear uma vez que, tomando como exemplo a velocidade, o seu aumento só é sinal de maior eficiência até um certo ponto, a partir daqui contribui negativamente para a eficiência uma vez que torna os movimentos menos preciso. Estas relações entre as variáveis e a *performance* foram estabelecidas em estudos anteriores [13, 62].

De seguida descrevem-se todas as variáveis que são extraídas dos registos de interação que contêm os eventos detalhados na secção anterior. Como já mencionado, cada instância contém os dados agregados a cada 5 minutos, sendo em alguns casos calculada a média, e em outros uma contagem.

KDT (*Key Down Time*/Tempo de pressão de tecla): tempo decorrido entre dois eventos consecutivos KEY_DOWN e KEY_UP. Ou seja, por quanto tempo o jogador pressionou uma tecla específica.

MV (*Mouse Velocity*/Velocidade do rato): a distância percorrida pelo rato (em pixéis) sobre

o tempo (em milissegundos). A velocidade é calculada para cada intervalo definido por dois eventos MOUSE_UP e MOUSE_DOWN consecutivos. A velocidade entre os dois cliques é dada pela fórmula $\frac{r_{dist}}{(time_2 - time_1)}$, em que r_{dist} é dada pela equação 1.

MA (*Mouse Acceleration*/Aceleração do rato): a variação de velocidade do rato sobre o tempo (em milissegundos). A velocidade é calculada para cada intervalo definido por dois eventos MOUSE_UP e MOUSE_DOWN consecutivos e utilizando os valores da velocidade calculados através da variável anterior (*Mouse Velocity*).

TBC (*Time Between Clicks*/Tempo entre cliques): o tempo decorrido entre dois eventos consecutivos dos tipos MOUSE_UP e MOUSE_DOWN. Ou seja, quanto tempo tardou a efectuar um clique depois de ter efectuado o anterior.

DDC (*Distance During Clicks*/Distância durante cliques): distância percorrida pelo rato durante cliques (movimento de arrasto do ponteiro). Geralmente, mesmo em tarefas em que não há necessidade deste movimento, existe sempre algum movimento enquanto o utilizador clica. Isto tende a aumentar com a fadiga. Por outro lado, existem também tarefas/jogos em que este tipo de movimento é necessário e que pode revelar informação relevante sobre o utilizador.

DBC (*Distance Between Click*/Distância entre cliques): define a distância percorrida pelo rato (em *pixéis*) entre cada dois cliques consecutivos. Sejam dois eventos consecutivos do tipo MOUSE_UP e MOUSE_DOWN, denominados mup e mdo, respectivamente nas coordenadas $x1, y1$ e $x2, y2$ e nos instantes $time_1$ e $time_2$. Vamos ainda assumir dois vectores pos_x e pos_y, de tamanho n, que contêm as coordenadas dos eventos sucessivos do tipo MOVE entre mup e mdo. A distância real percorrida pelo ponteiro do rato r_dist é dada pela equação 1.

$$r_{dist} = \sum_{i=0}^{n-1} \sqrt{(posx_{i+1} - posx_i)^2 + (posy_{i+1} - posy_i)^2} \quad (1)$$

MouseDistance(Distância percorrida pelo rato): distância total percorrida pelo rato entre cada dois eventos consecutivos do tipo MOUSE_UP e MOUSE_DOWN, denominados mup e mdo, respectivamente. Calculada pelo somatório das distâncias entre cada dois eventos MOVE consecutivos, entre mup e mdo (Figura 8).

AED (*Average Excess of Distance*/Excesso médio de distância): excesso de distância, em média, que o apontador do rato viaja entre cada dois eventos MOUSE_UP e MOUSE_DOWN consecutivos. Sejam dois eventos consecutivos do tipo MOUSE_UP e MOUSE_DOWN, denominados mup e mdo, respectivamente nas coordenadas (x_1, y_1) e (x_2, y_2). Vamos ainda

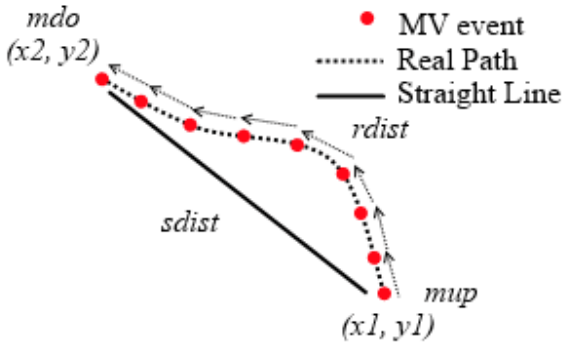


Figura 8: Distância em linha recta entre dois pontos no ecrã (s_{dist}) versus distância realmente percorrida pelo ponteiro (r_{dist}).

assumir dois vectores $posx$ e $posy$, se tamanho n, que contêm as coordenadas dos eventos sucessivos do tipo MOVE entre mup e mdo. Para calcular esta variável, é primeiro calculada a distância em linha recta entre (x_1, y_1) e (x_2, y_2) através da fórmula $s_{dist} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$. Depois é calculada a distância real percorrida pelo ponteiro do rato, através do somatório das distâncias entre cada dois eventos MOVE consecutivos, cujas coordenadas são dadas pelos vectores pos_x e pos_y O excesso médio de distância entre dois cliques consecutivos é então dado por $\frac{r_{dist}}{s_{dist}}$.

ADMSL (*Average Distance of the Mouse to the Straight Line*/Distância média do rato à linha recta): mede a distância média do ponteiro do rato à linha recta definida por cada dois cliques consecutivos. Sejam dois eventos consecutivos do tipo MOUSE_UP e MOUSE_DOWN, denominados mup e mdo, respectivamente nas coordenadas (x_1, y_1) e (x_2, y_2). Vamos ainda assumir dois vectores pos_x e pos_y, de tamanho n, que contêm as coordenadas dos eventos sucessivos do tipo MOVE entre mup e mdo. A soma das distâncias entre cada posição do rato e a linha recta definida pelos pontos (x_1, y_1) e (x_2, y_2) pela equação 2, em que o $ptLineDist$ devolve a distância mais curta entre o ponto e a linha recta estendida infinitamente, definida por (x_1, y_1) e (x_2, y_2). A distância média do rato à linha recta definida por dois cliques consecutivos é então dada por $\frac{s_{dists}}{n}$.

$$s_{dist} = \sum_{i=0}^{n-1} ptLineDist(posx_i, posy_i) \quad (2)$$

DMSL (*Distance of the Mouse to the Straight Line*/Distância do rato à linha recta): esta variável é similar à anterior, com a diferença que devolve o somatório do excesso de distância do rato em vez da sua média.

TBK (*Time Between Keys*/Tempo entre teclas): o tempo decorrido até pressionar uma tecla

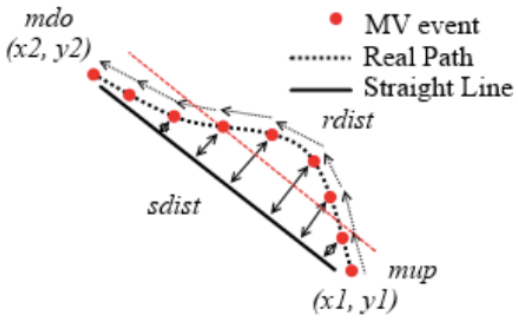


Figura 9: A distância média à linha recta é dada pela média do somatório das distâncias dos eventos MOVE à linha recta. A linha tracejada a vermelho representa a distância média a que o ponteiro viajou da linha recta.

depois de a anterior ter sido libertada. Exemplo: tempo entre dois eventos consecutivos dos tipos KEY_UP e KEY_DOWN.

LeftClicks(Cliques no botão esquerdo): contagem de cliques realizados com o botão esquerdo do rato, no intervalo temporal considerado.

RightClicks(Cliques no botão direito): contagem de cliques realizados com o botão direito do rato, no intervalo temporal considerado.

KeysPressed(Teclas pressionadas): contagem do número de teclas pressionadas no intervalo temporal considerado.

WV (*Writing Velocity*/Velocidade de escrita): velocidade de escrita, calculada pela divisão do número de teclas pressionadas num intervalo de tempo, pelo valor do intervalo.

ErrorPerKey(Erro por tecla): Contagem do número de vezes que a tecla *backspace* é pressionada, dividido pelo número total de teclas pressionadas no intervalo.

CD (*Click Duration*/Duração do clique): tempo decorrido entre dois eventos consecutivos MOUSE_DOWN e MOUSE_UP.

EDBC (*Excess of Distance Between Clicks*/Excesso da distância entre cliques): distância percorrida em excesso pelo rato entre dois cliques consecutivos, quando comparada com a linha reta entre as coordenadas desses dois cliques. Calculado através da diferença entre a variável MouseDistance e $sdist$ (Figura 10).

SSDBC (*Signed Sum of Distance Between Clicks*/Soma dos ângulos entre cliques): define

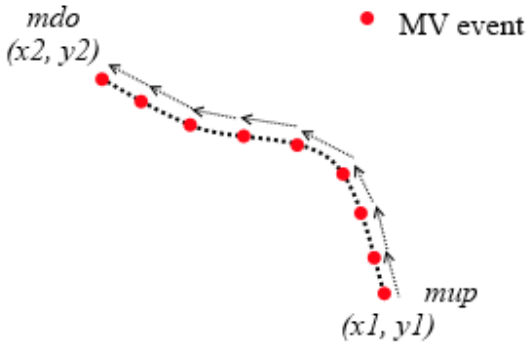


Figura 10: Processo de cálculo da distância real percorrida pelo rato, calculada através do somatório da distância entre cada dois eventos MOVE consecutivos, denotados na imagem pelos pontos vermelhos.

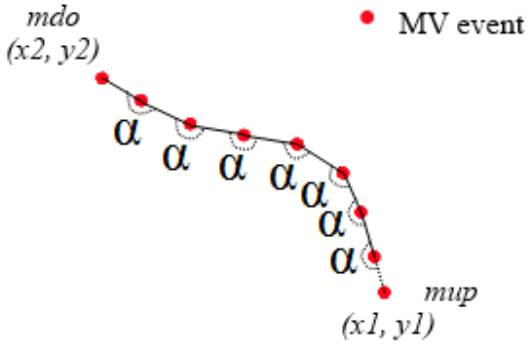


Figura 11: Processo de cálculo da curva da trajetória do rato, calculada através do somatório do ângulo entre cada dois eventos MOVE consecutivos, denotados na imagem pelos pontos vermelhos.

quanto o movimento do rato tendeu a curvar mais para a direita ou esquerda (Figura 11). Sejam dois eventos consecutivos do tipo MOVE, respectivamente nas coordenadas (x_1, y_1), (x_2, y_2) e (x_3, y_3). O ângulo α entre a primeira linha (definida por (x_1, y_1) e (x_2, y_2)) e a segunda linha (definida por (x_2, y_2) e (x_3, y_3)) é dado por $degree(x_1, y_1, x_2, y_2, x_3, y_3) = tan(y_3 - y_2, x_3 - x_2) - tan(y_2 - y_1, x_2 - x_1)$. Sejam ainda dois eventos consecutivos do tipo MOUSE_UP e MOUSE_DOWN, denominados mup e mdo respectivamente, e dois vectores pos_x e pos_y, de tamanho n, que contêm as coordenadas dos eventos sucessivos do tipo MOVE entre mup e mdo. A soma dos ângulos entre estes dois pontos é dada pela equação 3.

$$s_{angle} = \sum_{i=0}^{n-2} degree(posx_i, posy_i, posx_{i+1}, posy_{i+1}, posx_{i+2}, posy_{i+2}) \quad (3)$$

ASSDBC (*Absolute Signed Sum of Distance Between Clicks*/Soma absoluta dos ângulos entre cliques): variável calculada de forma semelhante à anterior, mas que não considera a direção da curva (i.e. esquerda ou direita), apenas a sua magnitude. É útil para distinguir os casos em que efetivamente existiu muita curva mas de magnitudes semelhantes entre a esquerda e a direita, e que por essa razão se anulam no cálculo da variável anterior.

TDC (*Time Double Click*/Tempo do duplo clique): o tempo decorrido entre dois cliques consecutivos (dois eventos do tipo MOUSE_UP), sempre que este tempo seja inferior a 200 ms (tempo típico de um duplo clique).

5.2 Metodologias

Nesta secção vão ser detalhados os dois métodos desenvolvidos para a deteção de fraude em *e-Sports*. Ambas as abordagens utilizam o mesmo *dataset*, assim como as variáveis que são utilizadas (descritas anteriormente). As duas abordagens vão focar-se em construir um modelo único do jogador com base nas sua interação durante o jogo com os periféricos (rato e teclado).

A primeira abordagem consiste em construir este modelo através de meios estatísticos, nomeadamente removendo os *outliers* através do cálculo do IQR (*Interquartile Range*) e calculando os limites mínimos e máximos para cada variável considerada.

A segunda abordagem usa *machine learning*, utilizando o algoritmo DRF (*Distributed Random Forest*) para o treino dos modelos.

5.2.1 IQR

A primeira abordagem consiste em determinar através de meios estatísticos qual era o comportamento considerado normal para o jogador. Esta abordagem foi implementada com o intuito de ser computacionalmente simples, evitando a complexidade do treino dos modelos de *machine learning*. De facto, e como se descreve de seguida, a abordagem assenta no cálculo de algumas medidas relativamente simples. Isto é particularmente importante num cenário em que os modelos têm que ser atualizados com frequência, à medida que novos dados vão chegando e os perfis dos utilizadores necessitam de ser atualizados.

Para além disso, esta abordagem tem ainda a vantagem de fornecer não apenas um resultado mas ainda uma explicação. Ou seja, é possível apresentar a um utilizador do sistema (não necessariamente o jogador) informação sobre o porquê de um dado resultado ter sido dado, em termos das variáveis de interação usadas (e.g. porque a velocidade do rato é muito acima da velocidade máxima normal para este jogador). Este tipo de explicações pode ainda

ser apresentado graficamente, como mostrado nesta secção, o que permite a análise fácil e intuitiva de várias variáveis e modelos simultaneamente.

A abordagem foi implementada da seguinte forma. Primeiramente foi efetuada uma normalização dos dados considerados acima para a construção do modelo do jogador. Posteriormente, foi então criado o modelo de interação de cada jogador. Este modelo foi criado através do cálculo do IQR (*Interquartile Range*) para cada uma das variáveis. O método mede a diferença entre os quartis **Q3** e **Q1**, sendo que um valor inferior a $Q1 - 1.5 * IQR$ ou superior a $Q3 + 1.5 * IQR$ é considerado anormal e não respeita o intervalo do que é considerado normal. Esta abordagem permite assim definir os intervalos do que é considerado o comportamento normal ou usual de um jogador.

Nesta abordagem o modelo é assim composto pelos limites máximo e mínimo de cada variável, como ilustrado na figura 12. Para fazer a classificação de uma determinada instância como fraude ou não, é calculado, para essa dada instância, o número de variáveis cujos valores se encontram entre os limites estabelecidos no modelo. A existência ou não de fraude é determinada pelo número de variáveis que se encontram dentro dos limites, e por um *threshold* mínimo estabelecido.

A Figura 13 representa, graficamente, o modelo de interação do Jogador 2, com os seus limites superior e inferior (a vermelho e verde, respetivamente). Ilustra-se ainda, em tracejado, uma instância de interação selecionada aleatoriamente, que neste caso se encontra dentro dos limites do modelo pelo que a interação deverá efetivamente pertencer ao Jogador 2. Por simplicidade, apenas algumas variáveis são incluídas nesta visualização.

Com vista a avaliar o desempenho desta abordagem na identificação dos jogadores, foi implementada uma metodologia que assenta nas seguintes métricas:

- *Accuracy*;
- *Precision*;
- *Recall*;
- *F1 Score*.

A metodologia foi aplicada para cada jogador individual, num total de 308, tal como descrito de seguida. Os dados de cada jogador foram divididos em duas metades. A primeira metade

```
{
    "username" : "player1",
    "ADMSL_inf" : 0.013806703,
    "ADMSL_sup" : 0.203437054,
    "AED_inf" : 0.0,
    "AED_sup" : 0.0,
    "KDT_inf" : 0.002092049,
    "KDT_sup" : 0.037962243,
    "CD_inf" : 0.0000006017113101,
    "CD_sup" : 0.00000106572314,
    "DBC_inf" : 0.00083241,
    "DBC_sup" : 0.003240363,
    "DMSL_inf" : 0.000013863,
    "DMSL_sup" : 0.000032122,
    "ErrorPerKey_inf" : 0.49754902,
    "ErrorPerKey_sup" : 0.504084967,
    "LeftClicks_inf" : 0.016248839,
    "LeftClicks_sup" : 0.128597957,
    "MA_inf" : 0.029280431,
    "MA_sup" : 0.227543125,
    "MV_inf" : 0.106438402,
    "MV_sup" : 0.461870881,
    "MouseDistance_inf" : 0.553229985,
    "MouseDistance_sup" : 0.771942135,
    "MouseExcessDistance_inf" : 0.034187271,
    "MouseExcessDistance_sup" : 0.096628433,
    "RightClicks_inf" : 0.750789889,
```


Figura 12: Excerto do modelo do jogador *player1*

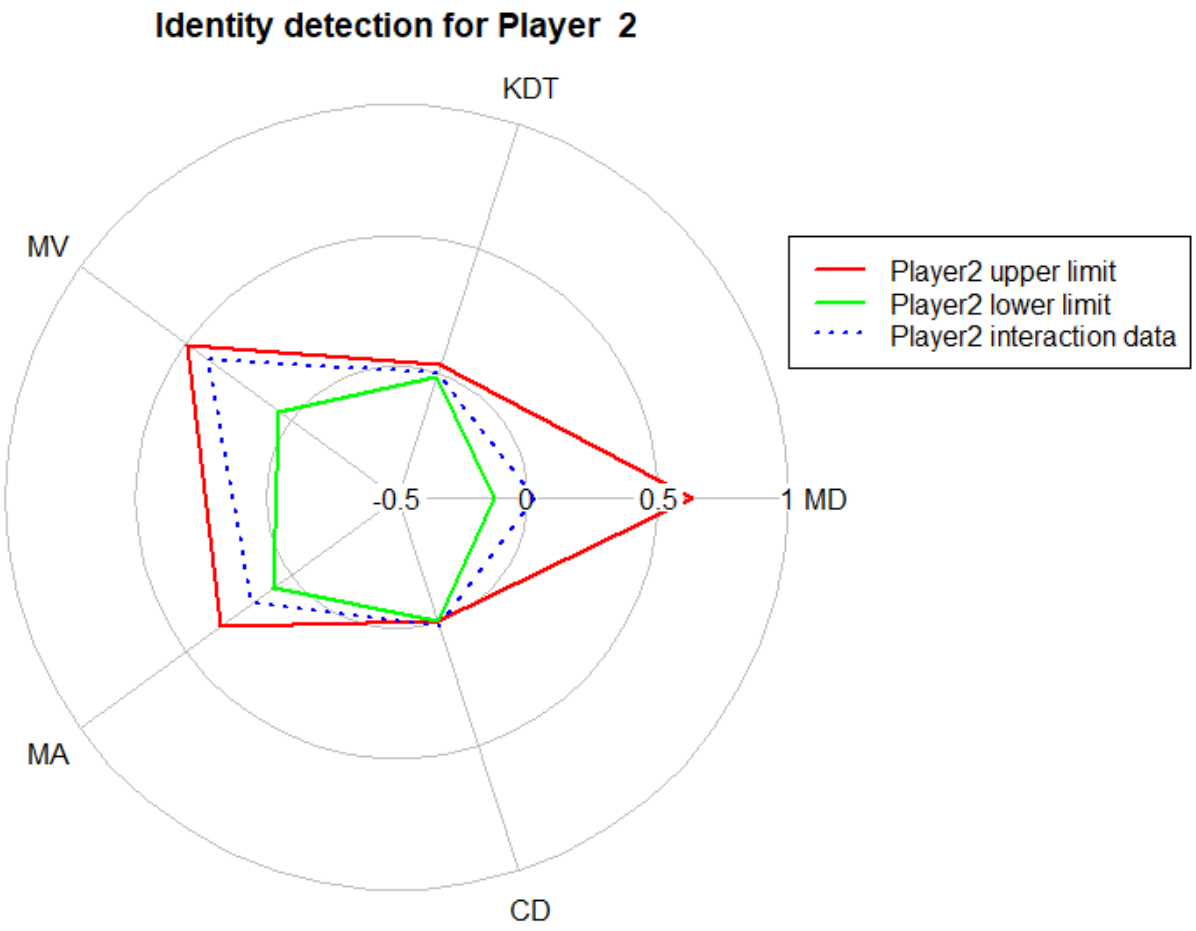


Figura 13: Interação do jogador 2. As linhas verde e vermelha representam os limites inferior e superior do seu modelo de interação, respetivamente, e a linha tracejada uma instância de interação aleatória.

Modelo	Accuracy	Precision	Recall	F1 Score
IQR	0,5538	0,5367	0,5993	0,5756

Tabela 5.1: Resultados para abordagem IQR

foi utilizada para calcular o modelo de interação de cada jogador, isto é, os limites superior e inferior de cada variável, tal como descrito anteriormente.

De seguida, e para cada modelo de interação, foram selecionados dados aleatoriamente da segunda metade dos *datasets* de todos os utilizadores. Uma vez que se sabe a proveniência de cada uma das instâncias destes *datasets*, é possível submetê-los para classificação ao modelo do jogador que se está a avaliar, e determinar a sua *performance*.

Essencialmente, este processo pretende simular a ocorrência de fraude de identidade, isto é, jogadores que se fazem passar por outros.

O resultado global desta validação encontra-se resumido na tabela 5.1. Dado o elevado número de modelos treinados, opta-se por colocar nesta tabela apenas o valor médio de cada métrica, para todos os modelos.

Os resultados observados estão aquém do esperado. Para isto poderão ter contribuído vários fatores, nomeadamente a agregação dos dados por jogador em vez de por jogador/jogo. No entanto, um dos objetivos era determinar se seria possível encontrar um perfil global de cada jogador, independente do jogo em causa. Contudo, veio a verificar-se que a variabilidade que existe na interação entre diferentes jogos torna esta abordagem pouco adequada. Isto é visível pela distribuição das variáveis que acaba por ser muito idêntica entre os jogadores, como ilustrado nas figuras 14, 15 e16.

Analisando as figuras 14, 15 e 16, notamos que na figura 14 a distribuição é muito idêntica para os 10 jogadores. Podemos olhar para casos particulares: a distribuição dos valores para jogador 1 é muito idêntica à do jogador 3, ou seja, é normal que a variável utilizada no caso do jogador 1 esteja entre os limites definidos no modelo do jogador 3. Num caso prático, e focando no jogador 1, se recebermos uma instância de dados do jogador 3 é muito provável que o método considere que a variável *Key Down Time* está dentro dos limites definidos para o modelo do jogador 1, pois a distribuição dos dados é muito idêntica.

Isto é um problema porque a interação dos jogadores na maioria das vezes está dentro dos limites de outros jogadores, daí os resultados não terem uma taxa de sucesso elevada.

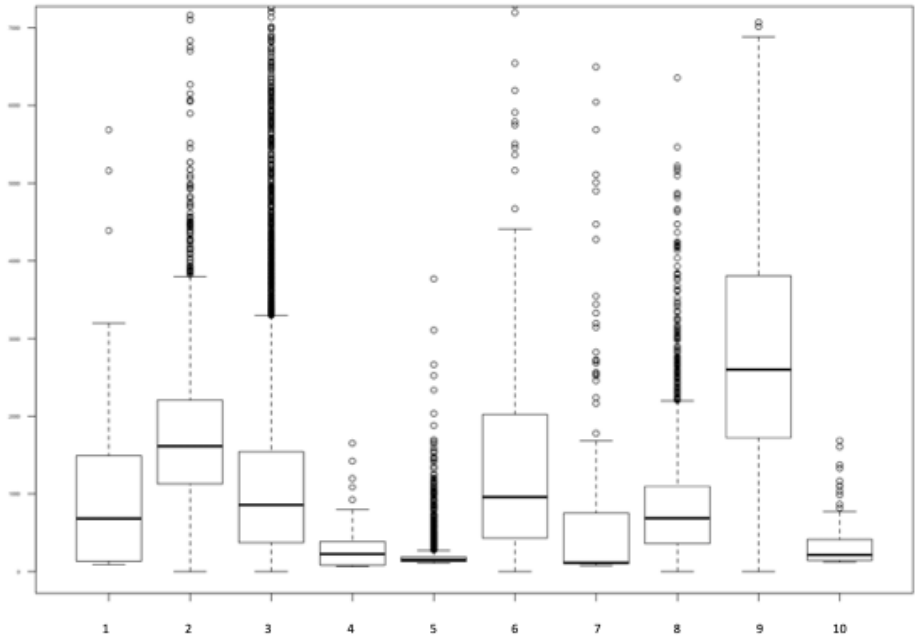


Figura 14: *Key Down Time* por jogador (apenas 10 jogadores)

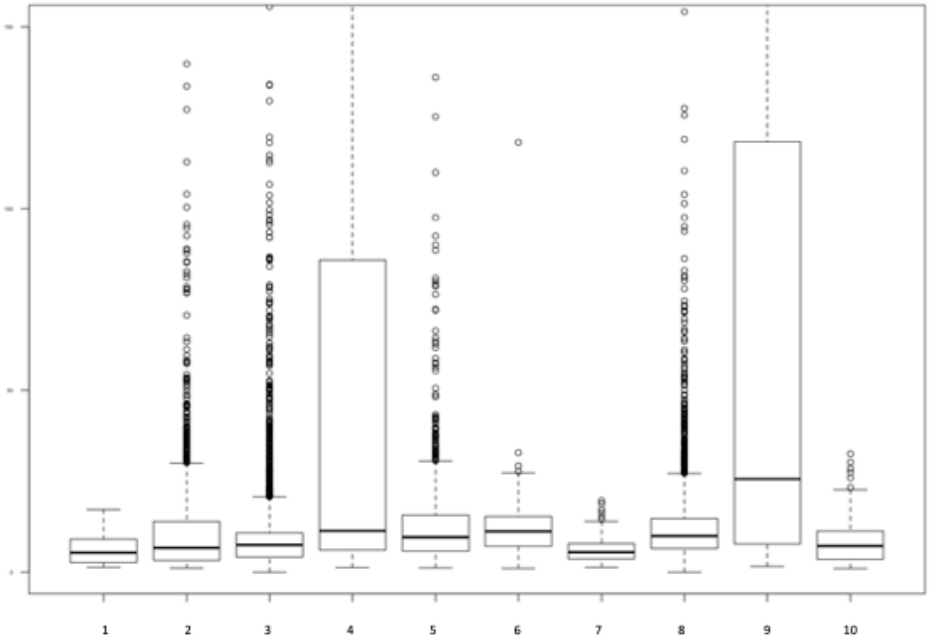


Figura 15: *Average Excess of Distance* por jogador (apenas 10 jogadores)

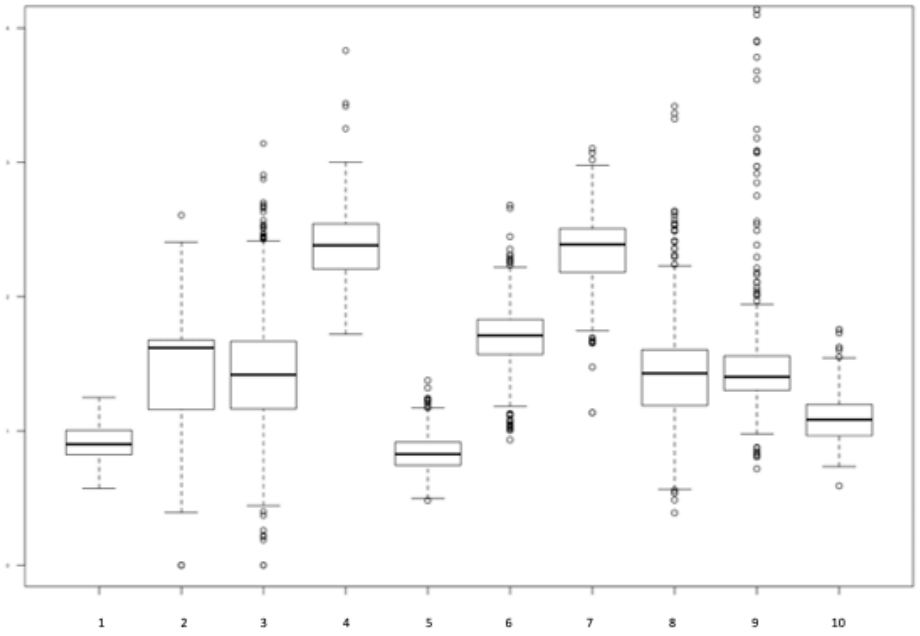


Figura 16: *Mouse velocity* por jogador (apenas 10 jogadores)

A figura 17 ilustra exatamente este problema. Os dados de interação do jogador que estamos a simular, que é o *Player* 2, são muito parecidos com os dados do jogador *Player* 1 e que 88% das suas características encaixam perfeitamente no modelo do outro jogador.

Em função dos resultados, e apesar das potenciais vantagens já enumeradas do método proposto, validou-se uma segunda abordagem baseada em *Machine Learning*, descrita na secção seguinte.

5.2.2 *Machine learning*

Em função dos problemas identificados na abordagem anterior, foi implementada uma segunda que consistiu em treinar um modelo de *machine learning* para cada jogador. O modelo usa as mesmas variáveis que as usadas na abordagem anterior e o *output* é também o mesmo: determinar se a instância de interação passada ao modelo terá ou não sido gerada pelo utilizador associado ao modelo.

De forma semelhante à abordagem anterior, o *output* contém a resposta à pergunta bem como o grau de certeza ou confiança na resposta, quantificado num valor entre 0 e 1. Para este desenvolvimento foi utilizada a *framework* H2O para o treino dos modelos bem como a previsão de resultados.

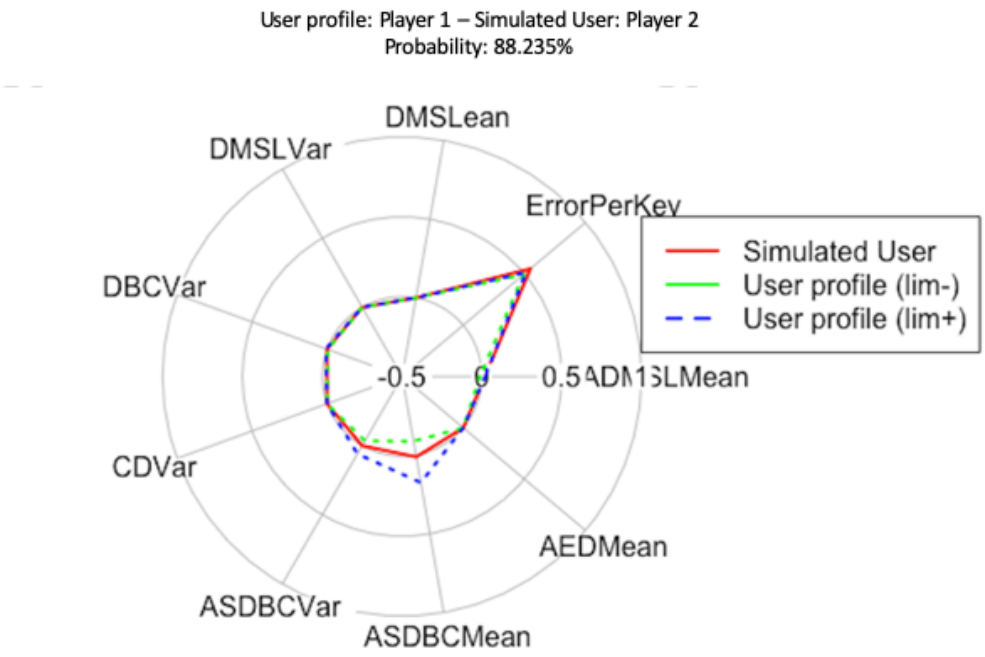


Figura 17: Interação do jogador *Player* 2 com o modelo do jogador *Player* 1

Os modelos, um por cada jogador, foram treinados com o algoritmo DRF (*Distributed Random Forest*). O algoritmo gera um grupo de árvores de decisão, denominado *forest* por ser um conjunto ou *ensemble* de árvores, em vez de uma única árvore (*tree*).

Cada árvore é treinada com uma parte das instâncias e uma parte das variáveis. De uma forma separada cada uma destas árvores é um modelo fraco no sentido em que, isoladamente, faz previsões geralmente fracas. Mas, quando combinadas entre si, permitem melhores resultados do que o uso de uma única árvore complexa.

Entre outros efeitos, isto permite evitar o *overfitting* e obter modelos mais generalizáveis. Ou seja, modelos capazes de ignorar dados que são outliers ou ruído e não são representativos da realidade modelada. Este modelo foi escolhido precisamente por este fator, à semelhança do propósito da abordagem anterior.

No caso concreto deste trabalho, cada modelo foi treinado com 50 árvores. A previsão do modelo é assim dada pela média de todas as árvores.

A metodologia seguida para o treino dos modelos foi a seguinte. Primeiramente, os dados de cada utilizador são divididos em duas partes iguais. É criado, para cada utilizador, um *dataset* constituído em partes iguais por: dados do próprio utilizador (constituídos pela primeira metade do seu *dataset*) e dados de outros utilizadores (retirados aleatoriamente das segundas

metades dos *datasets* dos restantes utilizadores). Uma nova coluna é acrescentada contendo a variável independente que indica, para cada instância, se esta pertence ou não ao utilizador ao qual o *dataset* está associado. Por último, cada *dataset* é aleatorizado. Obtém-se, por este processo, um *dataset* balanceado, apto para ser usado em tarefas de *machine learning*.

De seguida, cada um destes *datasets* é dividido em duas partes: a primeira contendo 60% dos dados, e que será usada para treino, e a segunda contendo os restantes 40%, e que será usada para validação. No total, os dados de treino contêm 127 mil instâncias e os dados de teste 84 mil.

De modo a determinar o desempenho dos modelos fo utilizado o método *K-Fold Cross Validation*, com o valor de K=5. O método *Cross Validation* utiliza precisamente dois *datasets*, sendo estes os dados de treino (*training*) e os dados de validação (*validation*). As métricas de precisão do modelo são calculadas através da comparação das previsões realizadas pelo modelo com os valores conhecidos (reais). O método está divido em 4 etapas:

1. Dividir o *dataset* para treino e para validação;
2. Treinar o modelo com o *dataset* de treino;
3. Validar o modelo com o *dataset* de validação;
4. Calcular as métricas de precisão através da comparação dos valores previstos com os conhecidos.

O método *K-Fold Cross Validation* é um tipo comum de *Cross Validation* usado para *machine learning*. Este método possui um único parâmetro denominado de K que representa em quantos grupos o *dataset* vai ser dividido. Tendo em conta que foi definido $K = 5$, então o *dataset* vai ser dividido em 5 grupos e o processo abaixo vai ser repetido 5 vezes, e em cada iteração terá um grupo de teste diferente, como detalhado na figura 18. Em cada iteração, são levadas a cabo as seguintes tarefas:

1. Selecionar o grupo de teste;
2. Selecionar o grupo de treino;
3. Treinar e avaliar o modelo com o grupo de teste;

Figura 18: Exemplo do método K-Fold Cross Validation, com K=5

4. Guardar a avaliação e descartar o modelo.

No final do processo descrito é treinado um único modelo com todos os dados, e são calculadas as métricas de precisão desse modelo com base na média das métricas de todos os 5 modelos criados. Ou seja, na prática são treinados 6 modelos, mas apenas o último é utilizado efetivamente para fazer previsões.

Para terminar é ainda calculada a importância relativa de cada variável para o modelo, através da influência que as mesmas tiveram durante o treino do modelo. A importância relativa de cada variável indica quanto a previsão depende de cada uma das variáveis, no sentido em que há variáveis que quando alteradas afetam pouco ou nada o valor da previsão, enquanto que outras o afetam de forma significativa. Ainda que de uma forma um pouco rudimentar, isto permite gerar alguma explicação sobre o modelo.

A importância das variáveis nos modelos permite determinar, por exemplo, e como hipotetizado, que cada jogador tem padrões de interação diferentes pois cada modelo tem importâncias relativas das variáveis diferentes.

As figuras 19, 20 e 21 representam isso mesmo para 3 jogadores diferentes: o treino de cada

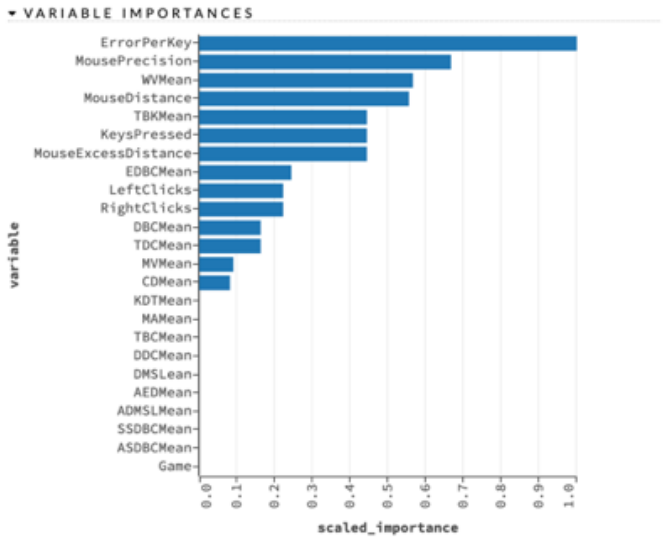


Figura 19: Importância das variáveis para o jogador 1

modelo originou importâncias relativas diferentes para cada variável, consoante o jogador. Após uma rápida análise das mesmas verificamos que a variável mais importante difere para os três jogadores, sendo *ErrorPerKey*, *CDMean*, *EDBCMean* as variáveis mais importantes para os modelos dos jogadores 1, 2 e 3 respectivamente.

Estas variáveis são as que têm uma maior grau de importância para determinar se dada uma instância de interação, ela pertence ou não ao utilizador associado ao modelo. Algumas das variáveis usadas no treino do modelo tornam-se insignificantes para o mesmo, pois não acrescentam qualquer mais valia. O conjunto destas variáveis tem como objetivo tornar o modelo único.

É possível verificar quais as variáveis mais importantes para cada modelo através da interface gráfica e podemos ver que as mesmas variam de modelo para modelo, o que permite saber que cada variável tem importâncias diferentes para cada modelo.

À semelhança do que foi feito para a primeira abordagem, foram também calculadas e analisadas as mesmas métricas de *performance* para os modelos. O resultado destes testes encontram-se na tabela 5.2 e, comparativamente aos da abordagem anterior, são muito mais satisfatórios. Dado o elevado número de modelos treinados, opta-se por colocar nesta tabela apenas o valor médio de cada métrica, para todos os modelos.

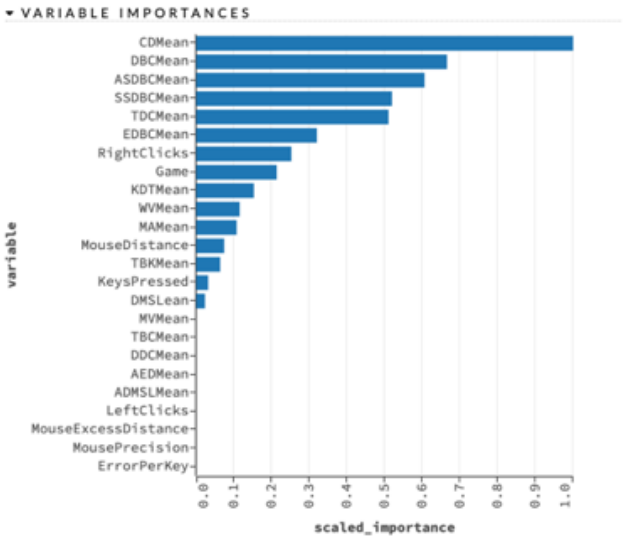


Figura 20: Importância das variáveis para o jogador 2

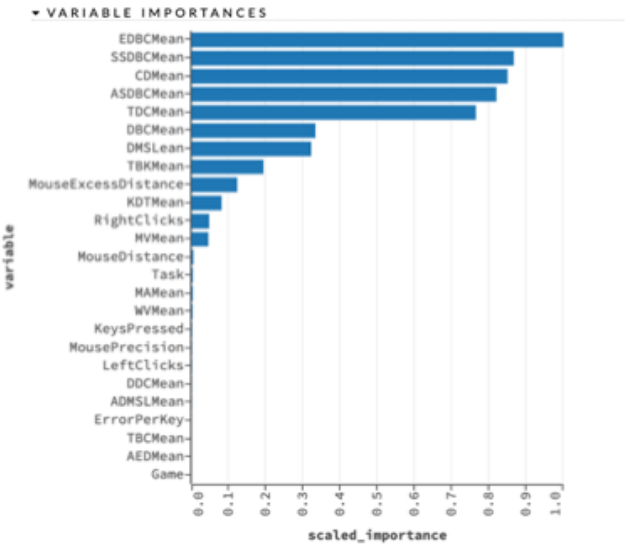


Figura 21: Importância das variáveis para o jogador 3

Modelo	Accuracy	Precision	Recall	F1 Score
Machine Learning	0,9491	0,9390	0,9378	0,9511

Tabela 5.2: Resultados para abordagem de *machine learning*

5.3 API

Como já referido na secção da arquitetura, existe uma API e a mesma foi desenvolvida em *Node.js*. A API só contempla a abordagem de *machine learning* e foi criada com o intuito de automatizar todo o processo para detetar e/ou prevenir fraude. A API recebe instâncias de dados, idênticos ao detalhado na figura 6 e verifica para cada instância se esses dados poderão ou não pertencer ao jogador associado, ou seja, que não está a usar nenhum tipo de *cheating* e o grau de confiança na previsão.

Para isso, foi desenvolvida uma aplicação no lado do servidor que implementa todo o *pipeline* de treino, gestão e manutenção dos modelos, da seguinte forma.

Sempre que existe, para um dado utilizador, determinado número novo de instâncias de dados, um novo modelo é treinado para esse utilizador automaticamente. Em alternativa, o treino do modelo também pode ser despoletado pela utilização do *endpoint* específico, descrito mais abaixo. Duas situações diferentes podem ocorrer:

1. Não existe um modelo prévio do utilizador - nesta situação, assumimos que todos os dados recebidos pertencem efetivamente ao utilizador e é treinado um modelo com os seus dados, tal como descrito anteriormente. Isto é uma limitação desta abordagem no sentido em que estes dados podem já ter sido gerados numa invasão de identidade, o que não sendo provável, é possível;
2. Existe um modelo prévio do utilizador - nesta situação, cada instância de dados que chegou ao servidor foi já classificada pelo modelo existente. Apenas os dados classificados como pertencendo efetivamente ao utilizador são utilizados no treino do novo modelo. Isto é feito para impedir que dados que não pertencem ao utilizador sejam considerados no treino do modelo.

Em qualquer dos casos descritos anteriormente, após o treino do modelo este é exportado e guardado numa pasta do sistema de ficheiros. É ainda guardada informação na base de dados acerca do modelo, incluindo as suas métricas, assim como a data de treino do mesmo. Isto permite determinar, num dado momento, a quantidade de dados novos e a consequente necessidade de treinar ou não um novo modelo.

Existe, assim, uma *pool* de modelos dos jogadores registados. Sempre que o servidor inicia e

```
"predict": [
    {
        "number": 1,
        "predict": "TRUE",
        "value": "0.7798076923005283"
    },
    {
        "number": 2,
        "predict": "FALSE",
        "value": "0.92"
    },
    {
        "number": 3,
        "predict": "FALSE",
        "value": "0.89"
    },
    {
        "number": 4,
        "predict": "FALSE",
        "value": "0.91"
    }
]
```


Figura 22: Resposta da API para o método *api/predict*

um jogador faz login, o seu modelo é obtido do disco e carregado para memória, ficando assim o servidor apto a fazer previsões. Sempre que um novo modelo é treinado, o modelo em disco e em memória (se existir) é substituído.

Quando um modelo está em memória a API pode então utilizá-lo para fazer a previsão da identidade do jogador. A figura 22 contém um exemplo da resposta da API para quatro instâncias de dados.

Todos os métodos desenvolvidos encontram-se descritos abaixo.

***Endpoint*:** api/userInfo
Método: POST
Descrição: Permite a inserção de uma ou mais instâncias de dados de um jogador na base de dados. Se já existir um modelo desse jogador, os dados são automaticamente classificados.
***Input*:** Instância de dados a ser adicionados. Exemplo figura 6 (excepto o campo **_id**)
***Output*:** Sucesso/Insucesso

***Endpoint*:** api/training/:user
Método: POST

Descrição: Inicia o treino de um modelo para um jogador.

***Input*:** Nome do modelo

***Output*:** Sucesso/Insucesso

***Endpoint*:** api/training/status/:user

Método: GET

Descrição: Permite obter informação de progresso sobre o estado do treino do modelo de um jogador.

***Input*:** Nome do modelo

***Output*:**

- *Started*;
- *In Progress*;
- *Done*;
- *Error*.

***Endpoint*:** api/predict

Método: POST

Descrição: Realiza uma previsão dado um conjunto de dados e o identificador do suposto jogador.

***Input*:**

- Nome do modelo;
- Instância de dados.

***Output*:** Sucesso/Insucesso

***Endpoint*:** api/classification

Método: GET

Descrição: Devolve todas as métricas de *performance* existentes para os modelos, permitindo perceber a qualidade dos modelos atualmente em memória

***Input*:** N/A

***Output*:** Todos os resultados existentes na base de dados

***Endpoint*:** api/classification/:user

Método: GET

Descrição: Devolve todas as métricas existentes para um modelo de um jogador específico

***Input*:** Nome do modelo

***Output*:** Todos os resultados existentes na base de dados para o modelo desejado

5.4 Análise Crítica

Depois de apresentadas, as duas abordagens têm resultados relativamente diferentes. A primeira abordagem, apesar de ter resultados aquém do esperado, permite de forma muito mais fácil explicar e perceber o porquê de um determinado resultado, pois o modelo não é modelado dentro de uma caixa negra: é explicável em função das próprias variáveis e de características tangíveis do domínio.

Contudo, verificaram-se com alguma frequência sobreposições entre os perfis dos jogadores, sobretudo com o aumento do número dos mesmos, que tornam a abordagem pouco adequada para os fins propostos.

Como referido anteriormente, este problema pode dever-se a uma má abordagem, pois ao tentar tornar o modelo genérico e independente do jogo, ou seja, o mesmo modelo para diferentes jogos acabamos por agregar todos os dados do jogador para todos os jogos, em vez de jogador/jogo. Isto levou a que a distribuição dos dados fosse muito idêntica na maior parte do modelos.

Acreditamos que ao separar os dados por jogador/jogo poderíamos ter de facto melhores resultados. Isto é, em parte, comprovado pelos resultados da segunda abordagem em alguns jogadores/modelos, a variável que identifica o jogo a ser jogado surge como uma variável importante para classificar a identidade do jogador. Isto significa que a interação dos jogadores muda de forma significativa entre jogos.

Em relação à segunda abordagem, os resultados são significativamente melhores que os da abordagem anterior. Isto significa que é possível desenvolver uma ferramenta baseada em *machine learning* para o fim da classificação da identidade dos jogadores. Contudo, esta ferramenta sofre da desvantagem associada aos modelos de caixa negra: não é possível ou é difícil dar uma justificação/explicação para o resultado de uma previsão, o que neste domínio

é relevante.

Em função destes resultados, a proposta é que ambas as abordagens sejam utilizadas em conjunto. A primeira pode ser utilizada não para fazer previsões mas como forma de gerar explicações, de apresentar de forma gráfica e intuitiva as diferenças entre perfis de jogadores e de perceber os perfis comportamentais e físicos de cada jogador. A segunda, pode efetivamente ser utilizada para fazer uma previsão da identidade de cada jogador.

Capítulo 6

Considerações Finais

6.1 Conclusão

Neste trabalho abordou-se o problema da deteção de fraude de identidade no domínio dos *e-Sports*. Este é um domínio muito específico com as suas particularidades. Nomeadamente, qualquer abordagem que seja desenvolvida não pode interferir com a *performance* quer do jogador quer do *hardware/software* em que o jogo decorre, nem do meio de comunicação existente entre o jogador e o servidor do jogo.

Nesse sentido, a principal contribuição deste trabalho é o desenvolvimento de uma abordagem baseada na interação do jogador com o computador. Esta abordagem é inovadora no sentido em que não existem, à data da escrita deste documento, abordagens ou sistemas similares.

De facto, os sistemas existentes baseiam-se em elementos como a observação de alterações no *hardware/software* utilizado, a monitorização de alterações nos pacotes de dados trocados entre o cliente e o servidor, ou ainda na utilização de elementos específicos do jogo (com a desvantagem da necessidade de integração da abordagem no próprio jogo, o que requer a participação das editoras).

Em contrapartida, a solução proposta apresenta as seguintes vantagens:

- Independência do jogo, não requerendo adaptações do mesmo e podendo ser utilizada em virtualmente qualquer jogo;

- Aplicação cliente muito pouco exigente em termos de recursos, não afetando a experiência de jogo;
- Possibilidade de formalizar um perfil de interação para cada jogador, o que abre a possibilidade de futuros desenvolvimentos tais como a análise da *performance* e o treino personalizado dos jogadores com vista à melhoria de determinados aspetos (e.g. velocidade, precisão).

O trabalho desenvolvido no decorrer desta dissertação serviu ainda para validar, numa primeira instância, uma abordagem que se encontra agora em fase de implementação na Performetric, empresa na qual o projeto foi desenvolvido.

Foi ainda submetido, através do Instituto Nacional da Propriedade Inteletual, um pedido provisório de patente para salvaguardar a propriedade intelectual do trabalho desenvolvido, com o título "Method and Apparatus for Fraud Prevention in E-Sports"(Pedido nº 20191000003627).

Por essa razão, não foi possível a escrita/publicação de qualquer artigo científico a detalhar este trabalho.

6.2 Trabalho futuro

Os próximos passos deste projeto passarão por testar num ambiente real, algo que a Performetric já se encontra a realizar e assim melhorar a implementação com base no *feedback* recebido e detetar possíveis pontos de melhoria e outras abordagens.

Passa ainda por retificar a primeira abordagem, ou seja, efectuar uma separação dos dados por jogador/jogo para assim validar se realmente o problema da primeira abordagem foi a agregação dos dados. Esta retificação vai levar a que passe a existir um modelo por jogador/jogo algo que até aqui ainda não tinha acontecido.

Por último e para aproveitar grande parte do que foi desenvolvido, desenvolver uma aplicação direccionada para os treinadores das equipas de *e-Sports*, onde através da análise das métricas dos jogadores e dos respectivos treinos avalia a *performance* e fornece *KPI's* que indicam se as mesmas estão a melhorar ou piorar. A partir daqui, o treinador pode estudar treinos específicos para melhorar determinada característica e perceber como os treinos estão a correr

e onde estão a falhar.

Bibliografia

[1] Emile Aarts and Boris De Ruyter. New research perspectives on ambient intelligence. *Journal of Ambient Intelligence and Smart Environments*, 1(1):5–14, 2009.

[2] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *International symposium on handheld and ubiquitous computing*, pages 304–307. Springer, 1999.

[3] Jake K Aggarwal and Quin Cai. Human motion analysis: A review. *Computer vision and image understanding*, 73(3):428–440, 1999.

[4] Unai Alegre, Juan Carlos Augusto, and Tony Clark. Engineering context-aware systems and applications: A survey. *Journal of Systems and Software*, 117:55–83, 2016.

[5] Arwa Alsultan and Kevin Warwick. Keystroke dynamics authentication: a survey of free-text methods. *International Journal of Computer Science Issues (IJCSI)*, 10(4):1, 2013.

[6] Mahnoush Babaeizadeh, Majid Bakhtiari, and Mohd Aizaini Maarof. *Keystroke dynamic authentication in mobile cloud computing*. PhD thesis, Universiti Teknologi Malaysia, 2014.

[7] Salil P Banerjee and Damon L Woodard. Biometric authentication and identification using keystroke dynamics: A survey. *Journal of Pattern Recognition Research*, 7(1):116–139, 2012.

[8] Alan Frank Blackwell. Hci as an inter-discipline. 2015.

[9] Ruud M Bolle, Jonathan H Connell, Sharath Pankanti, Nalini K Ratha, and Andrew W Senior. *Guide to biometrics*. Springer Science & Business Media, 2013.

[10] Richard A Bolt. *"Put-that-there": Voice and gesture at the graphics interface*, volume 14. ACM, 1980.

[11] C Burghart, O Schorr, S Yigit, N Hata, K Chinzei, A Timofeev, R Kikinis, H Wörn, and U Rembold. A multi-agent-system architecture for man-machine-interaction in computer aided surgery. In *Proceedings of the 16th IAR Annual Meeting*, pages 117–123, 2001.

[12] Joseph P Campbell. Jr (1997). speaker recognition: A tutorial. *Proc. IEEE*, 85(9).

[13] Davide Carneiro, José Carlos Castillo, Paulo Novais, Antonio Fernández-Caballero, and José Neves. Multimodal behavioral analysis for non-invasive stress detection. *Expert Systems with Applications*, 39(18):13376–13389, 2012.

[14] Rick Cattell. Scalable sql and nosql data stores. *Acm Sigmod Record*, 39(4):12–27, 2011.

[15] Joyce Y Chai, Pengyu Hong, and Michelle X Zhou. A probabilistic approach to reference resolution in multimodal user interfaces. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 70–77. ACM, 2004.

[16] Gong Chao. Human-computer interaction: process and principles of human-computer interface design. In *2009 International Conference on Computer and Automation Engineering*, pages 230–233. IEEE, 2009.

[17] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *ACM Sigmod record*, 26(1):65–74, 1997.

[18] Lawrence Shao-Hsien Chen and Thomas S Huang. *Joint processing of audio-visual information for the recognition of emotional expressions in human-computer interaction*. Citeseer, 2000.

[19] Byungchul Cho and Jong-Man Park. Technology review on multimodal biometric authentication. *The Journal of Korean Institute of Communications and Information Sciences*, 40(1):132–141, 2015.

[20] Ira Cohen, Nicu Sebe, Ashutosh Garg, Lawrence S Chen, and Thomas S Huang. Facial expression recognition from video sequences: temporal and static modeling. *Computer Vision and image understanding*, 91(1-2):160–187, 2003.

[21] Jill Dorrian, Nicole Lamond, and Drew Dawson. The ability to self-monitor performance when fatigued. *Journal of Sleep Research*, 9(2):137–144, 2000.

[22] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[23] Ken Ducatel, Union européenne. Technologies de la société de l'information, Union européenne. Institut d'études de prospectives technologiques, and Union européenne. Société de l'information conviviale. Scenarios for ambient intelligence in 2010. 2001.

[24] Ori Eisen. Systems and methods for detection of session tampering and fraud prevention, April 3 2012. US Patent 8,151,327.

[25] Michael W Eysenck and Mark T Keane. *Cognitive psychology: A student's handbook*. Psychology press, 2015.

[26] Beat Fasel and Juergen Luettin. Automatic facial expression analysis: a survey. *Pattern recognition*, 36(1):259–275, 2003.

[27] Pooja S Gandodhar and SM Chaware. Context aware computing systems: A survey. In *2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on*, pages 605–608. IEEE, 2018.

[28] Dariu M Gavrila. The visual analysis of human movement: A survey. *Computer vision and image understanding*, 73(1):82–98, 1999.

[29] Dirk Göger, Karsten Weiß, Catherina Burghart, and Heinz Wörn. Sensitive skin for a humanoid robot. In *Proceedings of the 2006 international conference on human-centered robotic systems*, 2006.

[30] Bijian Guo, Jiang Lin, and Qing Li. Anti-cheating method and system for online games, February 13 2014. US Patent App. 14/011,776.

[31] Vincent Hayward, Oliver R Astley, Manuel Cruz-Hernandez, Danny Grant, and Gabriel Robles-De-La-Torre. Haptic interfaces and devices. *Sensor Review*, 24(1):16–29, 2004.

[32] Hirro Iwata. History of haptic interface. In *Human haptic perception: Basics and applications*, pages 355–361. Springer, 2008.

[33] Alejandro Jaimes and Nicu Sebe. Multimodal human–computer interaction: A survey. *Computer vision and image understanding*, 108(1-2):116–134, 2007.

[34] Michael Johnston and Srinivas Bangalore. Matchkiosk: a multimodal interactive city guide. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, 2004.

[35] Zach Jorgensen and Ting Yu. On mouse dynamics as a behavioral biometric for authentication. In *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, pages 476–482. ACM, 2011.

[36] Arun Kejariwal, Sanjeev Kulkarni, and Karthik Ramasamy. Real time analytics: algorithms and systems. *Proceedings of the VLDB Endowment*, 8(12):2040–2041, 2015.

[37] Sanshzar Kettebekov and Rajeev Sharma. Understanding gestures in multimodal human computer interaction. *International Journal on Artificial Intelligence Tools*, 9(02):205–223, 2000.

[38] Oussama Khatib, Oliver Brock, Kyong-Sok Chang, Diego Ruspini, Luis Sentis, and Sriram Viji. Human-centered robotics and interactive haptic simulation. *The International Journal of Robotics Research*, 23(2):167–178, 2004.

[39] Hamzeh Khazaei, Marios Fokaefs, Saeed Zareian, Nasim Beigi-Mohammadi, Brian Ramprasad, Mark Shtern, Purwa Gaikwad, and Marin Litoiu. How do i choose the right nosql solution? a comprehensive theoretical and experimental survey. *Big Data and Information Analytics (BDIA)*, 2:1, 2016.

[40] Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. In *Advances in Neural Information Processing Systems*, pages 2280–2288, 2016.

[41] Eunhoe Kim and Jaeyoung Choi. A context management system for supporting context-aware applications. In *2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing*, volume 2, pages 577–582. IEEE, 2008.

[42] Scott Kim. Interdisciplinary cooperation. In *Readings in Human–Computer Interaction*, pages 304–311. Elsevier, 1995.

[43] Toshiyuki Kirishima, Kosuke Sato, and Kunihiro Chihara. Real-time gesture recognition by learning and selective control of visual interest points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):351–364, 2005.

[44] Bao Lin, Bodil Recke, Jørgen KH Knudsen, and Sten Bay Jørgensen. A systematic approach for soft sensor development. *Computers & chemical engineering*, 31(5-6):419–425, 2007.

[45] Stan Lipovetsky and Michael Conklin. Analysis of regression in game theory approach. *Applied Stochastic Models in Business and Industry*, 17(4):319–330, 2001.

[46] João Ricardo Lourenço, Bruno Cabral, Paulo Carreiro, Marco Vieira, and Jorge Bernardino. Choosing the right nosql database for the job: a quality attribute evaluation. *Journal of Big Data*, 2(1):18, 2015.

[47] Michael J Lyons, Michael Haehnel, and Nobuji Tetsutani. Designing, playing, and performing with a vision-based mouth interface. In *Proceedings of the 2003 conference on New interfaces for musical expression*, pages 116–121. National University of Singapore, 2003.

[48] L McCowan, Daniel Gatica-Perez, Samy Bengio, Guillaume Lathoud, Mark Barnard, and Dong Zhang. Automatic analysis of multimodal group actions in meetings. *IEEE transactions on pattern analysis and machine intelligence*, 27(3):305–317, 2005.

[49] Sven Meyer and Andry Rakotonirainy. A survey of research on context-aware homes. In *Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003-Volume 21*, pages 159–168. Australian Computer Society, Inc., 2003.

[50] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 2018.

[51] José Maia Neves, Manuel Filipe Santos, and José Manuel Machado. *Progress in Artificial Intelligence: 13th Portuguese Conference on Artificial Intelligence, EPIA 2007, Workshops: GAIW, AIASTS, ALEA, AMITA, BAOSW, BI, CMBSB, IROBOT, MASTA, STCS, and TEMA, Guimarães, Portugal, December 3-7, 2007, Proceedings*, volume 4874. Springer, 2007.

[52] NewZoo. eSports Global Economy. 2019. [Online; Acedido 10-Novembro-2019].

[53] Binh T Nguyen, Bryan D Wolf, and Brian Underdahl. Detecting and preventing bots and cheating in online gaming, August 8 2013. US Patent App. 13/752,027.

[54] J Nielsen. Response times: The three important limits. usability engineering. *San Francisco: Morgan Kaufmann Retrieved March*, 17:2007, 1994.

[55] Donald A Norman and Stephen W Draper. *User centered system design: New perspectives on human-computer interaction*. CRC Press, 1986.

[56] Adam J Overton. Cheater detection in a multi-player gaming environment, October 30 2007. US Patent 7,288,027.

[57] Sharon Oviatt. Multimodal interfaces. In *The human-computer interaction handbook*, pages 439–458. CRC press, 2007.

[58] Maja Pantic and Leon JM Rothkrantz. Automatic analysis of facial expressions: The state of the art. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (12):1424–1445, 2000.

[59] M Pavithra and KB Sri Sathya. Continuous user authentication using keystroke dynamics. *International Journal of Computer Science and Information Technologies*, 6(2):1922–1925, 2015.

[60] Siani Pearson and Andrew Norman. Method of validating performance of a participant in an interactive computing environment, April 22 2004. US Patent App. 10/632,135.

[61] Oudeyer Pierre-Yves. The production and recognition of emotions in speech: features and algorithms. *International Journal of Human-Computer Studies*, 59(1-2):157–183, 2003.

[62] André Pimenta, Davide Carneiro, José Neves, and Paulo Novais. A neural network to classify fatigue from human–computer interaction. *Neurocomputing*, 172:413–426, 2016.

[63] LR Rabiner and BH Juang. Fundamentals of speech recognition (prentice hall ptr. *Upper Saddle River, New Jersey*, 1993.

[64] Kenneth D Ray, James M Alkove, Lonny Dean McMichael, Nathan T Lewis, and Patrik Schnell. Trusted entity based anti-cheating mechanism, October 31 2017. US Patent 9,805,196.

[65] Judith A Ricci, Elsbeth Chee, Amy L Lorandeau, and Jan Berger. Fatigue in the us workforce: prevalence and implications for lost productive work time. *Journal of Occupational and Environmental Medicine*, 49(1):1–10, 2007.

[66] Gabriel Robles-De-La-Torre. The importance of the sense of touch in virtual and real environments. *Ieee Multimedia*, 13(3):24–30, 2006.

[67] Hataichanok Saevanee, Nathan L Clarke, and Steven M Furnell. Multi-modal behavioural biometric authentication for mobile devices. In *IFIP International Information Security Conference*, pages 465–474. Springer, 2012.

[68] Katie Salen, Katie Salen Tekinbaş, and Eric Zimmerman. *Rules of play: Game design fundamentals*. MIT press, 2004.

[69] Marten K Scheffers, Darryl G Humphrey, Robert R Stanny, Arthur F Kramer, and Michael GH Coles. Error-related processing during a period of extended wakefulness. *Psychophysiology*, 36(2):149–157, 1999.

[70] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *1994 First Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE, 1994.

[71] Chao Shen, Zhongmin Cai, Xiaohong Guan, Youtian Du, and Roy A Maxion. User authentication through mouse dynamics. *IEEE Transactions on Information Forensics and Security*, 8(1):16–30, 2012.

[72] Ben Shneiderman. 1.1 direct manipulation: a step beyond programming languages. *Sparks of innovation in human-computer interaction*, 17:1993, 1993.

[73] Ben Shneiderman, Catherine Plaisant, Maxine Cohen, Steven Jacobs, N Elmqvist, and N Diakopoulos. Designing the user interface: Strategies for effective human-computer interaction. 4th, 2005.

[74] Linda E Sibert and Robert JK Jacob. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pages 281–288. ACM, 2000.

[75] Michael E Smith, Linda K McEvoy, and Alan Gevins. The impact of moderate sleep loss on neurophysiologic signals during working-memory task performance. *Sleep*, 25(7):56–66, 2002.

[76] Michael Stonebraker. Sql databases v. nosql databases. *Communications of the ACM*, 53(4):10–11, 2010.

[77] Kalyan P Subbu and Athanasios V Vasilakos. Big data for context aware computing–perspectives and challenges. *Big Data Research*, 10:33–43, 2017.

[78] Dov Te'eni. An overview of fit conceptualizations in hci. *Human-Computer Interaction and Management Information Systems: Foundations*, 5:205, 2006.

[79] Cheng-Jung Tsai, Ting-Yi Chang, Yu-Ju Yang, Meng-Sung Wu, and Yu-Chiang Li. An approach for user authentication on non-keyboard devices using mouse click characteristics and statistical-based classification. *International Journal of Innovative Computing, Information and Control*, 8(11):7875–7886, 2012.

[80] Matthew George Tyler. Online gaming cheating prevention system and method, January 30 2007. US Patent 7,169,050.

[81] Matthew George Tyler. Online gaming cheating prevention system and method, January 29 2013. US Patent 8,360,890.

[82] Ying Wu and Thomas S Huang. Vision-based gesture recognition: A review. In *International Gesture Workshop*, pages 103–115. Springer, 1999.

[83] Roman V Yampolskiy and Venu Govindaraju. Behavioural biometrics: a survey and classification. *International Journal of Biometrics*, 1(1):81–113, 2008.

[84] Ping Zhang, Jane Carey, and Dov Te'eni. Human-computer interaction: Developing effective organizational information systems. *Te'eni, Dov, Jane Carey, & Ping Zhang, Human-Computer Interaction: Developing Effective Organizational Information Systems. John Wiley & Sons, Inc*, 2007.