INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Deteção de Fadiga através de uma abordagem baseada em eventos

HUGO JOSÉ GUEDES BENTO
Junho de 2022

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Deteção de Fadiga através de uma abordagem baseada em eventos

Hugo José Guedes Bento

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas de Informação e Conhecimento

Orientador: Isabel Praça

Co-orientador:

Júri:

Presidente:

Emanuel Silva, Professor adjunto, ISEP

Vogais:

Ana Faria, Professor adjunto, ISEP

Porto, Junho de 2022

Dedicatória

Este trabalho é dedicado aos meus pais, que em tudo me apoiaram nestes tempos tão difíceis, sem os quais nunca conseguiria completar o mesmo.

Resumo

O advento da Indústria 5.0, introduz na Europa uma abordagem centrada no ser humano, que pretende garantir o bem-estar dos trabalhadores nos processos produtivos, incluindo uma especial preocupação com a monitorização e gestão da sua saúde física e mental, recorrendo a novas tecnologias como as de interação homem máquina, os *Digital Twins* e a inteligência artificial. A literatura indica que se um trabalhador estiver cansado, é mais suscetível a cometer erros e tem uma pior performance, assim é importante avaliar o seu estado de fadiga.

Apesar de existirem várias investigações relacionando a fadiga com os tempos de reação em testes Go/NoGo, não existem detetores baseados neste sistema. Este trabalho reporta a criação de um detetor de fadiga baseado neste sistema, recorrendo à aprendizagem automática. Os dados recolhidos indicam uma maior correlação da tarefa NoGo com a fadiga, notando-se uma redução do tempo de reação para níveis de fadiga elevados. O modelo final, de previsão da fadiga em três classes apresenta uma taxa de acerto de 57%. Mas um modelo alternativo de deteção de fadiga muito elevada apresenta uma taxa de acerto de 84%, revelando uma adequação do teste Go/NoGo a este propósito.

Palavras-chave: Deteção de fadiga, Teste Go/NoGo, Aprendizagem automática

Abstract

The advent of Industry 5.0 introduces a human-centered approach in Europe, which aims to ensure the well-being of workers in production processes, including a special concern with the monitoring and management of their physical and mental health. Using new technologies such as human-machine interaction, Digital Twins and artificial intelligence. The literature indicates that if a worker is tired, he is more susceptible to making mistakes and has a worse performance, so it is important to assess his state of fatigue.

Although there are several investigations relating fatigue with reaction times in Go/NoGo tests, there are no detectors based on this system. This work reports the creation of a fatigue detector based on this system, using machine learning. The collected data indicates a greater correlation of the NoGo task with fatigue than the reaction time of the Go task, and a reduction in the reaction time for high levels of fatigue. The final model, predicting fatigue in three classes, has an accuracy of 57%. But an alternative model of very high fatigue detection has an accuracy of 84%, revealing the suitability of the Go/NoGo test for this purpose.

Keywords: Fatigue Detection, Go/NoGo testing, Machine Learning

Agradecimentos

São bastantes as pessoas sem as quais este projeto nunca poderia ter obtido estes resultados, pois a sua contribuição foi fundamental para o seu desenvolvimento.

Gostaria de agradecer principalmente à equipa do GECAD que colaborou no desenvolvimento deste projeto, sem os quais o mesmo não seria possível.

À minha orientadora do ISEP, a Doutora Isabel Praça, pela oportunidade de participar num projeto com um tema tão original e diferente de todos os que trabalhei no passado, e pela sua disponibilidade e orientação.

À Doutora Eva Maia, pela sua contribuição e sabedoria na apresentação de soluções e propostas de alternativas face aos problemas deparados.

Ao Doutor Orlando Sousa, pelo seu rigor, conhecimento e sugestões que contribui para o desenvolvimento do projeto.

Aos vários voluntários que contribuíram para a recolha de dados Go/NoGo, sem os quais este trabalho não seria possível.

Gostaria também de agradecer ao Instituto Superior de Engenharia do Porto que me tem acolhido durante mais de 16 anos, tendo sido quase como uma segunda casa. E ao Departamento de Engenharia Informática e a todos os seus docentes, que me ensinaram os segredos da arte da informática, de que tanto gosto.

Índice

Lista de Figuras

Lista de Tabelas

Lista de Código

Acrónimos e Símbolos

Lista de Acrónimos

AHP	*Analytic Hierarchy Process*
ANN	*Artificial Neural Network*
AUC	*Area Under the ROC Curve*
BPMN	*Business Process Model and Notation*
CRISP-DM	*Cross-Industry Standard Process for Data Mining*
EAR	*Eye Aspect Ratio*
ECG	Eletrocardiograma
EEG	Eletroencefalografia
EMG	Eletromiografia
EOG	Eletro-oculografia
ERP	*Event-Related Potentials*
FAST	*Function Analysis System Technique*
FN	*False Negatives*
FP	*False Positives*
FURPS+	*Functionality Usability Reliability Performance Supportability +*
GUI	G*raphical User Interface*
HRV	*Heart Rate Variability*
HTTP	*Hypertext Transfer Protocol*
HTTPS	*Hypertext Transfer Protocol Secure*
IP	*Internet Protocol*
ISEP	Instituto Superior de Engenharia do Porto
KNN	*k-nearest neighbors*
KSS	*Karolinska Sleepiness Scale*

MAR	*Mouth Aspect Ratio*
PERCLOS	*Percentage of Eyelid Closure Over the Pupil Over Time*
QEF	*Quantitative Evaluation Framework*
RA	Requisitos Adicionais
RAM	*Random Access Memory*
RC	Requisitos de Confiabilidade
RD	Requisitos de Desempenho
REST	*Representational State Transfer*
RF	Requisitos Funcionais
RGPD	Regulamento Geral sobre a Proteção de Dados
RNN	*Recurrent Neural Network*
ROC	*Receiver Operating Characteristic curve*
RS	Requisitos de suportabilidade
RU	Requisitos de usabilidade
SQL	*Structured Query Language*
SVM	S*upport Vector Machine*
TCP	*Transmission Control Protocol*
TN	*True Negatives*
TP	*True Positives*
UC	*Use Case*
USAFSAM	*United States of America School of Aerospace Medicine*
UML	*Unified Modeling Language*
VAS-F	V*isual Analogue Scale to evaluate Fatigue severity*

Lista de Símbolos

$\mathfrak{y}$	valor de PERCLOS
π	Pi

1 Introdução

Este trabalho apresenta a criação de um detetor de fadiga baseado em testes Go/NoGo, recorrendo a técnicas de aprendizagem automática. Neste capítulo introdutório é indicado o contexto do projeto, formulado o problema que se pretende resolver, são expostos os objetivos do trabalho e qual a abordagem usada. É também efetuado um breve resumo do estado da arte, indicado o método de investigação usado, e explicada a estrutura do documento.

1.1 Contexto

No âmbito da Indústria 5.0 (Breque, De Nul and Petridis, 2021) a Comissão Europeia centra o seu foco no ser humano, através do uso de tecnologias como as de interação homem máquina, dos *Digital Twins* e de inteligência artificial, e em uma maior importância do bem-estar dos trabalhadores nos processos produtivos. Existe uma especial preocupação com a saúde física e mental dos trabalhadores nos novos ambientes de trabalho cada vez mais digitais, e no uso das novas tecnologias na sua monitorização e gestão.

A fadiga é uma constante na sociedade moderna, fruto de longas horas de trabalho e elevadas espectativas profissionais e sociais. Cada vez mais dormimos menos e com menor qualidade, sendo que um terço dos trabalhadores americanos dorme menos que as 7 horas mínimas recomendadas (Caldwell *et al.*, 2019). A fadiga tem um preço na nossa sociedade com uma redução da qualidade de vida, e uma redução da performance profissional. Esta encontra-se relacionada com um aumento de acidentes no trabalho, viação e industriais, sendo indicada como uma das causas do maior acidente nuclear na América (Caldwell *et al.*, 2019). Apesar de todos sentirmos a fadiga, esta é difícil de definir e avaliar. Desde há vários anos que existe investigação científica no tema, sendo os dispositivos de deteção visual para monitorização da fadiga em condutores os que se encontram mais próximos do uso no cotidiano (Caldwell *et al.*, 2019). Existem vários outros métodos de deteção de fadiga, dos quais se destacam os baseados em sinais fisiológicos do corpo humano e os baseados em interpretação de gestos

ou movimentos, com destaque na interação com equipamentos informáticos. O uso da inteligência artificial com recurso a modelos de aprendizagem automática, é prevalente na área de investigação relacionada com a deteção de fadiga em humanos (Parekh, Shah and Shah, 2020).

Uma das consequências da fadiga é uma redução do tempo de resposta, especialmente na resposta a inibições. Os testes envolvendo tarefas Go/NoGo, que avaliam reações a estímulos em que a opção de inibição existe, são uma ferramenta muito usada em investigação das consequências da fadiga em neurologia e psicologia. Mas não é possível encontrar detetores de fadiga com base em métodos envolvendo testes Go/NoGo.

1.2 Problema

As pessoas são um componente integral de qualquer fábrica ou serviço, trabalhando diariamente para os suportar. Se um colaborador estiver cansado, não alerta, ou desmotivado, é mais suscetível a cometer erros, existindo ampla evidência (Caldwell *et al.*, 2019) que a fatiga tem um impacto negativo na performance de trabalhadores, especialmente na indústria. Assim é importante avaliar o seu estado de fadiga de uma forma expedita e não intrusiva.

Atualmente existem várias tecnologias para a deteção de fadiga, como a observação visual computadorizada utilizada nos sistemas de monitorização de condução, o reconhecimento de padrões de interação com rato e teclado, o monitoramento eletrofisiológico recorrendo a Eletroencefalografia (EEG) ou batimentos cardíacos.

Existem estudos da área da psicologia, que apoiados em cronometria mental, relacionam a fadiga com a diminuição de tempos de reação e o aumento das taxas de erro, recorrendo a testes como o Go/NoGo. Mas aparentemente não se encontra investigação que comprove a possibilidade do seu uso para a construção de um detetor de fadiga. Assim a determinação da sua aplicabilidade real com a criação de uma ferramenta de deteção de fadiga e o aprofundamento do conhecimento neste campo são uma mais-valia na investigação e deteção de fadiga. Sem a avaliação rigorosa de este tipo de ferramenta, nunca se poderá comprovar ou não a sua real aplicabilidade na deteção de fadiga.

1.3 Objetivos

O propósito deste projeto é a criação de uma aplicação que permita a avaliação do nível de fadiga com base em testes Go/NoGo, permitindo expandir o conhecimento nesta área, e a construção de um modelo que detete fadiga desta forma, em tempo útil.

Do problema entende-se que este detetor deve ser fácil de aceder, não deve perturbar a atividade normal dos trabalhadores, e deve ser rápido e fácil de usar. Mas estes propósitos são algo subjetivos, pelo que é necessário defini-los de forma a serem específicos, mesuráveis

e alcançáveis. A facilidade de acesso ao detetor pode ser providenciada pela sua disponibilização online. A rapidez do detetor pode ser convertida em algo concreto como unidades de tempo, indicando que o teste de fadiga deve ser possível efetuar num tempo inferior a um minuto. Este tempo poderia ser mais baixo, mas tendo em conta que um único ensaio Go/NoGo demora vários segundos, e que o sistema necessita de efetuar vários destes ensaios num teste, o intervalo de tempo de um minuto é bastante adequado. A fácil utilização do sistema de deteção é difícil de quantificar, pelo que como alternativa, pode-se especificar que quando questionadas, metade das pessoas que usam o sistema deverão indicar que este é fácil de utilizar. Por fim não devemos esquecer que o detetor deve detetar fadiga, assim a sua capacidade de deteção deverá ser pelo menos superior a um algoritmo de deteção aleatória, assim a sua precisão deverá ser superior a 50%, medida através da taxa de acerto.

Consequentemente, de um modo mais específico, é possível definir o objetivo principal do projeto como:

- A criação de um detetor de fadiga com base em testes Go/NoGo.

E objetivos mais particulares relacionados com a performance deste, como:

- O detetor deverá ser um software disponível online.
- A deteção deve ser efetuada num intervalo de tempo inferior a um minuto.
- Quando questionados, mais de metade dos utilizadores, devem indicar que o sistema é de fácil utilização.
- O detetor deve apresentar uma precisão superior a 50%.

1.4 Abordagem Preconizada

De forma a atingir os objetivos acima referidos, o projeto atravessou as seguintes fases:

- Análise e documentação do estado da arte de deteção de fadiga, com o foco no uso de testes Go/NoGo.
- Levantamento de requisitos e design do sistema.
- Design e desenvolvimento de um sistema de aplicações Web que permita efetuar testes Go/NoGo.
- Obtenção de um *dataset* experimental recorrendo à aplicação.
- Análise do dataset, desenvolvimento de modelos de classificação através de técnicas de aprendizagem automática, e avaliação da viabilidade do mesmo.

- Integração do modelo no sistema, de forma a possibilitar a deteção de fadiga.

A primeira fase passou pela pesquisa e leitura de artigos relevantes sobre fadiga, a sua deteção e o uso de testes Go/NoGo na deteção de fadiga. Incluindo a sua documentação em um estado da arte presente neste documento.

A segunda fase passou pelo levantamento dos requisitos necessários ao sistema para obter os objetivos propostos, e pelo design e documentação de uma solução capaz de os atingir.

A terceira fase englobou a criação de parte do sistema, nomeadamente os componentes que permitem efetuar os testes Go/NoGo, a gestão de utilizadores, e o armazenamento dos resultados numa base de dados.

Na quarta fase, com a ajuda de alguns voluntários, foi criado um *dataset* de testes Go/NoGo e a sua relação com a fadiga, usando como ferramenta o sistema criado na fase anterior.

A quinta fase protagonizou a análise dos dados recolhidos através de ferramentas estatísticas, e a criação de vários modelos de deteção com base em tecnologias de aprendizagem automática, possibilitando a avaliação e seleção do modelo de deteção mais adequado.

Na sexta e última fase, o modelo criado foi integrado no sistema possibilitando a deteção de fadiga por parte dos utilizadores.

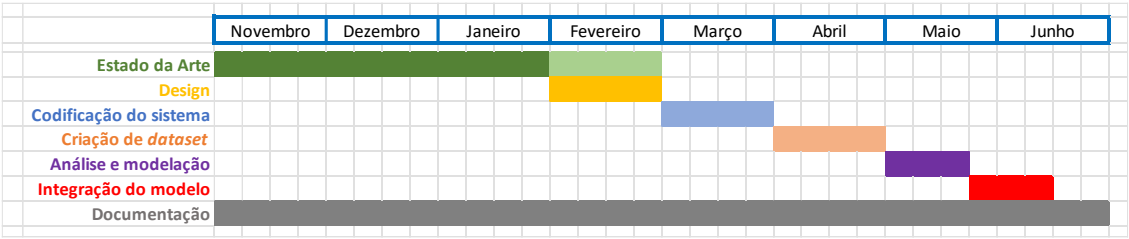

Figura 1 – Planeamento do trabalho

Na Figura 1 é possível verificar o plano do trabalho desenvolvido para o sistema, notando-se que o estado da arte e codificação do sistema apesar de serem apresentados em blocos isolados e definidos, sofreram atividades de melhoria, inerentes a um projeto que envolve o desenvolvimento de software. O estado da arte referente às várias tecnologias utilizadas, foi desenvolvido em paralelo com o design da solução.

1.5 Resumo do estado da arte

A fadiga é um fenómeno complexo que cria problemas de relevo na sociedade, devido às altas exigências profissionais, horários de trabalho longos e inconstantes, pressões sociais, disrupções do ritmo circadiano e descanso inadequado (Caldwell *et al.*, 2019). Segundo

(Matthews *et al.*, 2012) apesar da fadiga ser algo que nos afeta a todos, a sua definição é difícil, não existindo diferença clara entre fadiga e sonolência.

A fadiga tem um impacto na sociedade que abrange desde o abrandar da performance cognitiva ao aumento do número de acidentes de trabalho industriais, de viação, de aviação e nucleares (Caldwell *et al.*, 2019). Mas a fadiga pode também causar efeitos físicos, emocionais, e comportamentais (Parekh, Shah and Shah, 2020).

No estudo de deteção de fadiga existem três escalas que se destacam pelo seu uso comum em investigação. A escala de fadiga mental USAFSAM (*United States of America School of Aerospace Medicine*) (Samn and Perelli, 1982), a Karolinska Sleepiness Scale (KSS) (Shahid *et al.*, 2011), e a Visual Analogue Scale to evaluate Fatigue severity (VAS-F) (Lee, Hicks and Nino-Murcia, 1991).

Na literatura denotam-se vários tipos de deteção de fadiga, todos recorrendo a inteligência artificial. Uma das modalidades é a deteção visual muito usada na deteção contínua em condutores, que recorre ao processamento de imagem para analisar o piscar de olhos, o fechar dos mesmos e expressões faciais como o bocejar, obtendo-se por norma uma taxa de acerto (*accuracy*) por volta dos 90% (Laouz, Ayad and Terrissa, 2020).

Outro campo de deteção comum é o uso de sinais fisiológicos, pelo que segundo (Kolodziej *et al.*, 2020) os sinais mais usados são os obtidos através de EEG, da Eletro-oculografia (EOG), da Eletromiografia (EMG), de Eletrocardiograma (ECG) que mede a função cardíaca, e através da variabilidade da frequência cardíaca (HRV). Neste campo, os níveis da taxa de acerto também rondam os 90%. Com o trabalho de (Zhang *et al.*, 2021) a reportar 89% recorrendo a uma Recurrent Neural Network (RNN). E o trabalho de (Qin *et al.*, 2021) a referir 91.8% para *support vector machine* (SVM), 87.3% para *k-nearest neighbors* (KNN), e 87.5% para a árvore de decisão.

O uso de movimentos ou gestos também é outro campo de estudo na deteção de fadiga. O trabalho de (Carneiro *et al.*, 2017) apresenta um detetor com base em padrões de interação com rato e teclado acoplados com movimentos na cadeira, cujo classificador baseado em uma Artificial Neural Network (ANN) obtém uma taxa de acerto de 81%. Outro detetor, baseado apenas em dinâmicas de uso de teclado, segundo (Ulinskas *et al.*, 2018), apresenta uma taxa de acerto de 98% recorrendo a SVM. O uso de teclados virtuais em telemóveis também pode ser usado para detetar fadiga segundo (Al-Libawy *et al.*, 2017), que reporta taxas de acerto de 77% para ANN e 88% para SVM.

A cronometria mental, o estudo de tempos de resposta ou reação em tarefas, pode ser usada para estudar a fadiga, estando demonstrado que esta aumenta os tempos de reação e taxas de erro em tarefas, incluindo segundo (Guo *et al.*, 2018) uma clara diminuição da velocidade de resposta a decisões de inibição. Dos fatores que afetam os tempos de reação, encontram-se como fator principal a idade, não existindo para (Woods *et al.*, 2015) evidencia do efeito do

sexo ou educação. Mas a experiencia de (M. Guo *et al.*, 2016) que reforça o impacto da idade, indica que o sexo do sujeito pode afetar os tempos de reação.

O trabalho de (Woods *et al.*, 2015) sobre o registo de tempos de reação, revela várias preocupações, de onde se destacam a visibilidade do estimulo que despoleta a resposta, e os atrasos criados pelo hardware como o rato e equipamento de visualização, e pelo software a nível de interrupções.

Uma das ferramentas usadas na investigação dos mecanismos e impactos da fadiga são os testes Go/NoGo, uma tarefa onde um sujeito deve reagir a um estímulo Go ou inibir-se de reagir a outro estímulo NoGo. A reação passa por normalmente pressionar um botão, sendo os tempos de reação, número de falhas e falsos alarmes relacionados com NoGo registados. Estando provado que estes valores aumentam à medida que a fadiga se acumula (Magnuson, Doesburg and McNeil, 2021) (Guo *et al.*, 2018). Apesar de ambos os autores apresentarem estudos de fadiga que usam a tarefa Go/NoGo, apresentando o impacto da mesma nesta tarefa, não foi encontrado na literatura qualquer estudo ou trabalho relacionado com a relação inversa, o uso de Go/NoGo para detetar a fadiga.

1.6 Método de Investigação

Esta tese segue a metodologia de investigação baseado na Investigação Ação, pelo que é importante indicar qual a hipótese ou pergunta que pretendemos responder tendo como base o problema a resolver. Assim pretendemos responder à seguinte questão:

"Será possível criar um detetor de fadiga usando testes Go/NoGo e aprendizagem automática?"

A resposta a esta pergunta será o resultado da nossa investigação, uma solução de software que com base em testes Go/NoGo e aprendizagem automática, permite detetar a fadiga de um determinado sujeito.

O método de investigação usado é o *Action Research* ou Investigação Ação, baseado no trabalho de (Staron, 2020). Este método foi escolhido devido à sua abertura à mudança e natureza iterativa, sendo constituído por um ciclo de cinco fases: o diagnóstico, o planeamento de ações, a tomada de ação, a avaliação, e a aprendizagem. Existe neste método um foco no uso de diagramas e ferramentas de visualização de dados, de forma a guiar a discussão do campo especulativo para a análise de dados, o que é uma mais-valia neste projeto.

A validação da solução atingida é obtida recorrendo a framework de avaliação da qualidade, que nos indica a qualidade do software produzido, e pela análise da precisão dos algoritmos de aprendizagem automática na deteção de fadiga através da taxa de acerto.

A apresentação e disseminação da pesquisa resultante é efetuada através da criação desta tese de mestrado que será apresentada a um painel de avaliação do Mestrado de Engenharia Informática do ISEP e posteriormente acessível publicamente.

1.7 Estrutura do documento

Este documento encontra-se estruturado em vários capítulos, sendo o presente capítulo uma introdução ao mesmo apresentando o contexto, o problema, os objetivos, a abordagem preconizada, um breve resumo do estado da arte, e o método de investigação.

O segundo capítulo, o Estado da Arte, apresenta um estado da arte de deteção de fadiga e do uso de testes Go/NoGo, reportando artigos de relevo sobre o tema, que apoiam a especificação da solução. Aborda também a temática da aprendizagem automática, e as tecnologias e *frameworks* necessárias ao projeto.

O terceiro capítulo, a Análise de Valor, apresenta a análise efetuada para maximizar o valor gerado por este projeto. Incluindo uma análise e identificação da oportunidade, os principais valores criados, e uma proposta de valor de Osterwalder. O método de análise hierárquica é também usado neste capítulo para identificar qual a escala de fadiga que maior valor traz ao projeto.

O quarto capítulo, a Análise de Requisitos e Design Arquitetural, apresenta os requisitos funcionais e não funcionais identificados para o sistema, sendo também efetuado um design da solução apresentando várias alternativas recorrendo a vários diagramas elucidativos de 4+1 vistas. No final deste capítulo também são avaliadas as *frameworks* a usar, sendo selecionadas as ideais.

O quinto capítulo, a Solução Implementada, descreve a solução que foi implementada para obter os objetivos do projeto, descrevendo as suas particularidades.

O sexto capítulo, Experimentação e Avaliação, descreve o método de investigação utilizado, a Investigação Ação. Sendo indicado a hipótese ou pergunta inicial, o processo de validação da solução, através de uma avaliação do sistema recorrendo a *framework* de avaliação de qualidade. São descritos o design inicial da recolha de dados e a forma de disseminação deste trabalho.

O sétimo capítulo, Dataset e Modelos, descreve em detalhe o processo real de recolha de dados, a sua preparação e análise exploratória. São apresentados os modelos e a sua comparação, sendo por fim explicada a sua implementação.

O último capítulo, a Conclusão, apresenta a discussão do trabalho, as perspetivas de trabalho futuro a desenvolver, e as conclusões finais do trabalho que inclui uma revisão dos objetivos que pretendíamos atingir com a solução.

2 Estado da Arte

Este capítulo descreve o levantamento efetuado do estado da arte, iniciando pelo relato do foco da Comissão Europeia numa abordagem centrada no ser humano e no seu bem-estar, no contexto da Indústria 5.0. Prossegue com o tema da fadiga e das suas consequências, seguido das principais escalas em uso na sua deteção. Continua com um enumerar das tecnologias existentes para a deteção, e por fim pela aplicação de cronometria mental na deteção de fadiga através de testes Go/NoGo. Fornece também informação sobre a aprendizagem automática, e a metodologia CRISP-DM usada em ciência de dados. Por fim aborda as tecnologias e *frameworks* relevantes ao projeto, nomeadamente as de *frontend*, as de aprendizagem automática e as legais.

O levantamento do estado da arte iniciou-se por uma pesquisa no website da biblioteca do conhecimento online, usando várias palavras-chaves. As palavras "*fatigue detection*" foram as que mais resultados obtiveram, nomeadamente a nível da deteção de fadiga de condutores, mas não apresentaram resultados referentes a deteção com cronometria mental. O segundo conjunto de palavras utilizadas foram "*event based fatigue detection*", que proporcionaram poucos resultados de relevo. Face à dificuldade de encontrar artigos de interesse relativos ao uso de cronometria mental na deteção de fadiga, procedeu-se a uma pesquisa mais aleatória, por vezes baseada em autores proeminentes dos artigos encontrados. A análise dos artigos, baseou-se numa fase inicial apenas pela leitura do título e resumo dos mesmos de forma a determinar a sua relevância ou não para o tema, tendo em conta o ano de publicação. Seguida de uma leitura integral dos mais relevantes.

Não tendo sido encontrado artigos diretamente relacionados com a construção de um detetor de fadiga com uso da tarefa Go/NoGo, optou-se por não limitar a pesquisa aos últimos cinco anos, de forma a encontrar os artigos com informação relevante a sua construção.

2.1 A União Europeia e a Indústria 5.0

A Indústria na União Europeia é forte e altamente competitiva, mas faz face a desafios constantes resultantes da uma cada vez mais complexa economia global. De forma à indústria europeia poder continuar a trazer prosperidade à Europa, esta necessita de se adaptar continuamente, o que só é possível através de uma constante inovação. Esta inovação tem de provir na sua grande maioria da aplicação das cada vez mais avançadas tecnologias digitais como a interconectividade de sensores, e a automatização de soluções de *big data* e inteligência artificial. Esta transformação não afeta apenas as fábricas, pois o seu impacto será sentido também na sociedade.

Segundo a comissão europeia (Breque, De Nul and Petridis, 2021), a Indústria 5.0 não deve ser vista como uma continuação cronológica ou paradigma alternativo ao existente da Indústria 4.0, mas como um seu complemento. Esta enfatiza aspetos que serão centrais à indústria na sociedade europeia, fatores não apenas económicos ou tecnológicos, mas com importantes dimensões sociais e ambientais. Seis categorias estão identificadas na estrutura tecnológica que suporta a Indústria 5.0: as interações homem máquina individualizadas; as tecnologias e *smart materials* inspiradas na biologia; a simulação e os Digital Twins; as tecnologias de transmissão armazenagem e análise de dados; a inteligência artificial; e as tecnologias relacionadas com a eficiência, renovável, armazenamento e autonomia energética. De forma a ilustrar a crescente importância das tecnologias digitais, note-se que em 2009 apenas a Microsoft fazia parte do top 10 das companhias em bolsa. Em 2019 as cinco primeiras companhias deste ranking eram todas tecnológicas (Microsoft, Amazon, Apple, Alphabet, Facebook), segundo indica a comissão europeia. Assim os projetos devem olhar para o impacto dos ambientes de trabalho digitais nas condições de trabalho, segurança, saúde física e mental, e satisfação dos trabalhadores, segundo a perspetiva da Indústria 5.0.

Em vez de examinar o potencial para aumentos de eficiência das tecnologias emergentes, a abordagem centrada no ser humano da Indústria 5.0 coloca em primeiro lugar os interesses e necessidades humanos no processo produtivo. Em vez de o trabalhador se adaptar à nova indústria, esta deve usar a tecnologia para adaptar os processos às necessidades do trabalhador, servindo de guia e fonte de treino do mesmo. Colocando o seu bem-estar no centro do processo produtivo, criando um ambiente de trabalho seguro e benéfico para todos. A segurança dos trabalhadores não se refere apenas ao especto da saúde física, mas também à sua saúde mental. Existem novos riscos relacionados com as novas tecnologias, como os de esgotamento derivados da cultura digital de híper disponibilidade online e laboral. As tecnologias digitais podem ser usadas para suportar os trabalhadores, gerindo os riscos e impactos dos novos ambientes de trabalho na sua saúde, através da monitorização e alertas sobre o seu estado (Breque, De Nul and Petridis, 2021).

2.2 Fadiga

A fadiga é um fenómeno complexo que cria problemas de relevo na sociedade moderna, muito devido às altas exigências profissionais, longos e inconstantes horários de trabalho, pressões sociais, disrupções do ritmo circadiano e descanso inadequado (Caldwell *et al.*, 2019). Segundo este autor a fadiga pode ser definida como um sentimento de cansaço ou sonolência, resultante de sono insuficiente, de um prolongado ou monótono esforço que pode ser físico ou mental, ou de longos períodos de ocorrência de stress ou ansiedade.

Outra definição é a que surge em (Parekh, Shah and Shah, 2020), que indica que a fadiga é o estado psicobiológico causado por longos períodos de atividade cognitiva forte.

O trabalho de (Matthews *et al.*, 2012) indica que a fadiga é algo que todos sofremos, pelo que a definição do seu conceito deveria ser fácil, mas na realidade a sua definição apresenta uma miríade de problemas. Mesmo assim sugere a definição de fadiga como algo que todos nós conhecemos. Referindo uma redução da capacidade e vontade de reagir, caracterizada por um cansaço e uma aversão à continuação de trabalho, acompanhada por um desejo de descanso (p63). Este mesmo trabalho indica que a diferença entre fatiga e sonolência não é clara, muito devido à inexistência de consenso quer na definição de sonolência, quer na definição de fatiga. Sendo os termos usados indiferentemente, notando que são os investigadores na área do sono que se interessam por distinguir estes dois conceitos (pp 131-132).

De acordo com (Caldwell *et al.*, 2019), apesar de os especialistas recomendarem 7 a 9 horas de sono durante a noite, mais de um terço dos cidadãos trabalhadores dos Estados Unidos da América dorme menos de 7 horas por noite. Com uma redução do tempo médio de sono desta população de 1985 para 2012 de 7.4 para 7.18 horas. Segundo este mesmo autor o sono inadequado e a sua consequência, a sonolência no trabalho, são prevalentes entre adultos afetando a sua saúde, humor e performance.

A fadiga tem um impacto nos humanos e na sociedade em geral. Segundo (Caldwell *et al.*, 2019) tem a capacidade de degradar a performance cognitiva humana, fruto de longas horas de tarefas repetitivas e aborrecidas como a condução em autoestrada ou a monitorização de equipamentos. Em motoristas de pesados, leva a um aumento do risco de acidentes rodoviários. Em pilotos, a sua performance diminui à medida que a fadiga aumenta função da duração do voo. Em trabalhadores industriais, turnos prolongados levam a uma diminuição do estado de alerta, a uma degradação da performance, e a um maior risco de acidentes. Na indústria nuclear, o trabalho extraordinário encontra-se relacionado com o aumento de acidentes. Sendo que o pior acidente nuclear dos Estados Unidos da América, o acidente de Three Mile Island, foi parcialmente atribuído a fadiga (Caldwell *et al.*, 2019).

Segundo (Parekh, Shah and Shah, 2020) a fadiga pode causar efeitos físicos, emocionais e comportamentais. Descrevendo impactos emocionais como fúria, irritabilidade, ansiedade, apatia, depressão e falta de motivação. Impactos físicos, com dores de cabeça e corpo, fadiga

crônica, insónia, palpitações e mudanças de apetite. Efeitos comportamentais levando a dificuldades de pensamento como a confusão, dificuldades de concentração, esquecimento e falta de memória entre outros. Sendo a fadiga também refletida a nível social quer nas relações pessoais, quer a nível de trabalho. Onde origina a falha de prazos, baixa performance, falta de entusiasmo e a absência.

No entanto na sua introdução (Matthews *et al.*, 2012) indica que 'apesar de muitos estudos demonstrarem os efeitos adversos da fadiga, às vezes indivíduos que aparentam estar altamente fatigados continuam a apresentar níveis normais de performance' (p3).

Se os efeitos da fadiga acima mencionados sugerem uma necessidade em a controlar, a afirmação anterior revela a importância de uma medição confiável da mesma.

Segundo (Caldwell *et al.*, 2019) a investigação em deteção e monitorização de fadiga e em dispositivos de teste existe desde há muitos anos. Sendo os dispositivos montados em veículos para a deteção visual em condutores os que aparentam ser mais praticáveis, devido a disponibilidade de espaço, à existência de uma fonte de energia, e de os movimentos do condutor se encontrarem restringidos com o foco em uma única tarefa. Mas que apesar de mais de três décadas de investigação o sucesso da sua implementação operacional continua questionável.

Mas existem outros métodos para detetar a fadiga, (Parekh, Shah and Shah, 2020) menciona os mais importantes, além dos usados em condutores focados principalmente na deteção visual. Este refere o uso de sinais fisiológicos como a EEG e o batimento cardíaco, e a interpretação de gestos e movimentos através das dinâmicas de uso de teclados e ratos. Conforme este autor revela, e a nossa pesquisa mais à frente demonstra, o uso da inteligência artificial é prevalente neste campo com métodos de aprendizagem automática recorrendo a algoritmos como o SVM, o KNN e a ANN.

2.2.1 Escalas de fadiga

De forma a ser possível comparar a fadiga entre indivíduos, deve existir uma métrica comum, uma escala de fadiga. Existem várias destas escalas de fadiga, mas durante a pesquisa efetuada existiram três que se destacaram como as mais comuns na área. A escala de fadiga mental USAFSAM, a KSS e VAS-F. Estas escalas são por regra aplicadas, pedindo ao sujeito que indique qual o valor em que o seu nível de fadiga se encontra, o que conforme indica (Laouz, Ayad and Terrissa, 2020) origina uma subjetividade das escalas na sua avaliação da fadiga.

A escala de fadiga mental USAFSAM (Samn and Perelli, 1982) foi criada para avaliar a fadiga mental em tripulações de aviões pela força aérea dos Estados Unidos da América. Para além da escala de fadiga mental os autores também apresentam uma escala de carga de trabalho. A escala é composta por sete níveis descritos por frases claras e curtas que passamos a citar.

1. Fully alert. Wide awake. Extremely peppy.

2. Very lively. Responsive, but not at peak.
3. Okay. Somewhat fresh.
4. A little tired. Less than fresh.
5. Moderately tired. Let down.
6. Extremely tired. Very difficult to concentrate.
7. Completely exhausted. Unable to function effectively. Ready to drop

A KSS é uma escala de fadiga, mas na vertente de sonolência, sendo utilizada para avaliar o estado físico-psicológico de um sujeito nos últimos 10 minutos (Shahid *et al.*, 2011). Segundo este autor, ela é composta por 10 pontos, mas existem outras representações com nove pontos e apenas cinco entradas (Åkerstedt and Gillberg, 1990) correspondentes aos pontos 1, 3, 5, 7 e 9. A sua versão de dez pontos é a que segue.

1. Extremely alert
2. Very alert
3. Alert
4. Rather alert
5. Neither alert nor sleepy
6. Some signs of sleepiness
7. Sleepy, but no effort to keep awake
8. Sleepy, but some effort to keep awake
9. Very sleepy, great effort to keep awake, fighting sleep
10. Extremely sleepy, can't keep awake

A VAS-F (Lee, Hicks and Nino-Murcia, 1991) é uma escala que consiste de 18 itens relacionados com fadiga e energia, cada um acompanhado de uma linha, onde o sujeito indica na linha a sua proximidade de se rever ou não naquele item. A escala na sua conceção era composta por 37 itens, mas após validação e experiências envolvendo 75 sujeitos foi reduzida para 18. Fazem parte destes itens termos como *tired, drowsy, active, desire to close my eyes* entre outros. Os criadores indicam que os 18 itens da escala podem ser preenchidos com um esforço e tempo mínimo, sem no entanto, indicarem valores concretos.

2.2.2 Deteção de fadiga de condutores

Uma das áreas muito exploradas na deteção de fadiga é a deteção de fadiga e sonolência de condutores, existindo múltiplas publicações sobre este tema, especialmente focadas no uso de deteção visual.

O trabalho de (Laouz, Ayad and Terrissa, 2020) revela que existem inúmeros acidentes de viação, chegando a morrer mais de 1 milhão de pessoas anualmente nos Estados Unidos da América. Indica que mais de 1500 dessas mortes se devem ao resultado direto da sonolência dos condutores, e que na Alemanha um quarto das mortes em acidentes de viação são o resultado da sonolência momentânea do condutor.

Segundo este mesmo trabalho, existem duas maneiras principais de detetar sonolência: medições comportamentais do condutor e medições de sinais fisiológicos. As medições comportamentais do condutor observam o foco do condutor na condução através dos movimentos da cabeça, piscar de olhos, bocejar e expressões faciais, recorrendo principalmente a tecnologias de processamento de imagem. As medições de sinais fisiológicos como o nome indica focam-se em medir sinais fisiológicos como o ritmo cardíaco através de ECG ou a atividade eletrodérmica, sendo precisos na deteção e permitem efetuar-lha precocemente, ajudando a evitar acidentes. Após a deteção de um estado anormal do condutor, este é alertado pelo sistema de forma a focar a sua atenção de novo na estrada (Laouz, Ayad and Terrissa, 2020).

Tabela 1 – Comparação de deteção visual de fadiga (Laouz, Ayad and Terrissa, 2020)

Métrica	**Sensor**	**Método de classificação**	**Taxa de acerto**
Piscar de olhos	Câmara	Neural network based *model*	94%
Piscar de olhos	Câmara	*Eye aspect ratio*	Elevada
Piscar de olhos	Câmara	*Hierarchical Multiscale Long Short-Term Memory network*	80%+
PERCLOS	Câmara com infravermelhos	SVM	99%
Pupila (PERCLOS)	Câmara com infravermelhos	*Ratio of eye height and eye-width*	92%
Deteção de bocejos na face	Câmara	*Circular Hough transform*	98%
Bocejos	Câmara	*Ratio of mouth-height and mouth width*	94%
Fusão entre deteção de boca e olhos	Câmara	*Multi-task convolutional neural network + Optical flow*	97%

O uso de deteção visual de fadiga através do processamento de imagem é bastante comum, especialmente na sua deteção em condutores, sendo usadas duas principais métricas de acordo com (Laouz, Ayad and Terrissa, 2020), a análise do estado do olho e a deteção de bocejos. Estes métodos apresentam em geral uma elevada taxa de acerto (*accuracy*) superior a 90%, conforme visível na Tabela 1 do mesmo autor.

Na deteção do estado do olho, é importante detetar se o olho se encontra aberto ou fechado, para isso é usado o *Eye Aspect Ratio (EAR*) que indica o rácio entre a altura do olho e a sua largura. Assim é possível dizer que o EAR representa uma estimativa do estado de abertura do olho, sendo calculado com base em seis pontos conforme visível na Figura 2 extraída do trabalho de (Chandiwala and Agarwal, 2021). Este rácio é sempre calculado para ambos os olhos separadamente.

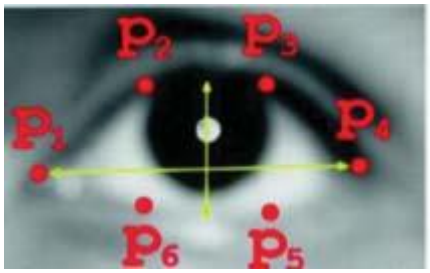

Figura 2 – EAR - *Eye Aspect Ratio* (Chandiwala and Agarwal, 2021)

O *Percentage of Eyelid Closure Over the Pupil Over Time* (PERCLOS) é a métrica usada para avaliar o estado da pupila ao longo do tempo, indicando a percentagem de tempo que o olho se encontra fechado. Segundo (Laouz, Ayad and Terrissa, 2020) o PERCLOS está testado como sendo o parâmetro mais preciso para deteção de fadiga através do olho. Sendo que o valor de PERCLOs pode ser calculado através da seguinte formula:

$$ŋ = \left[{(t3 - t2)} \Big/ {(t4 - t1)} \right] x\ 100\% \qquad (1)$$

,onde ŋ representa o valor de PERCOLOS, sendo t1 o tempo que demorou os olhos desde totalmente aberto até 80% aberto, t2 o tempo desde 80% aberto até 20% aberto, t3 o tempo que os olhos demoraram desde 20% aberto até fechar completamente e abrir de novo até 20% aberto, e t4 o tempo que os olhos demoraram a abrir de 20% a 80% aberto (Shen *et al.*, 2012).

A deteção visual de bocejos pode ser efetuada utilizando a métrica *Mouth Aspect Ratio* (MAR), desde que a pessoa observada não tenha o habito de colocar a mão à frente da boca quando boceja (Laouz, Ayad and Terrissa, 2020). Segundo (Chandiwala and Agarwal, 2021), e conforme visível na Figura 3 do seu trabalho, o MAR refere-se ao rácio entre a altura e a largura da boca, calculada com base em oito pontos.

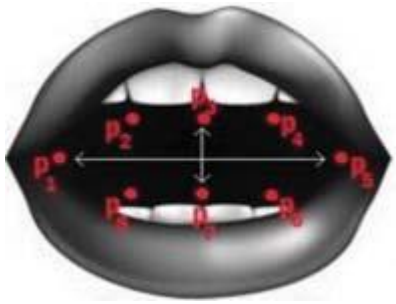

Figura 3 – MAR - *Mouth Aspect Ratio* (Chandiwala and Agarwal, 2021)

No sistema de deteção visual de fadiga desenvolvido por (Chandiwala and Agarwal, 2021) para condutores, são avaliados ambos o EAR e o MAR. Um alarme é despoletado caso o EAR sofra uma redução superior a 20% do valor inicial durante mais de 48 imagens, alertando o condutor. O condutor é considerado como estando a bocejar (uma indicação de cansaço) caso o valor de MAR ultrapasse o limiar de 20 durante mais de 25 imagens. Para referência o detetor apresenta valores de MAR de cerca de 2 para um sujeito com a boca fechada e de 33 para o mesmo sujeito a bocejar. Segundo o autor o sistema apresentou leituras corretas de situações de fadiga em 9 de 10 tentativas.

O trabalho de (Girish *et al.*, 2020) apresenta um detetor de fadiga de condução apoiado nas bibliotecas de programação OpenCV e dlib, que através do EAR e da distancia entre lábios alerta para potenciais situações de fadiga. Neste caso o alerta é acionado para valores de EAR inferiores a 0.3 mantidos durante mais de 50 imagens ou para valores de distância entre lábios superiores a 30, não sendo indicado as unidades ou a duração.

2.2.3 Deteção de fadiga com base em sinais fisiológicos

A deteção de fadiga também pode ser efetuada através da monitorização de sinais fisiológicos. Segundo o trabalho de (Kolodziej *et al.*, 2020) os sinais mais usados são os obtidos através de EEG que mede atividade elétrica do cérebro, de EOG que mede movimentos dos olhos, da EMG, do ECG que mede a função cardíaca, e através da HRV.

O detetor criado nesse trabalho recorre a EOG para obter informação sobre o piscar de olhos, focado em três componentes, a sua amplitude, a sua duração e o intervalo entre cada piscar de olhos. Nas experiências os sinais EOG eram recolhidos continuamente enquanto os sujeitos elaboravam uma tarefa computorizada exaustiva relacionada com memória, sendo o estado de fadiga dos sujeitos avaliado previamente e posteriormente com recurso a escala subjetiva *VAS-F*. O sistema de classificação não especificado de aprendizagem automática criado para o detetor permitiu obter uma taxa de acerto (*accuracy*) de 89 a 93% na deteção de fadiga.

Um exemplo do uso de EEG é o trabalho de (Zhang *et al.*, 2021) que apresenta um detetor de fadiga usando este tipo de sinal e com uma taxa de acerto de classificação de fadiga de 89% recorrendo a uma RNN. O classificador foi treinado com um dataset de sinais EEG recolhidos durante uma tarefa de condução em realidade virtual num cenário noturno com uma duração

de 90 minutos. O veículo deslocava-se a uma velocidade de 100 km/h sendo o condutor solicitado a mudar de faixa de rodagem a cada 6 a 10 segundos.

O estudo de (Qin *et al.*, 2021) indica que a recolha e análise de sinais da HRV é um processo simples que pode ser aplicado a deteção de fadiga mental, pelo que propõem um sistema robusto de deteção multimodal envolvendo a análise da HRV e dos movimentos dos olhos, para pilotos de aviões. As tarefas utilizadas no estudo foram a pilotagem de um avião em modo de navegação de cruzeiro em um simulador de voo durante 90 minutos. Numa variante a cada dois minutos ocorria um ataque por outro avião, necessitando o participante de carregar em um botão imediatamente, sendo os tempos de reação e falsos alarmes tidos em conta durante a tarefa. Outra variante envolvia combate simulado. Sinais de HRV e de movimentos dos olhos foram recolhidos continuamente durante a tarefa, sendo a cada 15 minutos aplicada uma escala VAS-F recorrendo a um telemóvel. Os métodos de classificação utilizados foram SVM, KNN, e árvores de decisão, sendo o método SVM o que obteve a maior taxa de acerto com um valor de 91.8%, seguido de 87.5% para a árvore de decisão e de 87.3% para KNN.

2.2.4 Deteção de fadiga com base em movimentos ou gestos

Outra forma de deteção de fadiga pode ser obtida analisando os gestos ou movimentos de sujeitos, existindo vários trabalhos sobre esta temática relacionados com a interação com computadores.

O trabalho de (Ribeiro, 2018) representa uma abordagem não invasiva ou intrusiva de deteção de fadiga mental recorrendo principalmente a interação com computadores. O sistema proposto pelo mesmo em (Pimenta *et al.*, 2016) e desenvolvido em (Carneiro *et al.*, 2017) apresenta um detetor de fadiga baseado em padrões de interação de utilizadores com o teclado e rato de um computador e nos movimentos das suas cadeiras. As cadeiras possuíam acelerómetros capazes de medir nos três eixos, sendo os dados recolhidos continuamente e compilados por séries de uma hora em conjunto com uma indicação da fadiga do utilizador, auto reportada recorrendo a escala de fadiga mental USAFSAM. Do teclado eram recolhidas informações sobre a velocidade ou ritmo de escrita, número de erros, quantidade de vezes que uma tecla era premida. Do rato eram obtidas métricas de velocidade, aceleração, distancia entre *clicks*, duração de *clicks* e excesso de distância viajada. O sistema de classificação selecionado foi uma ANN que operava na linguagem de programação R com uma taxa de acerto (*accuracy*) de 81%. Em (Pimenta *et al.*, 2016), que contém informação mais detalhada, é explicado que a ANN inicialmente utilizada era capaz de classificar os sete estados da escala de fadiga, tendo sido treinada com base num *dataset* com 74 entradas de fadiga e suas informações correspondentes. Os autores ressalvam que os 19% de previsões mal classificadas apresentavam valores vizinhos dos corretos.

O trabalho de (Ulinskas *et al.*, 2018) apresenta um sistema de deteção de fadiga de trabalhadores de escritório com base em dinâmicas de uso de teclado. O sistema capturava

cinco características relacionadas com o tempo de carregamento e o tempo entre teclas, sendo estas características relacionadas com três períodos temporais. Um de menor fadiga de manhã, um de fadiga intermédio à tarde e um de maior fadiga à noite. As experiências envolveram três sujeitos que transcreviam trechos de livros através do teclado durante a recolha de dados. Recorrendo ao método de classificação SVM, obtiveram uma taxa de acerto de 98% para estas três classes, sendo que as previsões mal classificadas apresentavam-se como vizinhas das corretas.

O trabalho de (Natnithikarat *et al.*, 2019) apresenta uma tentativa de correlação de fadiga indicada com a estimada por um detetor de sonolência. Recorrendo a dinâmicas de uso de teclado e rato, em conjunto com observação de movimentos dos olhos para trabalhadores de escritório. Na experiência de recolha de dados participaram 15 homens e 3 mulheres entre os 12 e 35 anos. A tarefa utilizada neste trabalho consistia na simulação de monitorização de transações financeiras. Durante a tarefa, a intervalos de três a oito minutos os sujeitos eram requeridos a responder a um questionário relacionado com a escala KSS. Os dados recolhidos incluíam o tempo de carregamento de teclas, o número de vezes que o botão "Delete" era pressionado, velocidade e erro médio do rato. Um dos modelos de previsão recorria a um modelo de aprendizagem automática SVM, que apresentou um coeficiente de correlação de 0.70 em comparação com um modelo de análise de regressão linear com uma correlação de 0.55 com os valores reais. Não sendo mencionada qualquer medida de taxa de acerto.

O uso de telemóveis também permite a deteção de fadiga. O trabalho de (Al-Libawy *et al.*, 2017) reporta a um método de deteção baseado em métricas temporais de registo de texto através de teclados virtuais em *smartphones*. As métricas extraídas dos teclados para previsão são o tempo de carregamento e o tempo de dígrafos (tempo entre duas teclas consecutivas). Este trabalho em particular recorre a um *dataset* já existente com dados de 100 indivíduos, criado para a investigação de autenticação com base em dinâmicas de teclado. Os dados não estão anotados com níveis de fadiga, mas recorrendo a métricas de velocidade de escrita e de taxas de erro os dados são anotados pelo autor em duas classes, alerta e fatigado. São analisados dois modelos de aprendizagem automática, um que recorre a uma ANN com apenas uma camada escondida e treinada com uma divisão dos dados para treino de 70%, e outro modelo baseado em SVM. Os resultados da taxa de acerto (*accuracy*) para os modelos ANN e SVM são respetivamente 77 e 88%, apresentando o modelo SVM uma especificidade de 100% em comparação com 73% para o ANN.

2.3 Cronometria mental

A cronometria mental, o estudo de tempos de resposta ou reação em tarefas, pode ser usada para estudar a fadiga. Desde há vários anos que está demonstrado que com o acumular de fadiga os tempos de reação aumentam e a taxa de acerto em tarefas diminuem (Z. Guo *et al.*, 2016), (Kuratsune *et al.*, 2012), (Kato, Endo and Kizuka, 2009), existindo inúmeros estudos deste efeito no campo da psicologia e neurociência. Mais recentemente, (Guo *et al.*, 2018),

indica que além da fadiga afetar tempos de reação e taxas de acerto, ela diminui claramente a velocidade de resposta a decisões de inibição de resposta, introduzindo uma latência.

O estudo do tempo de reação simples, ou tempo mínimo de resposta a um estímulo, iniciou no século XIX por Francis Galton e continua nos dias de hoje. Segundo (Woods *et al.*, 2015), que estudou os possíveis fatores que influenciam o tempo de reação simples, a fadiga não é o único fator a ter um impacto. Outro dos fatores de influência é a idade do sujeito, sendo registado um aumento dos tempos de reação à taxa de aproximadamente 0.5 ms por ano de idade. Sendo que a experiência realizada sugere que este aumento é reflexo da desaceleração da capacidade motora com a idade. Os tempos de reação simples indicados por faixas etárias são 217.9 ms para os 18 a 24 anos, 221.0 ms para os 25 a 31 anos, 224.8 ms para os 32 a 38 anos, 227.7 ms para os 39 a 45 anos, 233.6 ms para os 46 a 51 anos e 239.1 ms para os 59 a 65 anos, com desvio padrão inferior a 28.1 ms. Segundo o autor não foram encontradas influencias significativas relacionadas com educação ou sexo. Curiosamente os estudo de (M. Guo *et al.*, 2016) que se debruça sobre o impacto da fadiga na capacidade de reação de condutores, analisando o tempo de reação durante uma escolha (não o tempo de reação simples), indica uma redução da capacidade de reação com a fadiga mais pronunciada para mulheres que homens. E reforça o impacto crescente da idade nos tempos de reação.

No seu estudo sobre tempos de reação (Woods *et al.*, 2015) tem em consideração vários fatores externos que que influenciam a medição do mesmo. Um destes, é que o contraste e brilho do estímulo visual influência a sua perceção pelo sujeito, podendo criar latências. Pelo que recorre a um estímulo brilhante e com alto contraste. Outro fator de preocupação são os atrasos introduzidos pelo hardware e software desde o momento que o sujeito pressiona o botão de resposta até que o sistema regista esta mesma resposta. No caso do seu hardware, a latência relacionada com o rato e equipamento visual foi medida como 17.8 ms. Relativamente ao software esta foi calculada como inferior a 1.05 ms, existindo principalmente uma preocupação com atrasos devido a interrupções de software. Na recolha de dados, existe um cuidado de apenas validar respostas entre os 110 a 1000 ms. Sendo tempos de reação abaixo de 110 ms humanamente impossíveis, e os superiores a 1 segundo claramente aberrantes.

De acordo com (Sur and Sinha, 2009) os potenciais relacionados com eventos ou Event-Related Potentials (ERP) são pequenas voltagens geradas nas estruturas cerebrais como resposta a eventos ou estímulos. Eles são um reflexo dos potenciais produzidos quando milhões de neurónios disparam em sincronismo durante o processamento de informação. Os ERP em humanos encontram-se divididos em duas categorias, ERP sensoriais e ERP cognitivos. A primeira relacionada com a perceção de um estímulo e a segunda com o processamento deste. Uma das ferramentas usadas no estudo destes mecanismos de resposta a eventos, são os testes Go/NoGo, que (Guo *et al.*, 2018) utiliza em conjunto com EEG e PERCLOS para estudar o impacto da fadiga nas reações de condutores.

2.3.1 Go/No Go

A tarefa ou teste Go/NoGo é uma tarefa utilizada em inúmeros estudos de neurologia e psicologia, para avaliar inibições a respostas. É composta por múltiplos ensaios ou iterações de duas tarefas de resposta a um estímulo, a Go e a NoGo. A tarefa de Go envolve um estímulo positivo, onde o sujeito deve efetuar uma ação como carregar num botão o mais rapidamente possível. Em oposição, a tarefa NoGo envolve um estímulo negativo onde o sujeito se deve abster de responder com uma ação. O tempo que o sujeito demora a responder ao estímulo e a quantidade de erros efetuados ao responder a um estímulo NoGo são registados. Em vários estudos que usam este género de testes os estímulos são de origem visual. Os parâmetros possíveis para os testes são o tipo de estímulo, o tipo de ação a efetuar, a quantidade de ensaios ou tarefas a efetuar pelo sujeito, a razão de tarefas Go - NoGo, o tipo de alternância entre tarefas Go e NoGo, o tempo de espera entre ensaios consecutivos no mesmo teste, e o intervalo temporal para o qual a resposta Go é válida. Respostas demasiado rápidas são consideradas humanamente impossíveis e as demasiado lentas como um erro ou falso alarme (Magnuson, Doesburg and McNeil, 2021) (Guo *et al.*, 2018).

Um dos estudos que claramente demonstra o uso de Go/NoGo no estudo da fadiga é o de (Guo *et al.*, 2018). Este aborda o impacto da fadiga mental na inibição de respostas, recorrendo ao Go/NoGo como tarefa de inibição e a análise por EEG dos sujeitos. O estudo envolve 30 homens e 30 mulheres dos 18 aos 27 anos, divididos por dois grupos, controlo e normal. O grupo “normal” tem a tarefa de conduzir durante 90 minutos, enquanto o grupo de controlo descansa a ver filmes durante 90 minutos. Antes e posteriormente a estas tarefas os sujeitos são avaliados com uma bateria de tarefas Go/NoGo. Nestes os estímulos são um círculo verde (GO) ou um círculo vermelho (NoGo), apresentados durante 200 ms com um intervalo entre estímulos de 1100 a 1700 ms. O número de estímulos é de 384 Go e 96 NoGo, apresentados aleatoriamente. A tarefa de condução, decorre em um simulador num cenário de autoestrada. É pedido aos sujeitos que sigam um carro que viaja entre 55 e 65 km/h, mantendo uma distância de segurança. Periodicamente o veículo da frente trava, num total de 180 vezes em intervalos de 25 a 35 segundos, tendo os sujeitos de carregar em uma tecla assim que vislumbrem a luz de travagem. Neste estudo as sessões experimentais decorrem da seguinte forma. Primeiro os sujeitos efetuam quatro blocos de ensaios Go/NoGo, cada um a demorar quatro minutos e com três minutos de descanso entre eles. De seguida é efetuado um questionário sobre fadiga. Depois ocorre a tarefa de 90 minutos a conduzir ou a ver filmes. Seguida por outro questionário de fadiga. Por fim é efetuada outra bateria de testes Go/NoGo. O questionário de fadiga utilizado foi a versão chinesa do *Profile of Mood States Short Form*. Durante a experiência, uma camara regista o diâmetro das pupilas do sujeito para analise PERCLOS e um capacete com 64 elétrodos grava dados EEG. Os resultados da análise mostram que o grupo de controlo manteve os tempos de reação e percentagem de falhas Go/NoGo antes e depois da tarefa, enquanto para o grupo com a tarefa de condução estes aumentaram. Uma representação dos tempos e tipo de sinal usados nesta experiência é visível na Figura 4, autoria deste mesmo autor.

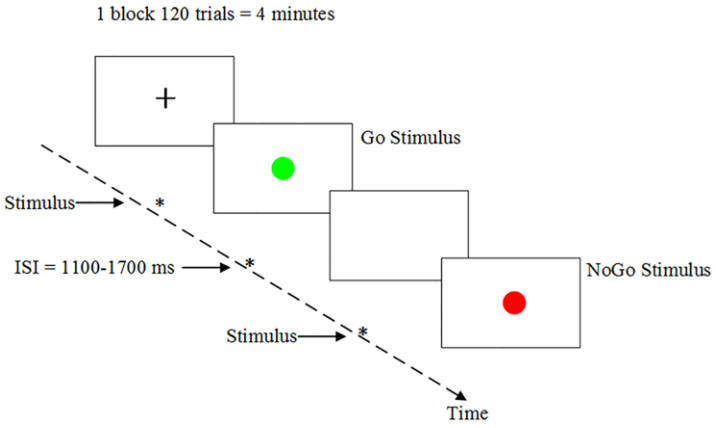

Figura 4 – Esquema representativo dos tempos em uma tarefa Go/NoGo (Guo *et al.*, 2018).

Um estudo mais recente que usa a tarefa Go/NoGo e EEG para avaliar as consequências da fadiga e a sua recuperação, é o trabalho de (Magnuson, Doesburg and McNeil, 2021). Neste estudo a tarefa Go/NoGo é efetuada num computador portátil, no ecrã do qual surgem os estímulos. Um quadrado colorido para o Go e um quadrado preto para o NoGo. Tendo os sujeitos de pressionar a tecla esquerda do *touchpad* como resposta ao estímulo Go. Dos 300 ensaios, 20% correspondiam a estímulos NoGo. O estudo reporta a participação de 20 indivíduos, dos quais 12 são mulheres, com idades entre 21 e 31 anos. O procedimento experimental inicia com uma bateria de quatro tipos de testes iniciais, dos quais o Go/NoGo faz parte. Seguido de 60 minutos de uma tarefa de memória relacionada com sequências de letras, utilizada para induzir fadiga. Após a qual, é repetida a bateria inicial de testes em três períodos. Que correspondem aos 0, 30 e 60 minutos após o fim da tarefa indutora de fadiga. Os resultados deste estudo referentes ao Go/NoGo são visíveis na Tabela 2 extraída do trabalho deste autor, onde este explica que não é observado um aumento dos tempos de reação e da taxa de erro contrariamente ao esperado. Sugerindo ter sido observado um mecanismo de compensação cerebral, possivelmente relacionado com os efeitos cognitivos da tarefa, do que uma redução da vigilância.

Tabela 2 – Resultados Go/NoGo de um conjunto de testes (Magnuson, Doesburg and McNeil, 2021)

	Precisão da resposta (%)	**Tempo de reação**
Pré fadiga	89.61 ± 5.79	318.31 ± 43.58
0 min depois	89.47 ± 6.41	326.18 ± 33.13
30 min depois	87.43 ± 7.69	311.69 ± 28.19
60 min depois	86.80 ± 7.01	317.98 ± 29.90

2.4 Aprendizagem automática

A aprendizagem automática ou *machine learning* é a deteção automática de padrões em dados, sendo as suas ferramentas criadas com o intuito de possuírem uma capacidade de aprendizagem (Osisanwo *et al.*, 2017) de forma a permitir a previsão de resultados com base nos dados disponíveis (Grandini, Bagli and Visani, 2020).

A aprendizagem automática pode ser efetuada de uma forma não supervisionada, com o intuito de obter relações na estrutura de dados sem existir um resultado mesurável. Ou de forma supervisionada com o propósito de rever ou classificar um determinado resultado recorrendo a dados estruturados (Jiang, Gradus and Rosellini, 2020). O trabalho de (Osisanwo *et al.*, 2017) analisa e compara vários algoritmos de classificação de aprendizagem supervisionada, como *Random Forest*, Naïve Bayes, SVM, redes neuronais, e árvores de decisão. Conclui que a classificação através de aprendizagem automática requere um ajuste cuidado dos parâmetros dos modelos e o uso de um conjunto de dados razoáveis. Afirma que o melhor algoritmo para um conjunto de dados específico, não garante igual precisão na sua aplicação em outro conjunto de dados logicamente diferente. Assim sugere o recurso a vários algoritmos em tarefas de previsão.

2.4.1 Descrição de alguns algoritmos usados na deteção de fadiga

Da análise da literatura existente sobre previsão de fadiga destacam-se vários algoritmos, que passamos a descrever.

O SVM foi introduzido por Vapnik como um modelo de aprendizagem automática com base em *kernel* para tarefas de regressão e classificação. A sua enorme capacidade de generalização, precisão e poder discriminatório atrai a comunidade de aprendizagem automática. O que o tornou num dos métodos mais usados de classificação, especialmente na resolução de problemas de classificação binários. Apesar das suas capacidades apresenta algumas desvantagens. A principal é a sua complexidade algorítmica que origina tempos de treino elevados em conjuntos de dados grandes. Devido ao SVM ter sido desenhado para classificação binaria, este apresenta uma performance menor no caso de classificação em várias classes. Assim como uma menor performance quando usado em conjunto com *datasets* em que a distribuição das suas classes se encontre desequilibrada (Cervantes *et al.*, 2020).

O KNN proposto por Hart em 1968, é um método estatístico não paramétrico de classificação e regressão. Este usa um modelo de vetores espaciais para classificar casos por categorias similares de uma forma simples e eficaz. O algoritmo KNN apresenta uma complexidade espacial e temporal elevada e uma dificuldade em lidar com *datasets* desequilibrados (Li, Chen and Song, 2020).

A ANN é um modelo de aprendizagem automática popular para classificação, agrupamento, reconhecimento de padrões e previsões. O seu grande potencial é a sua capacidade de paralelização que permite uma maior rapidez de processamento, usada no campo do

reconhecimento de imagem e processamento de linguagem natural. Nos dias de hoje as ANN são utilizadas universalmente na resolução dos mais variados problemas independentemente da complexidade. As ANN são um modelo que gere informação de uma forma semelhante ao sistema nervoso biológico de um cérebro, recorrendo a camadas de nós semelhantes a neurónios, que interagem umas com as outras. A rede neuronal de um ANN é composta por uma camada de entrada de informação, uma camada de saída que providencia o resultado e uma ou várias camadas escondidas que processam a informação entre estas duas camadas. As ANN apresentam várias vantagens como não requererem qualquer assunção sobre as propriedades e distribuição dos dados. Possuírem uma elevada flexibilidade e tolerância a falhas devido à sua elevada capacidade de lidar com dados incompletos e ruído. Apôs treino permitem efetuar previsões rapidamente (Abiodun *et al.*, 2018). Segundo (Jiang, Gradus and Rosellini, 2020) as redes neuronais requerem grandes amostras de dados de forma a obterem previsões precisas em certos tipos de dados, pelo que a sua performance pode ser pobre em pequenas amostras.

As árvores de decisão são um conjunto de algoritmos de classificação semelhantes a uma árvore. Da mesma forma que nestas existem raízes, ramos e folhas, nestes algoritmos existem estruturas semelhantes ao longo das quais se tomam decisões com base em caraterísticas. Desde o nó raiz até chegar aos nós folhas, são tomadas decisões num processo semelhante ao usado pelos humanos, percorrendo vários ramos até uma classificação ser atingida num nó folha. Existem vários algoritmos neste conjunto, que diferem na sua capacidade de tratarem apenas dados de categorias ou também dados numéricos (Patel and Prajapati, 2018). Segundo (Jiang, Gradus and Rosellini, 2020) estes métodos apresentam a vantagem de os modelos criados serem visualmente interpretáveis por humanos, mas apresentam um risco de um ajuste exagerado do modelo aos dados.

O algoritmo *Random Forest* é um algoritmo de classificação baseado em um método de particionamento recursivo que envolve a previsão com base em uma coleção de árvores de decisão individuais. Múltiplos modelos ou árvores são combinados numa única floresta aleatória, sendo cada resposta individual agregada em uma média da floresta de forma a obter uma previsão mais exata e estável. Uma das vantagens deste processo é a redução do risco de sobre ajustamento do modelo aos dados, mas que apresenta a desvantagem de que a floresta de centenas ou milhares de árvores é impossível de ser interpretada visualmente (Jiang, Gradus and Rosellini, 2020).

2.4.2 Métricas de avaliação

Segundo (Grandini, Bagli and Visani, 2020) existem várias métricas para avaliar a performance de um classificador, que para além de permitirem avaliar um modelo, permitem a comparação entre dois modelos ou entre um mesmo modelo com características diferentes. Muitas destas métricas são baseadas na matriz de confusão, que inclui toda a informação relevante sobre a performance de um modelo face a um conjunto de dados de teste. A matriz

de confusão é uma tabela que regista as ocorrências de duas situações, a classificação verdadeira ou real e a classificação prevista, conforme visível na Tabela 3.

Tabela 3 – Matriz de confusão

		Real	
Previsto	**classes**	**Positiva**	**Negativa**
	Positiva	TP	FP
	Negativa	FN	TN

Sendo os verdadeiros positivos (TP) as previsões classificadas como pertencente a uma classe que realmente pertencem a essa classe, e os falsos positivos (FP) as previsões classificadas como uma classe, mas que não pertencem a essa classe. De maneira similar os verdadeiros negativos (TN) são o número de previsões classificadas como não pertencentes a uma classe que realmente não pertencem a essa classe, e os falsos negativos (FN) as previsões classificadas como não pertencendo a uma classe, mas que realmente pertencem a essa classe.

Com a informação existente na matriz de confusão é possível obter informação sobre a *precision* do modelo que exprime a proporção de unidades no nosso modelo que este diz pertencerem a uma classe que realmente pertencem a essa classe, conforme visível na equação 2. A métrica *recall* do modelo visível na equação 3, permite medir a capacidade de efetuar previsões verdadeiras. Mas a métrica mais popular é a taxa de acerto ou *accuracy* visível na equação 4, que indica a probabilidade de o modelo efetuar uma previsão correta para uma unidade escolhida aleatoriamente.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$Taxa\ de\ acerto\ \ ou\ \ Accuracy = \frac{TP\ +\ TN}{TP\ + TN + FP + FN} \tag{4}$$

$$F1 - Score = 2\,.\frac{Precision\,.Recall}{Precision + Recall} \tag{5}$$

Outra métrica usada é o F1-Score que corresponde a uma média harmónica ponderada da *Precision* e *Recall*, conforme visível na equação 5, de maior interesse no caso de classes desequilibradas. O Cohens Kappa é uma métrica de avaliação que compara o modelo com um de classificação aleatória, podendo tomar valores de -1 a 1, onde 0 corresponde a um modelo completamente aleatório e o valor 1 ao modelo perfeito.

2.4.3 A metodologia CRISP-DM

O *cross-industry standard process for data mining*, mais conhecido por CRISP-DM (Chapman *et al.*, 2000), é uma metodologia ou modelo de processo para projetos de *data mining* . Este consiste em seis fases, a compreensão do negócio, a compreensão dos dados, a preparação dos dados, a modelação, a avaliação e a implantação. A fase de compreensão do negócio inclui a identificação do que realmente se pretende concretizar, definindo os objetivos do projeto. Nesta fase é também definida a métrica de avaliação do sucesso do projeto, e é criado um plano que inclui uma avaliação das ferramentas e técnicas a usar. A segunda fase referente à compreensão dos dados envolve a recolha, e posterior exploração de forma a permitir descrever e caracterizar os dados assim como as suas relações de uma forma estatística. Nesta fase é também avaliada a qualidade dos dados, se estes estão corretos e completos. A fase seguinte, de preparação de dados, envolve a seleção dos mais relevantes, a sua limpeza e uniformização, a construção de atributos derivados, e a agregação de parte dos mesmos por forma a criar novos atributos. A quarta fase é a de modelação, consiste na seleção do modelo ou modelos a usar, no desenho do mecanismo de teste do modelo, na construção do modelo e da sua avaliação. A fase seguinte é a de avaliação, não apenas da precisão do modelo, mas também de como este atinge os objetivos do negócio. O processo é revisto de forma a determinar os próximos passos a tomar, decidindo se o projeto avança para a última fase ou se é necessário iterar por fases anteriores. Na última fase, o modelo é implementado, sendo criados planos de manutenção e monitorização. Um relatório final é produzido, existindo uma revisão do que correu bem ou mal e do que é necessário melhorar.

Atualmente a análise de (Schröer, Kruse and Gómez, 2021) considera que CRISP-DM é o modelo de-facto standard em projetos de *data minning*. Segundo (Martinez-Plumed *et al.*, 2021) apesar de terem passado mais de duas décadas desde a origem do CRISP-DM, este método ainda representa uma das trajetórias mais comuns em ciência de dados, o caminho dos dados ao conhecimento, especialmente quando existe uma finalidade clara para o negócio. Este autor refere uma semelhança entre os projetos de ciência de dados e os de desenvolvimento de software, e sugere a aplicação de metodologias existentes nestes na flexibilização do desenvolvimento de projetos de ciência de dados.

2.5 Frameworks e Tecnologias

Uma *framework* é uma ferramenta que simplifica o desenvolvimento e manutenção de projetos complexos. Podem ser um software que contem módulos destinados a resolver tarefas comuns de programação. Ou apenas um conjunto de orientações, como no caso de diretivas legais. Em software são muitas vezes apoiadas por várias tecnologias, como as linguagens de programação que as constituem.

2.5.1 Frameworks de frontend

Em desenvolvimento Web o termo *frontend* aplica-se a interface gráfica de um site que o utilizador utiliza. Sendo as *frameworks* de *frontend* as plataformas ou ferramentas utilizadas na construção de websites. Das existentes, as que mais se destacam em 2022 são React, Angular, Vue.Js, e JQuery (Sakovich, 2021), (Technostacks, 2021).

React é uma biblioteca *open source* de JavaScript desenvolvida para criar interfaces de utilizador em 2011 pelo Facebook. É utilizada atualmente por grandes companhias como o PayPal, a Netflix e a Tesla. Nas suas vantagens incluem-se a facilidade de aprendizagem, uma grande capacidade de reutilização de código e da sua fácil leitura, um rápido tempo de desenvolvimento e facilidade de manutenção. Como desvantagens a *framework* possui uma documentação pobre, e uma dificuldade em aprender a sua complexa extensão sintática de JavaScript. O React é usado principalmente no rápido desenvolvimento de *single-page web applications* de interface com o utilizador (Sakovich, 2021), (Technostacks, 2021).

Angular é uma *framework* de *frontend* baseada em TypeScript, criada pela Google em 2010, que é atualmente utilizada pela Microsoft, BMW, Forbes e Gmail. Angular é muitas vezes comparada com React, sendo a principal diferença que a primeira permite *two-way binding*, ou seja, a reflexão de alterações efetuadas ao modelo na vista. Angular não se encontra limitado a aplicações *single-page*, permitindo múltiplas páginas. Como vantagens apresenta uma grande flexibilidade e capacidade de customização, uma elevada performance devido à uma pré-compilação, a existência de injeção de dependência, a sincronização em tempo real, e uma grande comunidade capaz de prestar ajuda. Como desvantagem é referenciada a sua complexidade que cria dificuldades de aprendizagem e uma menor performance no caso de aplicações com conteúdos dinâmicos. O seu uso é desaconselhado quando se pretende apenas construir pequenas aplicações, sendo mais destinado a aplicações complexas (Sakovich, 2021), (Technostacks, 2021), (Technostacks, 2018).

Vue.js é uma *framework* de *frontend* baseada em TypeScript, originalmente lançada em JavaScript em 2015. Permite facilmente o desenvolvimento de pequenas e grandes aplicações dinâmicas, mas não é popular em grandes companhias. De entre os utilizadores destacam-se a Alibaba e a Xiaomi. Das suas vantagens destaca-se a facilidade de aprendizagem, apresentar uma elevada performance e escalabilidade, e a existência de documentação detalhada. Como desvantagens apresenta, uma comunidade reduzida de utilizadores capazes de providenciarem apoio, o facto de muita da sua documentação e módulos estarem escritos em chinês, e não possuir tantas capacidades como React ou Angular (Sakovich, 2021), (Technostacks, 2021).

JQuery é uma *framework* mais antiga, desenvolvida em 2006 como uma biblioteca *open source* de JavaScript. É utilizada na criação de aplicações desktop de e para melhorar a interatividade das interfaces de utilizador. Companhias como o LinkedIn, Twitter e Slack usam JQuery nos seus produtos. Como vantagens apresenta a sua simplicidade que se traduz numa fácil aprendizagem, uma independência do tipo de navegador de internet, e existir uma larga

comunidade disponível a prestar ajuda. Como desvantagens apresenta uma performance mais lenta, o facto de se encontrar desatualizada devido a sua idade, e de ser difícil criar aplicações maiores e dinâmicas (Sakovich, 2021), (Technostacks, 2021).

2.5.2 Frameworks de aprendizagem automática

As *frameworks* e as tecnologias existentes para desenvolver projetos de aprendizagem automática dividem-se em duas escolhas, as associadas com a linguagem R e as associadas com a linguagem Python. O R é uma linguagem e ao mesmo tempo um ambiente de computação estatística e de criação de gráficos, desenvolvida nos laboratórios Bell, antiga AT&T. O R providencia uma grande variedade de ferramentas estatísticas de modelação, teste, classificação e *clustering*, além de ferramentas de visualização gráfica de alta qualidade em código aberto multiplataforma. Este também suporta uma variedade de formas de apresentar, calcular e manipular dados recorrendo a vetores, matrizes, gráficos e bibliotecas externas. Por ser uma verdadeira linguagem de programação, o R permite também a criação de funções, o uso de extensões modulares via pacotes de instalação, e a ligação a outras linguagens como C para resolver tarefas que exigem maior performance computacional (The R Foundation, 2021). O Python é uma linguagem genérica desenvolvida em 1989, orientada a objetos e que enfatiza a sua fácil leitura. Conjuntamente com Java e C, é uma das linguagens mais populares do mundo. Possui várias bibliotecas em código aberto dedicadas a *data science* como Numpy para trabalhar com vetores complexos, Pandas para a análise e manipulação de dados e Matplotlib para a visualização de dados (IBM, 2021). O Python é usado por várias companhias como o Google, o YouTube, e o Instagram, e apresenta várias vantagens sobre a linguagem R. O facto de ser uma linguagem genérica permite o seu uso em outros domínios que os da estatística, permitindo o design de websites e uma melhor integração com outros módulos de software desenvolvidos em outras linguagens. A sua fácil leitura permite ganhos de produtividade e uma mais fácil aprendizagem que o R. Em contrapartida o R é mais adequado à análise e visualização, permitindo a rápida criação de protótipos de aprendizagem automática, e uma fácil análise exploratória (Kumar, 2019). Ambas as linguagens são suportadas por uma grande comunidade que desenvolve bibliotecas de código aberto na área da *data science*. Sendo que o R é mais adequado à aprendizagem estatística com a sua vasta biblioteca de módulos de experimentação e análise exploratória de dados, enquanto que o Python é a melhor escolha em aprendizagem automática e aplicações de larga escala que envolvam a análise de dados de aplicação web (IBM, 2021).

Existem várias bibliotecas ou *frameworks* (consoante o autor) destinadas a área de aprendizagem automática disponíveis quer em Pyton ou em R, das quais destacamos a Scikit-Learn, a TenserFlow e a Keras.

Scikit-Learn é fácil de usar e implementa eficientemente vários algoritmos de aprendizagem automática. Criada em 2007 por David Cournapeau é agora dirigida por uma equipa de investigadores do Instituto Francês de Pesquisa em Ciência de Computadores e Automação (Géron, 2019). Segundo (Phaladisailoed and Numnonda, 2018) é uma biblioteca de código

aberto, com algoritmos de classificação, regressão e *clustering*. Permite também a normalização, uniformização e limpeza de dados.

TensorFlow é uma biblioteca complexa de computação numérica distribuída que permite treinar e correr grandes redes neurais eficientemente, distribuindo a computação por várias unidades de processamento gráficas. O TenserFlow foi criado na Google e suporta muitas das suas aplicações nesta área. Desde 2015 que é uma biblioteca de código aberto, e em 2019 foi lançada a sua segunda versão (Géron, 2019). Segundo (Phaladisailoed and Numnonda, 2018) esta *framework* pode ser aplicada também nas áreas de reconhecimento de voz, visão computacional e robóticas entre outras.

Keras é uma API de d*eep learning* de alto nível que simplifica o treino e execução de redes neurais. Pode correr em cima de outras bibliotecas, sendo que no caso do TensorFlow este inclui a sua própria implementação de Keras (Géron, 2019). Segundo (Phaladisailoed and Numnonda, 2018) esta biblioteca é de código aberto.

O trabalho de (Stancin and Jovic, 2019) classifica scikit-learn como a melhor biblioteca no campo de aprendizagem automática, possuindo uma documentação bastante rica e um vasto conjunto de algoritmos. No caso de *deep learning*, recomenda o uso de Keras para protótipos e TensorFlow em projetos com grande customização.

2.5.3 Frameworks legais

Uma das *frameworks* legais, que dizem respeito ao projeto é o Regulamento Geral sobre a Proteção de Dados (RGPD) (Assembleia da República, 2019), que indica as regras relativas ao tratamento, armazenamento e circulação de dados pessoais. O RGPD possui vários artigos que são especialmente aplicáveis a este projeto, nomeadamente os artigos 21 em 31 que aceitam o prejuízo dos direitos consagrados no RGPD, relativos ao acesso, retificação, limitação do tratamento e oposição, na medida do necessário para os fins de investigação científica. Mas afirmam que o seu tratamento deve respeitar o princípio da minimização dos dados através da sua anonimização ou pseudonimização.

2.5.4 Keycloak

O Keycloak é um software de código aberto de gestão de utilizadores, que permite o recurso a *single sing-on* e efetuar uma gestão de identidades e acessos. Possui uma interface gráfica de fácil configuração que permite a criação anónima de utilizadores, e a sua gestão pelos mesmos. Suporta a sua integração com outros sistemas de gestão de utilizadores, como por exemplo o Active Directory da Microsoft, ou mesmo redes sociais como o Facebook ou Google. O Keycloak permite também autenticar utilizadores através do protocolo OpenId Connect ou do Security Assertion Markup Language 2.0. O seu interface gráfico permite a administradores gerirem centralmente os vários aspetos do servidor Keycloak, ativando ou desativando opções complexas com o simples carregar de um botão. A sua vasta gama de interligações inclui a

possibilidade de autenticação através de contas sistemas externos como o Facebook ou o Google. Este software é financiado pela Red Hat, responsável por vários outros *software* de código aberto (Keycloak, 2022).

2.6 Breves conclusões

Da literatura revista é notório o foco da Comissão Europeia no ser humano, no âmbito da Indústria 5.0, que propõe o uso das novas tecnologias como a inteligência artificial para garantir o bem-estar físico e mental dos trabalhadores. Que a fadiga é pervasiva na sociedade moderna, quer por falta de sono ou excesso de trabalho, originando acidentes, problemas de saúde e diminuição de performance. Existe uma variedade de investigação na área de deteção, com vários métodos utilizados, como a análise visual, de sinais fisiológicos e de movimentos e gestos. A análise dos vários artigos de deteção e investigação de fadiga permite identificar o que é necessário para a construção de um detetor, sendo notório a necessidade de uma escala e a prevalência da inteligência artificial com modelos de aprendizagem automática. Das escalas destacam-se três, a USAFSAM, a KSS e a VAS-F, com a ressalva que todas estas são subjetivas. Dos modelos destacam-se os algoritmos de classificação SVM, KNN e ANN, com taxas de acerto (*accuracy*) de cerca de 90%, não sendo normalmente indicado de forma explicita o número de classes correspondentes a esta elevada precisão. No estudo dos efeitos da fadiga é notório o uso da cronometria mental, devido ao impacto desta nos tempos de reação e na inibição de respostas. A fadiga origina um aumento dos tempos de reação e da taxa de erro em tarefas. Dos fatores com impacto no tempo de reação destacam-se a idade e possivelmente o sexo biológico. Uma das ferramentas prevalentes nos estudos de neurologia e psicologia relacionados com fadiga é a tarefa Go/NoGo, que avalia tempos de reação incluindo a inibição de resposta. Apesar da correlação da fadiga nas métricas desta tarefa de teste, não foi identificado qualquer detetor baseado na mesma. Em várias experiências de deteção ou investigação de fadiga, é notório um padrão de atuação. Iniciado por uma autoavaliação da fadiga e recolha de sinais, seguido por uma tarefa indutora de fadiga, e terminando por outra autoavaliação da fadiga e mais recolha de sinais. Note-se que a existência de uma tarefa indutora de fadiga não é garantia absoluta da sua indução. Na aplicação do teste Go/NoGo, existe uma necessidade de reduzir a latência introduzida pelo *software* e *hardware* de captura de dados e de escolher estímulos claros e inequívocos.

3 Análise de Valor

Este capítulo aborda a análise de valor do projeto, aplicando o método de *Value Analysis* ao mesmo. Com o modelo *New Concept Development* é analisada e identificada a oportunidade a alcançar com este projeto. Recorrendo a um diagrama *Function Analysis System Technique* (FAST) é efetuada uma análise funcional das funções no projeto, e através de um método de análise hierárquica ou *Analytic Hierachy Process* (AHP) é selecionada qual a escala de fadiga que mais valor gera. São também avaliados outros valores do projeto, nomeadamente a nível do valor para o cliente e do valor por este percebido, enumerando benefícios e sacrifícios. Finalmente a análise culmina com a construção de uma proposta de valor recorrendo ao modelo de *value proposition* de Osterwalder.

3.1 Contexto da Análise de Valor

Existem vários modelos para efetuar uma análise de valor a um projeto, como o modelo *Fuzy Front End* e o *Value Analysis*. Neste trabalho escolhemos aplicar o modelo *Value Analysis* baseado no trabalho de (Rich and Holweg, 2000). Segundo estes, a *value analysis* é um processo que visa aumentar os lucros da aplicação de produtos, recorrendo a variadas técnicas, muitas das quais comuns. A sua abordagem é universal, podendo ser tanto aplicada a produtos como a serviços, por fábricas ou empresas de serviços. Encontrando-se no cerne deste processo a identificação de funcionalidades ou particularidades que não trazem valor acrescentado, mas apenas custos adicionais. No caso de novos productos ou funcionalidades o processo refere-se como *Value Engineering*.

Segundo o trabalho, o processo de análise é composto por cinco fases, a Orientação, a Identificação e Análise Funcional, a Criação de Alternativas, a Análise e Avaliação, e a Implementação, conforme visível na Figura 5.

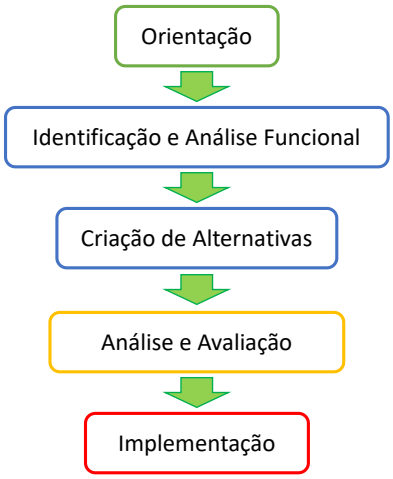

Figura 5 – Cinco fases da *Value Analysis*

A fase de orientação compreende a formação da equipa de especialistas que efetua a análise de valor, podendo incluir clientes e mesmo fornecedores. É nesta fase que é escolhido o produto ou família de produtos a estudar.

A fase seguinte é a identificação e análise funcional, onde se inicia a análise, por identificar sistematicamente as funções mais importantes do produto ou serviço que contribuem para a sua comercialização. Primeiro as funções são descritas, sendo de seguida classificadas por pares pela sua importância segundo o critério menor, média e maior.

A terceira fase é a de criação de alternativas recorrendo a brainstorming, que visa o desenvolvimento de alternativas criativas com melhor custo-benefício para atingir as funções básicas. Vários truques relacionados com *brainstorming* podem ser usados, como a escrita de sugestões anónimas em cartões, desenhar diagramas, ou a técnica do patinho de borracha de software onde a funcionalidade tem de ser explicada a um cliente virtual que nada conhece do produto.

A quarta fase foca-se na análise e avaliação do custo e valor de cada função, permitindo avaliar a proporção entre os dois. O valor é determinado estimando o custo mais baixo para a produção de cada função básica no seu mínimo, sendo o potencial de valor calculado pela diferença entre o custo e o valor obtido. Nesta fase, se for verificado que alguma das partes não tem propósito real ou valor para o cliente, mas apenas um custo, então essa parte é eliminada. Sendo também possível modificar ou substituir partes de forma a baixar o custo do produto e melhorar o produto.

A última fase é a implementação, onde a equipa composta na fase inicial reporta os resultados obtidos à equipa de decisores, confiantes que o produto ou serviço pode ser implementado visto gerar lucros para o negócio e valor para o cliente. Sendo este aprovado e inicializada a sua produção.

3.2 A Oportunidade

Este projeto envolve uma oportunidade, pelo que vamos proceder à sua identificação e análise recorrendo ao modelo *New Concept Development* originário do trabalho de (Koen *et al.*, 2001), sendo visível na Figura 6 um diagrama deste modelo extraído do mesmo autor.

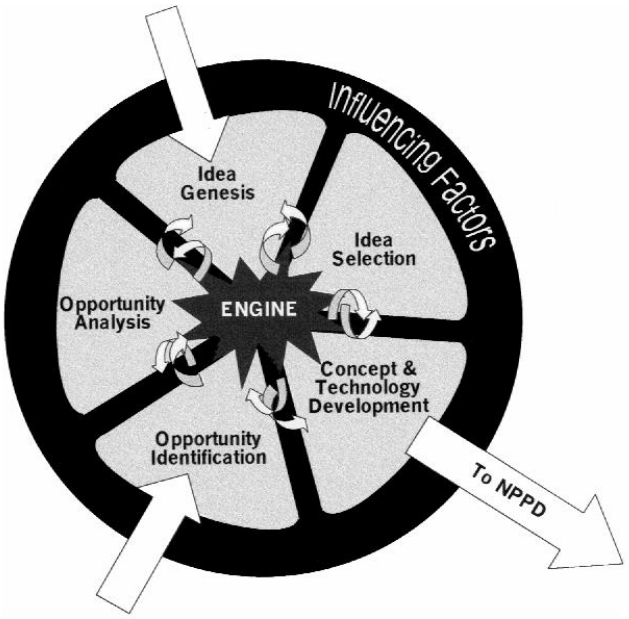

Figura 6 – Diagrama do modelo do *New Concept Development* (Koen *et al.*, 2001)

Este conceito envolve três partes principais, o motor de inovação, os fatores influenciadores correspondentes ao ambiente onde se insere, e os cinco elementos centrais que os unem. Visto que neste projeto não estamos a usar o Fuzy Front End mas sim o Value Analisys, vamos aproveitar o conceito para apenas apoiar a identificação e análise da oportunidade.

Existe uma necessidade de detetores de fadiga, especialmente os capazes de a avaliar em trabalhadores, ajudando a prevenir acidentes e custos económicos. Existem vários modelos de deteção presentes na literatura para a sua identificação, alguns baseiam-se em métodos que são intrusivos como os que recorrem a sinais fisiológicos, muitos necessitam de equipamentos dedicados com considerável processamento, como os sistemas de monitorização de condutores que usam câmaras fotográficas especiais. Outros são tecnologias bastante inovadoras capazes de identificar fadiga apenas pela escrita de texto em *smartphones*. Assim existe uma oportunidade de criar um novo tipo de detetor facilmente acessível a todos, de uma forma pouco intrusiva e expedita, recorrendo a um método inovador na área de deteção de fadiga, os testes Go/NoGo disponíveis online. Tendo em conta esta oportunidade e comparando-a com os outros tipos de deteção é possível realizarmos uma análise da força desta oportunidade, da qual resultou a Tabela 4. Apesar de existir informação sobre algumas das características dos métodos na literatura, está não é quantitativa tendo que ser interpretada o que origina alguma subjetividade. Assim optou-se por representar as características por níveis de 1 a 5 representados por bolas negras.

Tabela 4 – Análise comparativa da oportunidade

Método	Continuo	Velocidade	Acessibilidade	Não Intrusivo	Inovador
Go/Nogo	Não	●●●●○	●●●●●	●●●●○	●●●●●
Visual	Sim	●●●●●	●●●○○	●●●●●	●●●○○
Fisiológicos	Sim	●●●●●	●○○○○	●○○○○	●○○○○
Texto	Semi	●●●○○	●●●●●	●●●●○	●●●●○
Texto e Postura	Semi	●●●○○	●●●○○	●○○○○	●●●●○

O método de deteção por Go/NoGo não é possível de ser efetuado em tempo real, ao contrário das análises feitas por sinais fisiológicos ou visuais. Note-se que a deteção por texto e postura apenas são semi contínuos, porque apesar de a análise poder ser feita em continuo, os resultados são referentes a intervalos de tempo. Relativamente à velocidade um teste Go/NoGo demora menos de um minuto, mas as análises visuais e fisiológicas são instantâneas. A medição por texto necessita sempre da introdução de um excerto de texto que demora um tempo superior a um teste Go/NoGo. Em termos de acessibilidade, a análise de texto pode ser feita em um telemóvel, um item comum a todos nós, sendo que o detetor Go/Nogo necessita apenas de acesso à internet, algo extremamente comum nos dias de hoje através de um computador. Os métodos visuais, por postura e fisiológicos, necessitam de equipamentos especiais cuja disponibilidade é mais restrita. Em termos de intrusão os métodos baseados na postura e em sinais fisiológicos necessitam de equipamentos que podem inibir o movimento e perturbar o trabalho de um trabalhador, enquanto uma análise de escrita ou um teste Go/NoGo pode ser facilmente efetuada por um trabalhador de escritório. Nota-se que os métodos visuais são os menos intrusivos, bastando o sujeito apenas se encontrar em frente a uma câmara. Relativamente à inovação, o uso de análise de escrita ou postura são os métodos menos abordados na literatura, com exceção do uso de testes Go/NoGo.

Apesar das suas vantagens o uso de Go/NoGo não pode ser efetuado em contínuo, pelo que necessita de uma vontade expressa do sujeito para efetuar o teste. A sua principal competição provém dos métodos visuais, que apesar de serem mais rápidos e menos intrusivos, são também menos acessíveis e inovadores.

3.3 FAST

Uma das ferramentas de análise funcional usadas em análise de valor é o diagrama FAST, que permite listar as funções e subfunções do projeto. Neste diagrama, à esquerda aparecem as funções de mais alto nível ou *outputs*, e à direita as de menor nível relacionadas com *inputs*. Existem três eixos, da direita para a esquerda o WHY que indica por ordem crescente o porquê de as funções existirem, da esquerda para a direita o HOW que indica como as funções são conseguidas. O terceiro eixo, WHEN, é diferente. Aplicável apenas quando uma ligação vertical se afasta de um bloco, é usado para indicar uma função que acontece depois.

No nosso diagrama destaca-se outra convenção, a lógica E representada por uma linha horizontal que se divide para ligar a outros blocos, e a lógica OU representada por várias linhas que saem horizontalmente do mesmo bloco e ligam a blocos diferentes. O diagrama resultante desta análise é visível na Figura 7.

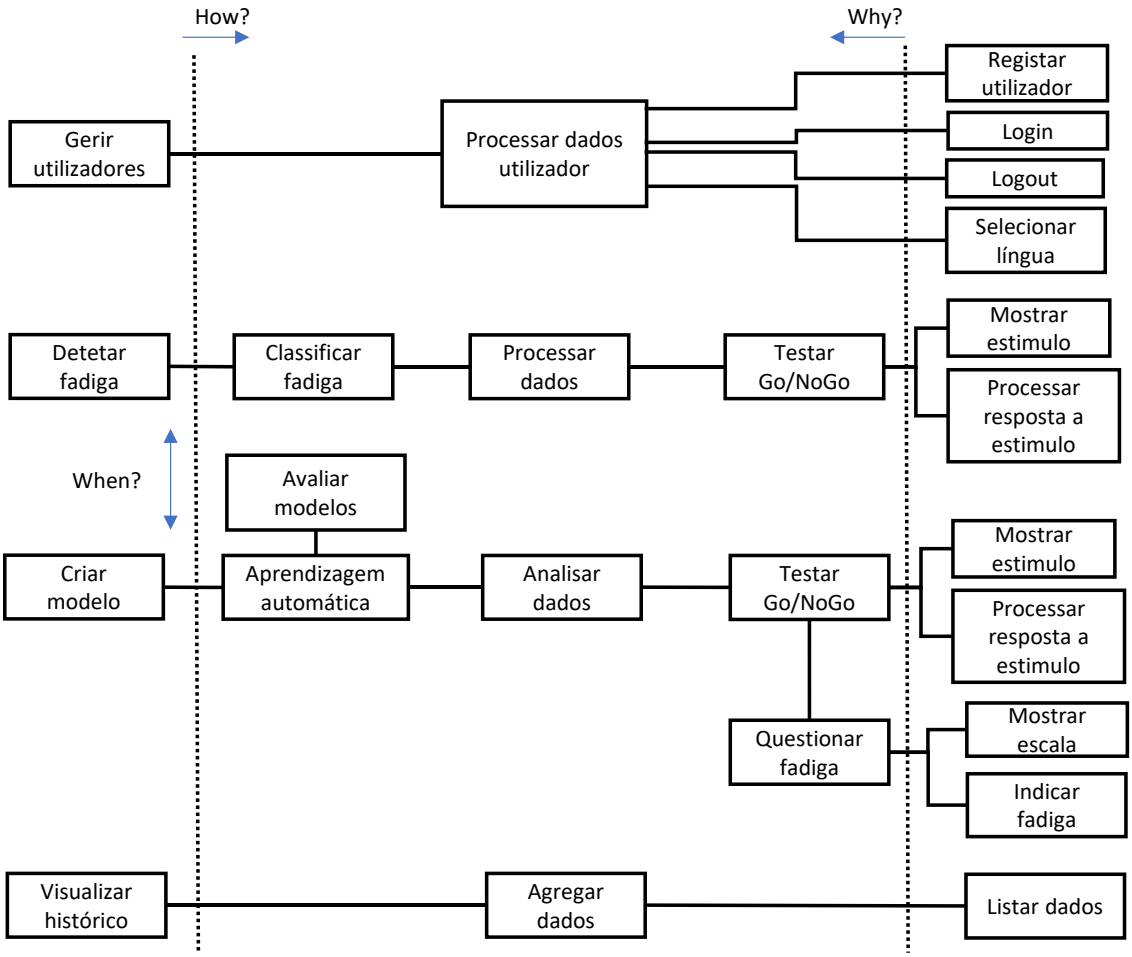

Figura 7—Diagrama FAST

Neste existem quatro caminhos principais. O primeiro referente à função de gerir utilizadores, composto por um processamento dos dados de utilizador relacionados com o registo, ou o *login*, ou o *logout* ou a opção de língua dos mesmos. O caminho de detetar fadiga é suportado pelo mostrar dos estímulos e processamento das suas respostas, o que permite fazer um teste Go/NoGo. Este teste depois é processado, e de seguida classificado em um classificador de forma a detetar a fadiga. O caminho de criar o modelo é ligeiramente diferente, visto envolver funções de software e de análise humana. O software efetua o teste Go/NoGo, depois questiona a fadiga, mostrando a escala e recebendo a escolha do utilizador. Estes dados são agregados e analisados por um humano, o que permite efetuar a aprendizagem automática por *software* recorrendo a vários modelos. Estes modelos são posteriormente avaliados por um humano, sendo selecionando um único, finalizando a criação de um modelo de classificação e deteção de fadiga. O último caminho é o de visualização do histórico de fadiga,

pelo que existe um pedido para listar os dados, que obriga ao agregar dos mesmos de forma a se poder visualizar o histórico.

3.4 AHP

Um dos problemas que o projeto enfrenta é decidir qual a escala de fadiga a usar no detetor, para a construção do *dataset* que relaciona os resultados dos testes com a fadiga autoavaliada pelos sujeitos. Existem vários critérios que podemos usar para avaliar estas escalas. Um destes é velocidade de preenchimento da escala, que é influenciado pela sua estrutura, facilidade de interpretação e preenchimento da mesma. Outro critério é a complexidade de tratar a escala preenchida, sendo ideal que possa ser resumida em um único valor. Outro critério bastante importante é a adequação da escala a medição da fadiga que pretendemos. Estes critérios e suas importâncias podem ser consideradas subjetivas e certamente qualitativas, assim vamos aplicar um método multicritério para este caso, o AHP de (Saaty, 2004). As três escalas a considerar são a escala de fadiga mental USAFSAM, a KSS e a VAS-F. Na Figura 8 é possível observar uma representação desta divisão hierárquica, que indica o problema, os critérios e as alternativas a escolher.

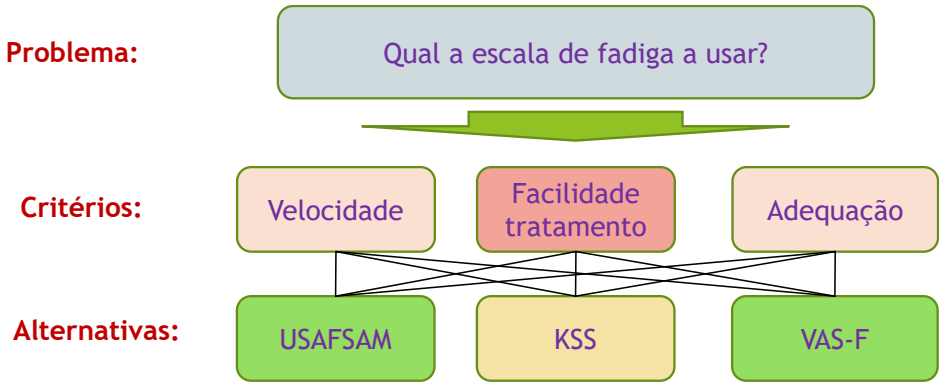

Figura 8 – AHP divisão hierárquica

A Tabela 5 representa a classificação comparativa dos critérios segundo o nível de intensidade de importância sugerido pelo autor do método, 1 de igual importância, 2 de fraca ou ligeira importância superior, e 3 de moderada importância superior.

Tabela 5 – AHP Comparação entre critérios

Critérios	**Velocidade**	**Facilidade**	**Adequação**
Velocidade	1	3	1/2
Facilidade	1/3	1	1/3
Adequação	2	3	1

Note-se que foi dada prioridade à velocidade em relação à facilidade de tratamento dos resultados, fruto do objetivo temporal a cumprir, e uma maior prioridade à adequação da escala em relação aos outros critérios.

Tabela 6 – AHP critério velocidade

Velocidade	**USAFSAM**	**KSS**	**VAS-F**
USAFSAM	1	1	3
KSS	1	1	3
VAS-F	1/3	1/3	1

A Tabela 6 representa a comparação do critério velocidade entre as três escalas, ressalvando-se que de facto fazer uma escolha única é moderadamente mais rápido que decidir entre 18 critérios.

Tabela 7 – AHP critério facilidade

Facilidade	**USAFSAM**	**KSS**	**VAS-F**
USAFSAM	1	1	3
KSS	1	1	3
VAS-F	1/3	1/3	1

A Tabela 7 representa a comparação do critério de facilidade de processamento, tendo em conta que é moderadamente mais complicado processar e armazenar um conjunto de 18 valores, que um único valor.

Tabela 8 – AHP - critério adequação

Adequação	**USAFSAM**	**KSS**	**VAS-F**
USAFSAM	1	3	1/2
KSS	1/3	1	1/3
VAS-F	2	3	1

A Tabela 8 representa a comparação da adequação das escalas à medição da fadiga. Tendo em conta que todas as escalas medem a fadiga, importa indicar que a KSS mede a sonolência, enquanto a VAS-F abrange um conjunto de 18 critérios de fadiga. No meio apresenta-se a USAFSAM que mede a fadiga mental.

O resultado da aplicação do método AHP encontra-se visível na equação abaixo,

$$\begin{bmatrix} 0.428 & 0.428 & 0.333 \\ 0.428 & 0.428 & 0.141 \\ 0.142 & 0.142 & 0.524 \end{bmatrix} X \begin{bmatrix} 0.333 \\ 0.141 \\ 0.524 \end{bmatrix} = \begin{bmatrix} 0.378 \\ 0.277 \\ 0.343 \end{bmatrix} \quad (6)$$

que indica a escala USAFSAM como a favorita, com um valor de 0.378. A escala VAS-F encontra-se próxima com o valor 0.343, reflexo da sua maior complexidade que a penaliza na

sua competição com a escala vencedora. A KSS apresenta-se como última escolha com o valor 0.277. Uma análise da razão de consistência revela que o maior valor obtido é 0.046, inferior a 0.1 o que nos leva a concluir que os valores das prioridades relativas são consistentes. Recordamos que este método é uma tentativa de reduzir a subjetividade desta escolha, pelo que com valores tão próximos para a USAFSAM e VAS-F, seria provável que outros autores, com interpretações ligeiramente diferentes, chegassem a uma conclusão inversa, o que não seria menos válido.

3.5 Valor

Na criação e futuro uso do sistema de deteção de fadiga podemos dividir os valores criados em dois tipos. O valor real que o sistema traz para o cliente e o valor que este se apercebe como um benefício para si. Tendo em atenção que o benefício trazido por um valor tem por norma associado um sacrifício para a sua incorporação. Note-se que neste projeto podemos abordar a questão do cliente, através de uma simplificação, indicando dois tipos de clientes. O cliente institucional que se preocupa com a investigação científica do projeto e o usuário do sistema interessado em examinar a sua fadiga, sendo que vários dos seus interesses são comuns.

Assim além do valor real do sistema, a capacidade de detetar fadiga, existem outros valores que são percebidos pela sua importância e valorizados pelo cliente, como a acessibilidade, a velocidade de utilização e usabilidade do sistema. Além da qualidade da deteção e a qualidade ou beleza estética do sistema.

Tabela 9 – Valor, benefícios e sacrifícios

Benefício	**Sacrifício**
Qualidade de deteção	Tempo de desenvolvimento Velocidade de deteção
Acessibilidade	Custos de servidor
Velocidade de utilização	Menor qualidade na deteção
Usabilidade	Tempo de desenvolvimento
Beleza estética	Tempo de desenvolvimento
Suporte multilingue	Tempo de desenvolvimento

O benefício de uma elevada qualidade de deteção é obtido à custa de um maior tempo gasto no desenvolvimento e aperfeiçoamento do modelo. Sendo necessário testar mais modelos e obter mais dados de inquéritos de fadiga. Os ganhos da acessibilidade da aplicação através da sua disponibilidade online, implicam um custo relacionado com a manutenção de um servidor disponível ao publico. Uma maior velocidade de utilização do sistema, implica um questionário de fadiga mais simples e um menor tempo de disponibilidade para a execução do

modelo de previsão. Melhorias a nível de usabilidade e facilidade de utilização implicam principalmente um maior tempo gasto no desenvolvimento do software e o uso de uma escala mais simples que pode ter impacto na qualidade de deteção. Duas das preocupações levantadas aquando das reuniões de *brainstorming* do projeto, são uma preocupação com o aspeto estético do detetor e a possibilidade de suporte multilingue de português e inglês. Assim é percetível que estas funcionalidades secundárias, são valorizadas a nível institucional e do usuário, implicando como contrapartida um maior tempo de desenvolvimento. Um pequeno resumo destes benefícios e sacrifícios encontram-se visíveis na Tabela 9.

3.6 Proposta de valor

Tendo em conta que identificamos o problema e o seu contexto no capítulo inicial deste trabalho, conhecemos os clientes institucionais e usuários do sistema, identificamos e analisámos a oportunidade relativamente às alternativas existentes, e analisámos os valores e seus correspondentes benefícios e sacrifícios, podemos iniciar a construção de uma proposta de valor. Existem vários tipos de propostas de valor, alguns mais simples como o *elevator pitch* e outros mais complexos, pelo que vamos utilizar o modelo de Osterwalder (Osterwalder *et al.*, 2014). Visto que este modelo foi desenvolvido para modelar negócios e não investigação de software, poderão existir algumas pequenas incoerências no mesmo.

A Figura 9 apresenta o modelo de proposta de valor, com a seção à direita representando o perfil do cliente e a da esquerda os valores da solução. Do lado do perfil do cliente, identificamos os trabalhos que estes pretendem, a criação de um detetor de fadiga com base em tarefas Go/NoGo da parte do cliente institucional, e a simples deteção de fadiga pelos usuários. Como resultados ou benefícios, estes pretendem a investigação do uso de Go/NoGo na deteção de fadiga, a avaliação de fadiga, e a possibilidade de selecionar a língua inglesa ou portuguesa. As dores ou irritações que preocupam os nossos clientes são o tempo de deteção, a intrusão do detetor ou complicações no seu uso, a possibilidade de erro na deteção de fadiga, e de o detetor apresentar um aspeto estético feio.

Relativamente aos valores à esquerda, nos produtos e serviços disponibilizados pela solução são visíveis a deteção de fadiga e a gestão de utilizadores, essenciais à criação e uso do mesmo. Os criadores de ganhos descrevem como estes produtos e serviços trazem ganhos ao cliente, notando-se a existência desta tese e da sua disseminação como um avanço investigacional nesta área. A existência do detetor capaz de classificar a fadiga em vários níveis. A disponibilidade de um histórico de resultados, de forma ao utilizador poder avaliar as alterações. E a existência de uma opção para mudar a língua de português para inglês. A última seção é a dos aliviadores de dores de clientes, onde se destaca o uso de um questionário de fadiga simples. A disponibilização do sistema online, de forma a ser facilmente acessível, sem necessidade de equipamentos especiais intrusivos. O uso de inteligência artificial de forma a aumentar a precisão da deteção. O uso de métodos de validação e avaliação de forma a garantir a qualidade dos resultados. E por fim o uso de uma

framework de software especificamente dedicada ao design de *frontends*, com o intuito de melhorar a estética do mesmo.

Figura 9 – Value Proposition Canvas

4 Análise de Requisitos e Design Arquitetural

Neste capítulo abordamos um tema critico ao sucesso do projeto, a análise de requisitos e o design da solução. Iniciamos a análise com um modelo do domínio do problema, seguido do método "*Functionality Usability Reliability Performance Supportability +*" (FURPS+), que nos permite estruturar os diferentes requisitos por categoria. De seguida, tendo em conta que a solução passa por um sistema de aplicações e componentes interligados, existe um foco no design arquitetural do sistema recorrendo ao modelo arquitetural de 4+1 vistas de (Kruchten, 1995). Este modelo é composto por cinco vistas que abordamos individualmente, a vista de cenários, a vista lógica, a vista de implementação, a vista de processo e a vista de implantação. Tendo em conta que a *Unified Modeling Language* (UML) foi adotada posteriormente ao desenvolvimento deste modelo, vamos utilizar preferencialmente os seus diagramas na descrição de cada vista.

A primeira vista abordada é a vista de cenários, mais direcionada para o utilizador, onde é visível um diagrama de casos de uso e uma descrição dos mesmos. A segunda vista é a vista lógica, onde através de um diagrama de classes explicamos a lógica de modelação destas entidades. A terceira vista é a vista de implementação, direcionada aos programadores, onde através de diagramas de componentes são apresentadas duas alternativas de arquitetura possíveis. Na quarta vista, a de processos, é mostrado o fluxo no sistema recorrendo a um diagrama de sequência do sistema, e um diagrama *Business Process Model and Notation* (BPMN) do processo de aprendizagem automática. A última vista é a de implantação, mais direcionada aos engenheiros de sistemas, que mostra através de diagramas de implantação, duas opções de mapeamento do software em máquinas físicas.

No final do capítulo abordamos a escolha das *frameworks* de *frontend* a utilizar, assim como a *framework* a usar na análise exploratória de dados e aprendizagem automática.

4.1 Modelo do domínio do problema

Uma das ferramentas que permite organizar o conhecimento do problema de uma forma estruturada é o diagrama de modelo de domínio, visível na Figura 10. Apesar de não pretendemos efetuar *Domain Driven Design*, esta ferramenta mostra de uma forma clara e percetível, os principais conceitos com que trabalhamos e as suas inter-relações.

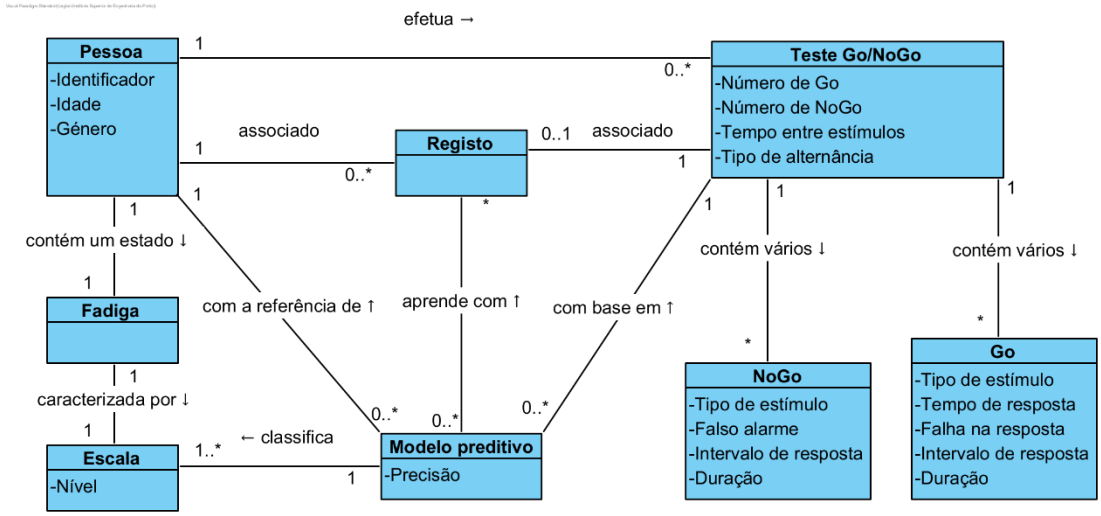

Figura 10 – Modelo de Domínio

Um dos conceitos é a Pessoa, que contem um identificativo que deve ser único, uma idade e um género (masculino ou feminino). Por norma o uso da data de nascimento como atributo é mais correto do que a idade, mas no nosso problema para o registo de um teste Go/NoGo o que realmente importa é a idade. Cada pessoa contém um estado de fadiga, que pode ser caracterizado por uma escala e seu correspondente nível. A pessoa pode efetuar o teste Go/NoGo que contem um número de tarefas ou estímulos Go e NoGo, um tempo entre estímulos, e um tipo de alternância que determina se é efetuado um Go ou um NoGo. O Go é definido pelo tipo de estímulo, o tempo de resposta ao mesmo ou na sua ausência uma falha, o intervalo de tempo durante o qual essa resposta é válida, e a duração durante a qual é visível o estímulo. O NoGo por um tipo de estímulo, a existência de um falso alarme aquando de uma resposta, o intervalo durante o qual uma resposta é possível e a sua duração. Um teste quando associado a uma pessoa pode levar à criação de um registo, e um conjunto dos mesmos são usados recorrendo à aprendizagem automática para a criação de um modelo preditivo. Este modelo preditivo permite, com base em um teste Go/NoGo e com a referência da pessoa que o efetuou, classificar a fadiga prevendo um nível da escala em uso. Note-se que o modelo não prediz a fadiga diretamente, mas o nível da escala, razão pela qual não existe ligação modelo – fadiga. Note-se também que apesar de existirem vários registos e pessoas, por convenção no modelo são representados no singular.

4.2 Requisitos FURPS+

Uma das ferramentas que ajuda a classificar os requisitos funcionais e não funcionais de um software é o FURPS. Este admite cinco categorias, funcionalidade, usabilidade, confiabilidade, desempenho e suportabilidade. A sua posterior extensão, o FURPS+, introduz categorias adicionais relacionadas com especificação suplementar referentes a restrições de design, de implementação, de interface e físicas. Com a ajuda deste sistema de classificação, vamos proceder à sistematização dos requisitos do sistema.

Funcionalidade – Estes requisitos referem-se aos requisitos funcionais (RF), sendo primariamente identificados através dos casos de uso. Assim estão identificados três casos de uso, cujo detalhe se encontra visível na próxima seção:

- RF1 – UC1 Gerir utilizador
- RF2 – UC2 Consultar histórico
- RF3 – UC3 Avaliar fadiga

Usabilidade – Estes requisitos de usabilidade (RU) referem-se a fatores humanos como a beleza estética, a documentação, e são relacionados com o uso do sistema. Assim, da análise de valor surge um valor percebido pelo cliente, uma preocupação com o especto estético do detetor. Dos objetivos do projeto surge outra preocupação com a usabilidade, pelo que o sistema deve ser fácil de usar.

- RU1 - Deve existir uma preocupação estética com o sistema percecionado pelos utilizadores.
- RU2 - O sistema deve ser simples e fácil de usar.

Confiabilidade – Estes requisitos de confiabilidade (RC) referem-se a temáticas como a disponibilidade, segurança, erros e precisão do sistema. Dos objetivos sabemos que o sistema deve prever fadiga, e consequentemente, apresentar uma precisão (taxa de acerto) superior a 50% nesta deteção, o correspondente a um detetor aleatório.

- RC1 - O detetor deve apresentar uma taxa de acerto superior a 50%.

Desempenho – Estes requisitos de desempenho (RD) referem-se à velocidade e eficiência do sistema. Dos objetivos existe a clara necessidade de o processo de deteção ser rápido, com uma duração inferior a um minuto. Mas existe outro requisito deste tipo que pode ser inferido, sabendo que o sistema necessita de ser disponibilizado online e que os custos de alojamento de servidores são função direta dos recursos alocados ao mesmo. Consequentemente, o consumo de recursos a nível do sistema (RAM, CPU) deve ser moderado.

- RD1 - A deteção deve ser efetuada num intervalo de tempo inferior a um minuto.

- RD2 - O sistema deve apresentar um consumo moderado de recursos de forma a reduzir o seu custo de alojamento.

Suportabilidade – Estes requisitos de suportabilidade (RS) referem-se a temas como a flexibilidade, instalação, configuração e localização. Da análise de valor, outro dos valores percebidos pelo cliente é a necessidade de suporte multilingue, nomeadamente Português e Inglês. Assim o sistema deve ser configurável de forma a poder operar tanto em português como em inglês.

- RS1 - O sistema deve suportar a língua Inglesa e Portuguesa.

Requisitos adicionais (RA) – Estes dizem respeito a temáticas relacionadas com o design do sistema, a restrições de implementação a nível de bibliotecas e linguagens, a restrições de interface como a forma como comunica com o exterior, e restrições físicas relacionadas com o hardware onde o sistema deve correr. Dos objetivos, existe a necessidade de o sistema estar disponível online de forma a garantir a sua acessibilidade, ou seja, a interface com o sistema para os utilizadores deve ser possível através de um vulgar navegador de internet. Tendo em consideração que os dados de login e de criação de utilizador vão atravessar a internet, deve existir segurança no protocolo utilizado. Assim é requerido o uso de *Hypertext Transfer Protocol Secure* (HTTPS), uma extensão do *Hypertext Transfer Protocol* (HTTP) que inclui encriptação, na interação do usuário com o sistema.

- RA1 - O sistema necessita ser disponibilizado online, devendo existir essa consideração a nível da implementação e implantação.

- RA2 - A interface entre os utilizadores e o sistema deve ser efetuada por HTTPS.

Requisitos legais – Devido ao RGPD, e às suas obrigações legais, deve ser evitado o uso de dados pessoais no sistema, por exemplo substituindo o nome real da pessoa por uma alcunha de utilizador aquando da criação do utilizador, e da não utilização de outros dados que possam levar à identificação da pessoa, com vista a cumprir o princípio da minimização dos dados através da sua anonimização ou pseudonimização, inscrito no artigo 31 do RGPD. Devendo sempre que possível adotar estratégias compatíveis com a manutenção de dados pessoais, de forma a facilitar a expansão do projeto no futuro. Nomeadamente a nível do consentimento informado do tratamento de dados, e da possibilidade de obter, alterar e apagar os mesmos.

- RA3 - O sistema deve cumprir o princípio da minimização dos dados.

- RA4 -O sistema deve cumprir o RGPD.

4.3 Vista de cenários - Casos de uso

Na vista de cenários apresentamos os casos de uso que representam como a solução é usada na resolução do problema. São identificáveis três casos de uso, a gestão de um utilizador, a avaliação de fadiga e a consulta do histórico de avaliações/testes de fadiga. Um diagrama dos mesmos é visível na Figura 11.

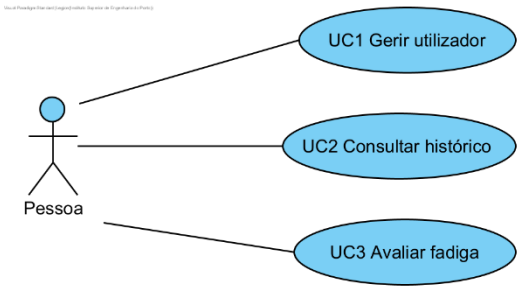

Figura 11— Diagrama de casos de uso

O UC1 "Gerir utilizador" refere-se a gestão de utilizadores, nomeadamente criação de um novo utilizador, *login* e *logout*. É uma tarefa comum em informática, sendo norma não efetuar o detalhe da sua sequência exceto em casos pontuais. Convém referir que durante a criação de um novo utilizador é necessário adquirir os dados referentes a um nome de utilizador, data de nascimento, e o género da pessoa.

O UC2 "Consultar histórico", refere-se apenas à possibilidade de o utilizador visualizar a lista de avaliações efetuadas para si, guardadas no sistema, através de um simples pedido. Apesar de a Figura 12 mostrar um diagrama de sequência para este, o facto de ser um passo básico torna a sua representação desnecessária. Encontrando-se representado para proteger este trabalho de variações ao mesmo, caso surjam novos requisitos fruto de *feedback* dos utilizadores nas suas interações com o sistema.

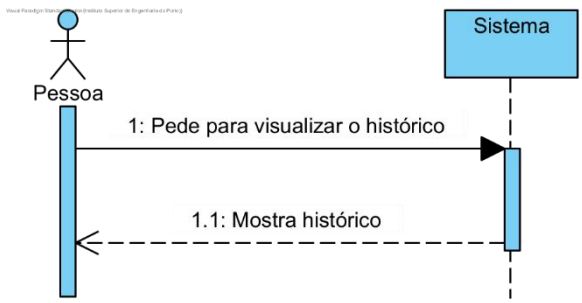

Figura 12 – Diagrama de sequência do caso de uso UC2 "Consultar histórico"

O caso de uso, UC3 "Avaliar fadiga" corresponde à sequência principal de uso do sistema, onde um utilizador pretende ver a sua fadiga avaliada efetuando um teste Go/NoGo. O detalhe deste teste é mostrado sob a forma de um diagrama de atividade conforme visível na Figura 13, que permite uma representação mais clara que um diagrama de sequência.

Inicia quando uma pessoa indica que pretende avaliar a sua fadiga, sendo apresentado pelo sistema uma série de tarefas Go ou NoGo aleatoriamente ao utilizador. O sistema pode pedir à pessoa para responder mostrando um estímulo Go, ou se abster de responder com um estímulo NoGo. Após o teste terminar, o sistema mostra uma escala de fadiga e pede à pessoa para responder com o nível de fadiga em que pensa encontrar-se. O sistema então efetua uma previsão do nível com base nos dados do teste e informação referente à pessoa, e mostra-lhe o resultado do teste e a previsão efetuada. A razão pela qual é pedido uma auto estimativa de fadiga, é para poder confrontar o utilizador, comparando a sua autoavaliação com o teste e a previsão resultante do mesmo.

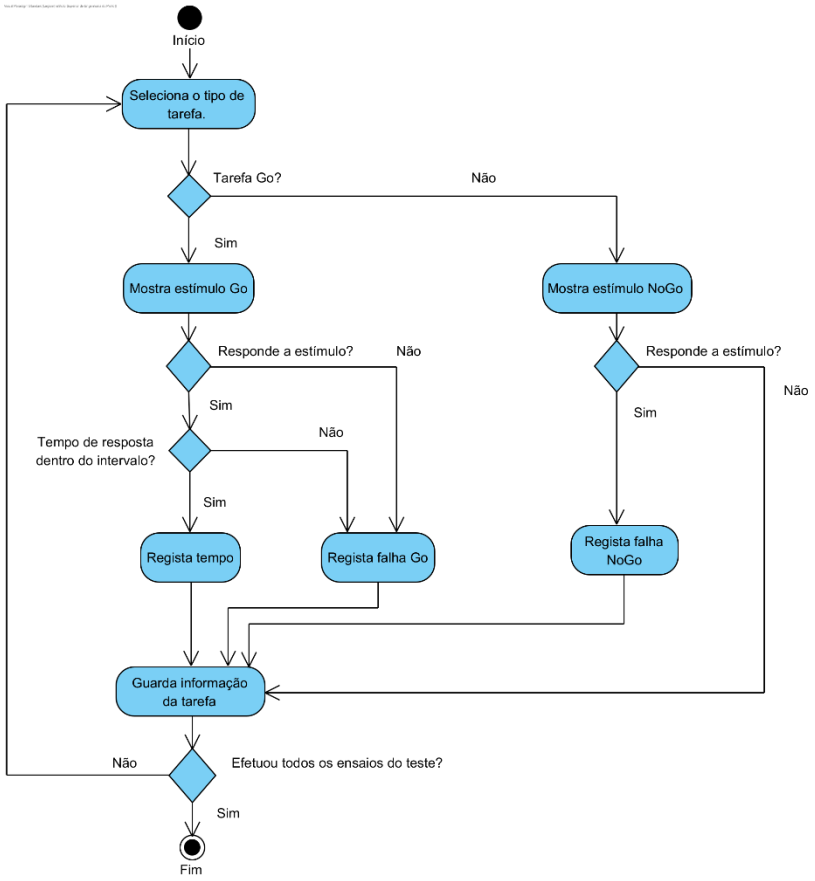

Figura 13 – Diagrama de atividade do teste Go/NoGo

Conforme visível no diagrama de atividade do teste Go/NoGo, no início é selecionado o tipo de tarefa de uma forma aleatória de entre o número de tarefas Go e NoGo disponíveis. Caso esta seja Go, é registado o tempo de resposta que ocorre dentro do intervalo de aceitação, senão é registada uma falha Go. No caso NoGo, apenas é registada uma falha quando ocorre resposta ao estímulo. Após ser guardada a informação da tarefa, caso seja a última termina-se o teste, senão é selecionada um novo tipo de tarefa.

4.4 Vista lógica

A vista lógica preocupa-se com o design do modelo de objetos, sendo visível normalmente um diagrama de classes, como o visível na Figura 14, onde explicamos a lógica de modelação das entidades do sistema que consideramos mais importantes.

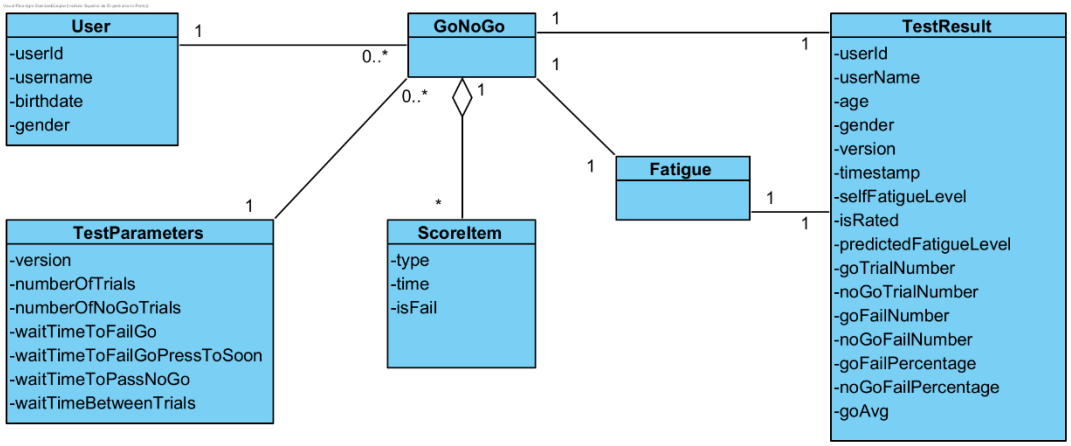

Figura 14 – Diagrama de classes

O utilizador, classe User é composto por um id, um nome de utilizador, uma data de nascimento e o seu género. Encontrando-se relacionado com a classe responsável por efetuar os testes Go/NoGo, a GoNoGo. Está esta relacionada com a classe Fatigue que avalia o nível de fadiga. O teste está relacionado com os seus parâmetros, TestParameters, que contém informação como a versão, o número de ensaios, o número de ensaios NoGo, o tempo de espera até a falha Go ocorrer, o tempo mínimo para aceitar o ensaio Go, o tempo que demora um ensaio NoGo e o tempo de espera entre ensaios. Durante cada ensaio do teste é guardado um item referente à pontuação deste ensaio, ScoreItem, composto pelo tipo de ensaio, tempo no ensaio e se o ensaio corresponde a uma falha. O teste no final produz um resultado, o TestResult, que contem o id do utilizador, o seu nome de utilizador, a idade quando efetuou o teste, o género, a versão do teste Go/NoGo, a data e tempo em que se realizou o teste, o nível de fadiga indicado pelo utilizador, se existe uma previsão para este teste e qual o nível previsto, o número de ensaios Go e de ensaios NoGo, o número de ensaios falhados Go e NoGo, a percentagem de falhas Go e NoGo, e o tempo médio de resposta aos ensaios Go.

Note-se que no resultado do teste, a versão do mesmo é incluída com o racional de proteger o sistema de variações. Caso no futuro os parâmetros do teste mudem, a versão permite diferenciar os resultados de maneira que os mesmo sejam apenas usados com o modelo de previsão correspondente. A razão pela qual não são visíveis classes associadas ao modelo de previsão deve-se ao facto de estas serem implementadas pelas *frameworks* de aprendizagem automática, cujo detalhe não é passível de inferência nesta modelação lógica.

4.5 Vista de implementação

A vista de implementação ou desenvolvimento apresenta a visão do sistema que mais diz respeito ao programador, nomeadamente a organização do software por módulos. Um dos diagramas que nos permite obter uma visão rápida da lógica de implementação destes módulos é o diagrama de componentes. Cada um destes componentes deve ser em teoria um modulo intercambiável, capaz de ser facilmente substituído por outro de função idêntica. Assim apresentamos duas alternativas de design, a monolítica visível na Figura 15 e a arquitetura em camadas e serviços na Figura 16. Cada uma delas apresentando os seus benefícios e sacrifícios em termos de valor para o projeto. Os componentes comuns a ambas são o componente de gestão de utilizadores, responsável por lidar com o registo, *login* e *logout* de utilizadores, o componente de teste Go/NoGo e o componente da escala de fadiga. Note-se que existe um componente de linguagem, de forma a permitir o suporte multilingue. Existem dois componentes de base de dados, o de utilizadores responsável por guardar os dados dos utilizadores, e o de fadiga por armazenar os dados relativos à fadiga. Estes encontram-se separados de forma a aumentar a modularidade e segurança do sistema. O modelo de fadiga é o componente responsável pela classificação da fadiga. Note-se que a comunicação com o sistema é feita através de uma interface gráfica de utilizador ou *Graphical User Interface* (GUI).

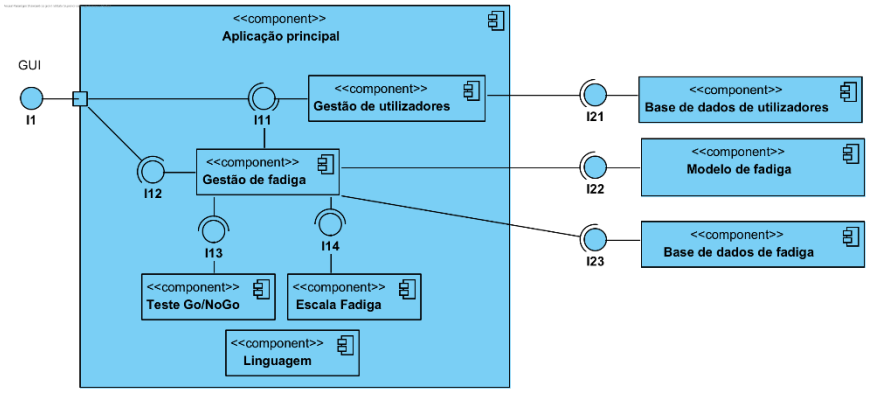

Figura 15 – Diagrama de componentes I Monolítico

Na opção monolítica, existe a prevalência de uma aplicação principal, responsável pela gestão de utilizadores e gestão de fadiga. Os requisitos de acesso online obrigam à existência de um *front-end*, que será esta aplicação. O modelo de fadiga, visto ser desenvolvido posteriormente, fica à parte da aplicação, permitindo a sua fácil substituição. A grande vantagem desta arquitetura é que permite poupar tempo no seu desenvolvimento, concentrando os esforços principais numa única aplicação e reduzindo o número de interfaces o que leva a uma menor complexidade do sistema. Mas acarreta um sacrifício em termos de modularidade, ficando o sistema mais acoplado entre si.

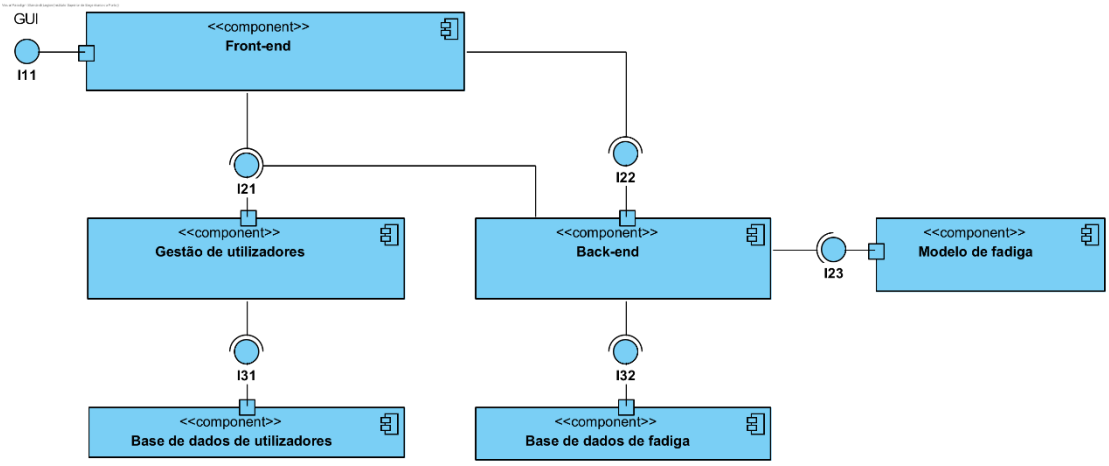

Figura 16 – Diagrama de componentes II Arquitetura por camadas e serviços

Na opção de arquitetura por camadas, subdividimos a solução em três camadas, uma de apresentação, o *front-end*, outra intermédia responsável pela lógica principal e interligações, e uma terceira responsável pelo armazenamento de dados. O *front-end* é responsável pela apresentação do teste e da escala de fadiga, assim como da gestão da linguagem. A gestão de utilizadores passa para a segunda camada, ficando também com a responsabilidade de autenticação. Visto a maioria dos componentes estarem separados e comunicarem entre si, existe a necessidade no *back-end* de autenticar o pedido garantindo a segurança dos dados a tratamento. Este componente de *back-end* é responsável pela lógica da gestão de fadiga e de interagir quer com o módulo de fadiga usado para a sua classificação, quer com a base de dados que armazena a informação. Note-se que cada componente principal apresenta uma interface que pode ser por exemplo HTTP ou *Structured Query Language* (SQL) no caso das bases de dados, e comunicar usando o protocolo *Transmission Control Protocol* e *Internet Protocol* (TCP/IP). Ou seja, cada um deles encontra-se a disponibilizar um serviço, um serviço de gestão de utilizadores e autenticação, um serviço de modelação de fadiga, um serviço de gestão de fadiga, e um serviço de armazenamento e acesso a dados. Serviços esse que se encontram subdivididos por camadas de acesso. Assim esta arquitetura em adição a camadas, também se encontra orientada a serviços. A sua grande vantagem é a clara separação de responsabilidade e elevada modularidade, o que permite teoricamente a substituição de componentes a qualquer momento. Origina um elevado desacoplamento lógico, promove a coesão dos módulos, e aumenta a segurança. Como sacrifício, esta arquitetura acarreta uma maior complexidade e uma menor performance. A elevada complexidade devida às inúmeras interfaces requer um maior tempo de codificação, o uso de uma maior variedade de *frameworks*, e cria uma maior carga no servidor levando a um aumento de custos operacionais. Outra grande vantagem desta arquitetura é que existem várias frameworks de serviços de gestão de utilizadores com autenticação já disponíveis no mercado, permitindo evitar reinventar a roda e diminuir o tempo de desenvolvimento.

Apesar de ambas as arquiteturas apresentarem vantagens e desvantagens, e do facto de a arquitetura monolítica representar uma redução do tempo de codificação bastante apetecível,

a opção correta é a escolha da arquitetura por camadas e serviços. O uso de arquiteturas orientadas a serviços domina o panorama atual de novos desenvolvimentos quer na internet quer no mundo empresarial. Mas a principal razão para esta escolha é a enorme flexibilidade à mudança, que será uma mais-valia neste projeto a médio prazo. Permite proteger o sistema caso existam novos requisitos ou alterações dos mesmos, através de uma mais fácil integração de mudanças.

A nível de organização dos diferentes pacotes, tendo em conta a arquitetura escolhida é possível desde já subdividir esta organização em quatro grandes grupos conforme visível na Figura 17. O pacote de *front-end* onde os pacotes relacionados com o teste e com a escala dependem do pacote da linguagem e das suas traduções. O pacote referente a *framework* de gestão de utilizadores, que contem a framework completa, não sendo prudente especular qual a sua divisão interna. O pacote de gestão de fadiga, que contem o código de comunicação e interação com os outros serviços. E finalmente o pacote referente ao modelo de fadiga através de aprendizagem automática. Este contem o código do serviço HTTP que depende do modelo de predição, que por sua vez depende do *dataset* existente. Note-se que não existe interdependências entre os quatro grandes pacotes visto corresponderem a aplicações de serviços diferentes, sendo espectável que a cada uma corresponda um repositório diferente.

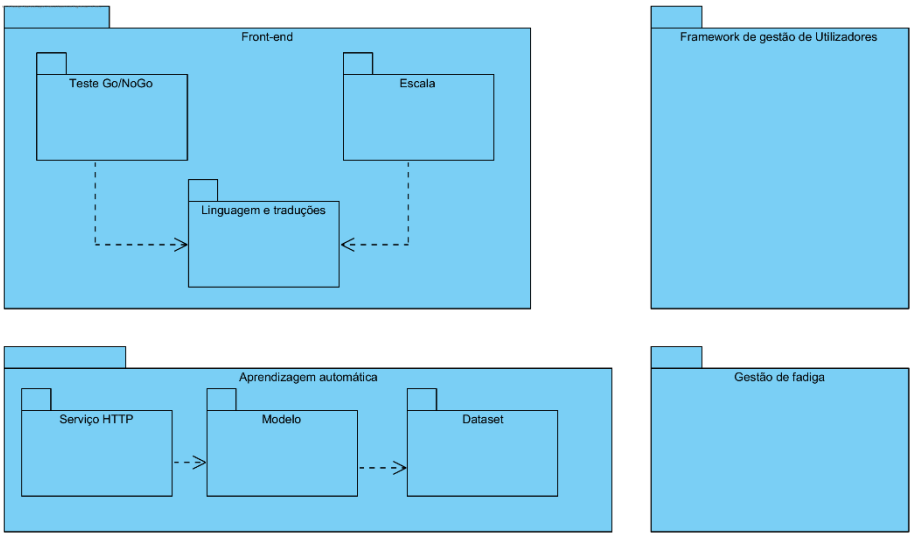

Figura 17 – Diagrama de pacotes

4.6 Vista de processos

A vista de processos é uma vista que captura aspetos do design relacionados com concorrência e sincronização, pelo que optamos por mostrar o fluxo no sistema recorrendo a um diagrama de sequência do sistema conforme visível na Figura 18.

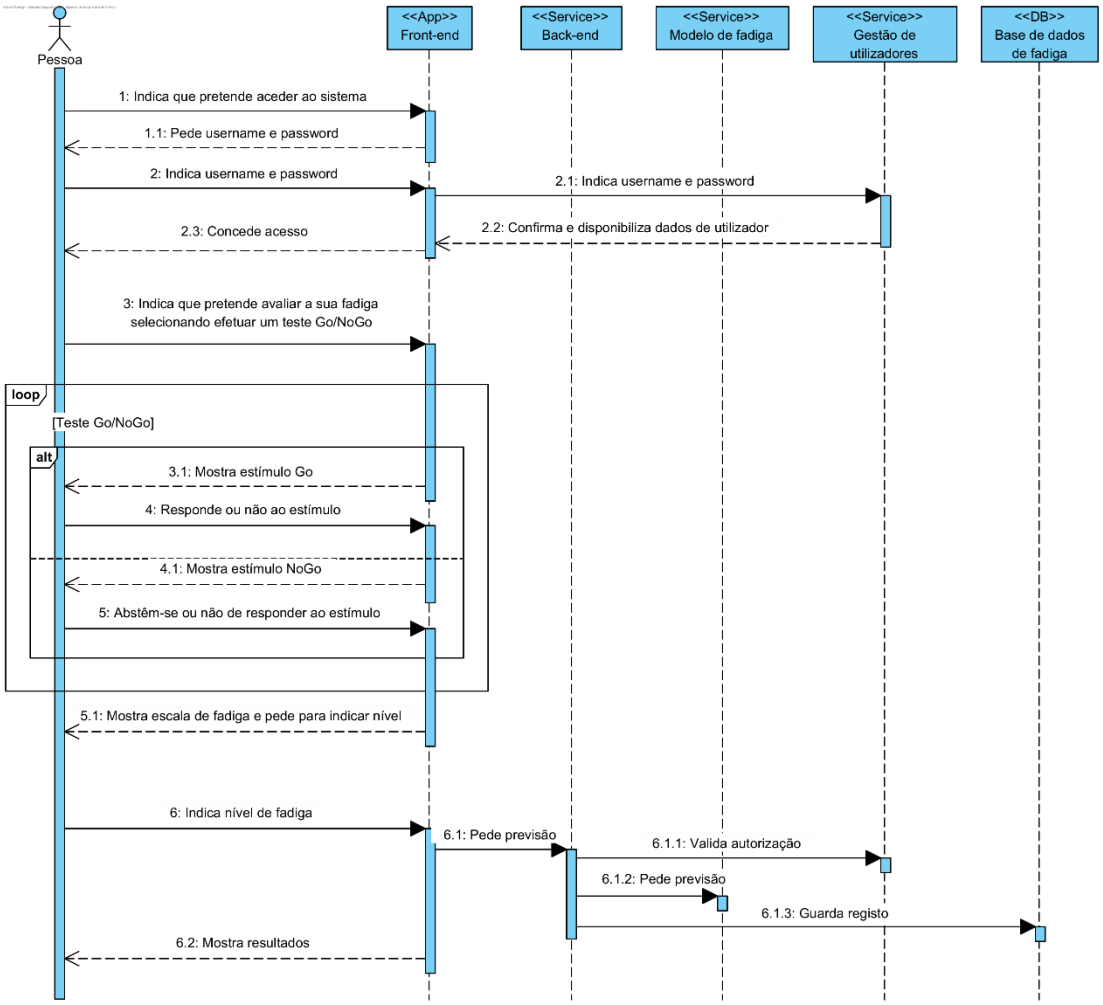

Figura 18 - Diagrama de sequência de sistema

Neste detalhe do design explicitamos a forma como se processa o *login*, mostrando a sua interação com um serviço de gestão de utilizadores, que mais tarde é usado para validar o pedido de previsão de fadiga, garantindo que os registos ficam corretamente associados. Note-se que apesar de não representado para este nível de granularidade do sistema, o teste Go/NoGo e o questionário de fadiga são efetuados em janelas diferentes do *front-end*, aumentando a modularidade, a coesão, reduzindo as interdependências, e protegendo o *front-end* em caso de variações. Relativamente à base de dados de fadiga e modelo de fadiga, o acesso a estes é efetuado através do serviço de *back-end*, garantindo o seu isolamento do exterior e a segurança dos dados.

Em adição à sequência de processos ligados ao software do sistema existe também outro processo, mas de natureza parcialmente manual, o da criação do *dataset* e do modelo de previsão de fadiga. Na Figura 19 é possível ver um diagrama de BPMN que representa como decorre este processo. Neste são visíveis diferentes faixas correspondentes ao sistema, ao investigador principal e aos sujeitos que participam na colheita de dados. O investigador cria o programa de teste que é usado pelos sujeitos, originando um conjunto de dados armazenados

pelo sistema. Estes dados são analisados de forma a criar um *dataset* e extrair informação que permita selecionar as características a alimentar aos modelos escolhidos. O sistema treina automaticamente estes modelos, tendo o investigador o cuidado de afinar os seus parâmetros, de forma a aumentar a sua taxa de acerto. Após uma avaliação o melhor modelo é selecionado e incorporado no software permitindo a deteção de fadiga. Caso nenhum modelo seja aceitável, existe um retrocesso que implica a escolha de novos modelos e/ou características.

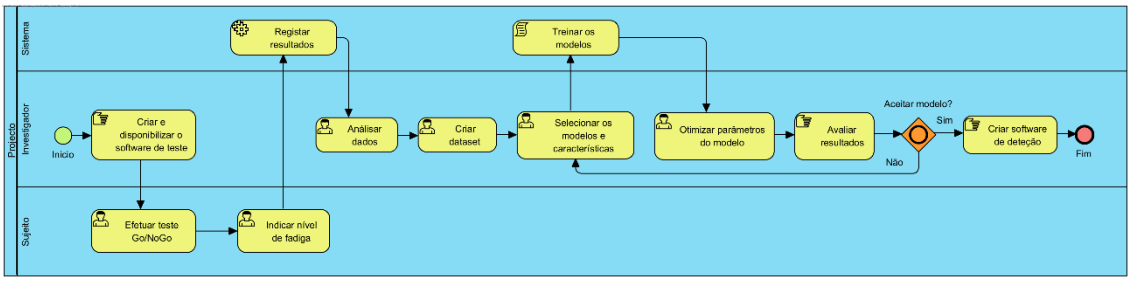

Figura 19 – Diagrama de BPMN

4.7 Vista de implantação

A vista de implantação preocupa-se com informação mais direcionada aos engenheiros de sistemas, mostrando o detalhe do mapeamento do software pelas máquinas físicas, refletindo a sua distribuição. Nesta vista apresentamos duas alternativas, uma referente a uma opção com um único servidor para todo o sistema visível na Figura 20 e outra onde os vários módulos se encontram alojados em vários servidores da mesma rede conforme visível na Figura 21.

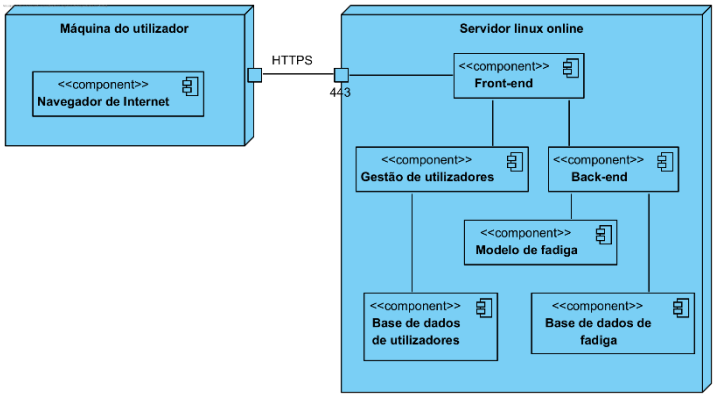

Figura 20 – Diagrama de implantação I com servidor único

A primeira opção apresenta uma elevada vantagem económica, devido aos custos reportarem apenas a um único servidor, sendo que as empresas de alojamento cobram por servidor. Outra vantagem é o facto de os recursos serem partilhado, pelo que permite um melhor balanço dos mesmos.

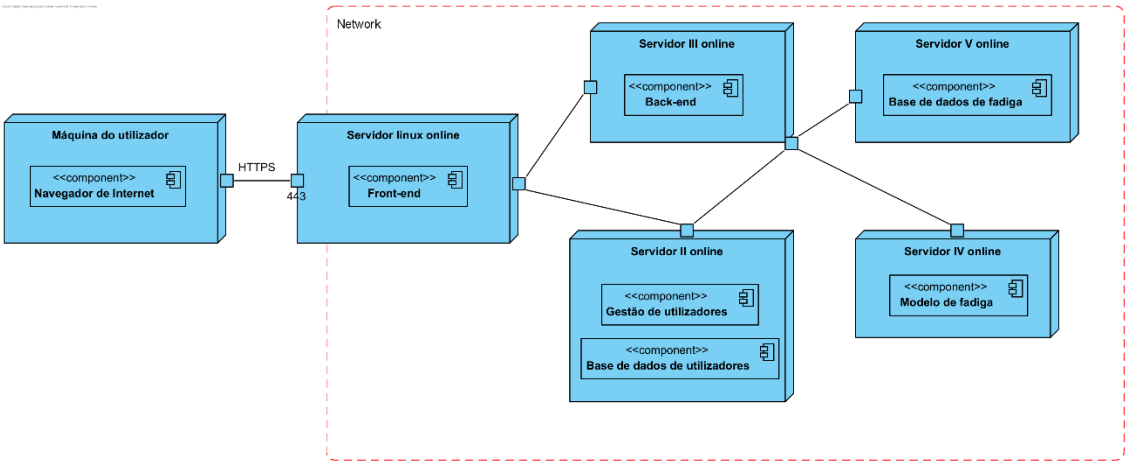

Figura 21 – Diagrama de implantação II com vários servidores

A segunda opção, com recurso a vários servidores, apresenta uma clara vantagem a nível de escalabilidade, modularidade e manutenção. Possibilita o uso de tecnologia de virtualização disponibilizada pelas companhias de alojamento, o que permite rapidamente escalar os servidores em casos de maior carga. O facto de existir virtualização torna a disponibilização e gestão dos módulos mais rápida e eficiente. Mas introduz sacrifícios quer a nível financeiro, com cada ambiente de virtualização a ser cobrado em separado, quer a nível do aumento do tempo de desenvolvimento e da necessidade de uma máquina de desenvolvimento mais poderosa para suportar a virtualização localmente.

Visto existirem requisitos a nível económico, mas não a nível de escalabilidade do sistema, a melhor opção é que recorre ao uso de um único servidor, não inviabilizando um trabalho futuro de expansão para um sistema com virtualização. Note-se que a escolha por um servidor Linux é clara, devido a não existirem custos de licenciamento deste sistema operativo, comparativamente com um servidor Windows.

4.8 Frameworks

Uma das grandes decisões no design do sistema é a escolha da *framework de frontend* a utilizar. Em 2022 destacam-se quatro pela sua utilização no mercado, React, Angular, Vue.Js e JQuery. A Tabela 10 ilustra uma comparação entre estas *frameworks* recorrendo a critérios como a performance, a facilidade de aprendizagem, a documentação existente, a presença de uma comunidade capaz de prestar apoio, a linguagem, as capacidades ou funcionalidades da mesma e a experiência do autor na *framework*. Utilizando um sistema de uma a três estrelas para pontuar cada um destes critérios, de forma a minimizar a subjetividade da escolha.

Tabela 10 – Comparação de *frameworks* de *frontend*

Critério	**React**	**Angular**	**Vue.Js**	**JQuery**
Performance	★★★	★★★	★★★	★
Linguagem	JavaScript	TypeScript	TypeScript	JavaScript
Capacidades	★★	★★★	★★	★
Aprendizagem	★★	★	★★	★★★
Documentação	★	★★★	★	★★★
Comunidade	★★★	★★★	★	★★★
Experiência	★	★★★	★	★

Uma análise da comparação revela que o Angular é a que aparenta ser mais favorável. Este facto é expectável, sendo que o Angular e o React são as *frameworks* mais populares neste momento para aplicações web. Por eliminação de escolhas, o JQuery com a sua baixa performance e capacidades é a primeira a ser excluída. O Vue.Js é o próximo a ser excluído devido a muita da comunidade e documentação serem em chinês, o que dificulta a aprendizagem e resolução de problemas. Restando o React e Angular, ambos bons candidatos a serem a *framework* usada. Ambos apresentam uma boa performance e uma comunidade sólida. Apesar de o React ser mais fácil de aprender, a sua documentação é mais pobre e o autor já possui experiência em Angular o que nega a sua principal desvantagem. O uso de TypeScript também é uma vantagem, visto permitir uma melhor estrutura do código e reduzir erros. Em termos de capacidades, o Angular apresenta mais capacidades, que apesar de não serem essenciais, facilitam o desenvolvimento da aplicação. Assim a escolha recai sobre o Angular.

Na escolha da framework de aprendizagem automática existem duas escolhas, o R e o Python, mas nada nos obriga escolher apenas uma destas *frameworks*. Tendo em conta que a informação presente no estado da arte indica uma melhor adequação do R a fase de experimentação e análise exploratória de dados, e que o Python é mais forte no desenvolvimento de modelos de aprendizagem automática, a escolha é obvia. O ideal é usar o R na análise inicial de exploração de dados, e o Python para correr e integrar o modelo de previsão.

O Keycloak é um software de gestão de utilizadores chaves na mão que é sugerido por vários especialistas como a (Sennovate, 2022). O uso de um software já estabelecido no mercado permite assegurar um maior nível de segurança e um melhor cumprimento do RGPD, comparativamente a um desenvolvimento próprio. O Keycloak apresenta várias vantagens, como a existência de um ambiente gráfico configurável de criação de utilizadores, uma base de dados própria de raiz, e um sistema robusto de autenticação baseado em *web tokens*. Além de ser completamente gratuito, o autor possui alguma experiência na sua configuração.

Por estes motivos, faz sentido substituir o componente de gestão de utilizadores por uma única peça de software, o Keycloak.

4.9 Breve sumário

Neste capítulo efetuamos a análise de requisitos recorrendo ao método FURPS+ e o design da solução através do modelo de 4+1 vistas. Identificamos três casos de uso, Gerir utilizador, Consultar histórico e Avaliar fadiga, sendo o teste Go/NoGo descrito através de um diagrama de atividade. Com recurso a diagramas de componentes, de sequência de sistema e de implantação, várias opções são consideradas sendo escolhida uma arquitetura de camadas e serviços, em oposição a uma mais monolítica, com implantação em um único servidor por motivos económicos. A *framework* Angular é selecionada para a construção do *frontend*, o software Keycloak para o sistema gestão de utilizadores, e a nível da aprendizagem automática o R é selecionado para a análise inicial e o Python para correr e integrar o modelo de previsão.

5 Solução implementada

Este capítulo expõe a solução final obtida de forma a atingir os objetivos propostos, descrevendo o sistema final através da descrição da sua implementação e implantação, e uma breve descrição do processo de testes.

5.1 Implementação

A solução criada segue na sua essência o design inicial, com algumas pequenas alterações derivadas da transformação de algo teórico em algo concreto. No diagrama de componentes final detalhado na Figura 22, é visível uma descrição geral da solução.

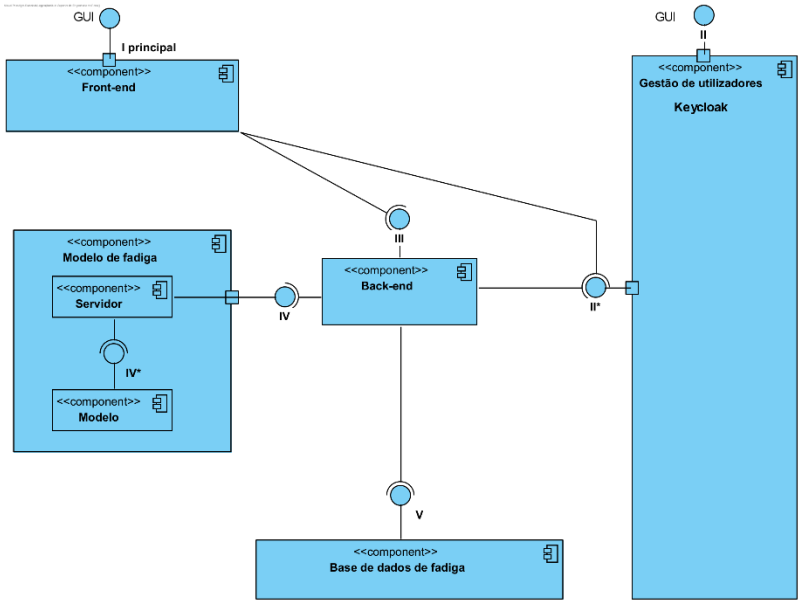

Figura 22 - Diagrama de componentes final

O uso de um navegador de internet permite aceder ao *front-end* através da interface gráfica e efetuar o login e o teste de fadiga. O componente de gestão de utilizadores com base no Keycloak permite o registo de novos utilizadores, e a sua autenticação. O componente de *back-end* orquestra a previsão de fadiga e o acesso a base de dados dos registos. O componente do modelo de fadiga permite efetuar a previsão de fadiga com base nos dados do teste Go/NoGo, e a base de dados de fadiga permite o registo e consulta dos dados.

5.1.1 Gestão de utilizadores

A gestão de utilizadores é suportada pelo Keycloak, um software dedicado a esta função. Ele permite, através de uma interface web, o registo de novos utilizadores e a sua gestão, incluindo a alteração de preferências linguísticas. Este contém uma interface do tipo *Representational State Transfer* (REST) que permite autenticar um utilizador com base no seu nome de utilizador e palavra-chave, gerando um Json web *token* que é utilizado pelo *front-end* e *back-end*. O Keycloak inclui uma base de dados gerida pelo mesmo que permite guardar todos os dados relacionados com o utilizador. Apesar de o design inicial separar em camadas a gestão de utilizadores da sua base de dados, esta separação existe a nível interno do Keycloak, existindo módulos de *front-end* que interagem com módulos de internos de autenticação e gestão REST, que por sua vez interagem com a base de dados.

5.1.2 Front-end

O *front-end*, criado em Angular, é um servidor web do tipo *single page application* que corre no navegador de internet do usuário. Este suporta os casos de uso RF2 – Consultar histórico e RF3 – Avaliar fadiga, sendo caso de uso RF1 – Gerir utilizador suportado pelo componente anterior. Em adição as três grandes responsabilidades ou componentes internos iniciais previstos (Teste Go/NoGo, Escala de Fadiga e Linguagem), foi necessário subdividir responsabilidades por mais componentes devido a natureza modular do Angular, com o intuito de aumentar a coesão e reduzir acoplamento. O componente de utilizador é responsável pelo *login*, visualização da sua informação e redireccionamento para o Keycloak para registo de novo utilizador (sendo este o único momento em que existe a necessidade de um utilizador interagir visualmente com o Keycloak). Na Figura 23 é visível o aspeto visual aprumado da página inicial de login, responsabilidade deste componente interno. Este aspeto estético é o resultado da introdução do Bootstrap, uma das mais populares *frameworks* de estética de desenvolvimento web.

Figura 23 – Visão da página de login

O componente de linguagem é responsável pela seleção da linguagem, recorrendo à biblioteca "ngx-translate" para gerir as traduções existentes em dois ficheiros, um para português e outro para inglês. O componente teste Go/NoGo é responsável por apresentar o teste Go/NoGo conforme visível na Figura 24, e o de fadiga por apresentar a escala de fadiga.

Figura 24 – Visão do início do teste Go/NoGo

Os resultados são mostrados com recurso ao componente de resultados. Existe um novo componente de treino criado a pedido dos voluntários da recolha de dados, que permite efetuar um treino correndo unicamente o teste Go/NoGo sem qualquer seguimento. Foi adicionado o componente de histórico de forma a lidar com a responsabilidade de permitir a visualização do histórico recente de testes.

O teste GoNoGo implementado inclui 20 tarefas, 16 de Go e 4 de NoGo que se alternam aleatoriamente com um intervalo de 500 ms. A janela de aceitação de resposta válida de Go inicia aos 80 ms e termina aos 500 ms, sendo o estímulo uma elipse verde em fundo preto

cuja resposta é efetuada através da tecla barra de espaços. O estímulo NoGo é apresentado como uma elipse vermelha com a duração de 2000 ms.

5.1.3 Back-end

O *back-end* é um servidor criado em C# .Net 5 que permite a orquestração de dois serviços para o *front-end*. O primeiro de avaliação e registo de fadiga, que recebe a informação do resultado do teste Go/NoGo, confirma a autenticação do utilizador no Keycloak através do web *token*, obtém a previsão de fadiga do serviço do modelo de fadiga, regista o resultado na base de dados, e por fim retorna a informação com a previsão ao *front-end*. O segundo serviço permite obter o histórico de resultados, ocorrendo primeiro a autenticação do utilizador através do *token*, seguido de uma consulta à base de dados onde são obtidos os resultados para os dias pretendidos. A escolha do uso de .Net foi devida à facilidade da sua utilização, a várias experiências positivas do autor com o uso da mesma, ao seu suporte multiplataforma que inclui Linux, à existência de vastas bibliotecas de apoio, e à enorme abundância de informação sobre a mesma.

5.1.4 Base de dados de fadiga

A base de dados para registo de informação referente aos testes Go/NoGo e fadiga foi desenvolvida em SQL Server, consequência da sua fácil integração com o servidor .Net, da experiência do autor e do seu suporte em Linux. O diagrama visível na Figura 25, mostra a estrutura dos registos existentes na base de dados, onde consta apenas uma única tabela, a TestResult. A informação especifica dos utilizadores, não referente a um resultado, não consta nesta base de dados por motivos de segurança.

TestResult

Column Name	Condensed Type	Nullable
id	int	No
userId	nvarchar(50)	No
username	nvarchar(50)	No
rgpd	nvarchar(50)	No
timestamp	datetime	No
age	int	No
gender	nvarchar(50)	No
version	nvarchar(50)	No
selfFatigueLevel	int	No
isRated	bit	No
predictedFatigueLevel	int	No
gotrialnumber	int	No
nogotrialnumber	int	No
gofailnumber	int	No
nogofailnumber	int	No
gofailpercentage	float	No
nogofailpercentage	float	No
goavg	float	No

Figura 25 – Diagrama da estrutura dos registos da base de dados

5.1.5 *Modelo de fadiga*

O modelo de fadiga é um servidor escrito na linguagem Python 3 com recurso a biblioteca "Flask". Este é composto pelo componente do servidor propriamente dito e por dois modelos de classificação de fadiga disponibilizados como serviços. Os modelos foram gerados com a biblioteca "scikit-learn" e guardados como ficheiros binários com recurso a biblioteca "Pickle". Isto permite evitar o treino dos modelos pelo servidor, que se limita a carregar os ficheiros binários para memoria durante o arranque. O primeiro modelo permite classificar a fadiga em três classes (fresco, cansado e muito cansado) recorrendo ao algoritmo *Random Forest* com uma taxa de acerto de 57% (F1-Score de 54% e Kappa de 29%). O segundo modelo permite classificar a fadiga em duas classes, extremamente cansado e não extremamente cansado, com uma taxa de acerto de 89% (F1-Score de 79% e Kappa de 59%) recorrendo ao mesmo tipo de algoritmo. Apesar de existirem dois modelos, o serviço do *back-end* recorre apenas ao serviço do segundo modelo para efetuar previsões, devido proporcionar mais classes de discriminação e a ter atingido o objetivo mínimo de taxa de acerto de 50%.

5.2 Implantação

O sistema foi implantado em um único servidor com o sistema operativo Ubunto 18.04 com dois processadores virtuais e 4 GB de memoria RAM, estando acessível em https://www.gonogoapp.org e hospedado na plataforma Google Cloud. A visão desta implementação é visível na Figura 26.

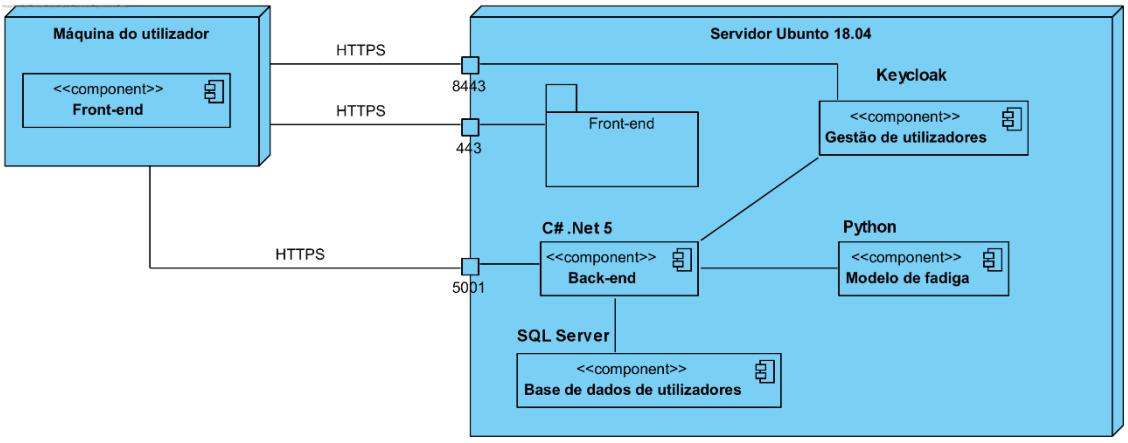

Figura 26 - Diagrama de implantação final

O sistema é acedido externamente através do protocolo HTTPS recorrendo a um certificado valido criado pela autoridade "Let's Encrypt" para o domínio gonogoapp.org. O *front-end* é disponibilizado no porto 443, o normal para tráfego web HTTPS. A gestão de utilizadores Keycloak é acedida no porto 8443 por HTTPS, e o *back-end* no porto 5001 por HTTPS. Note-se que o *front-end* Angular, como grande parte da tecnologia Web, corre no próprio navegador de internet do utilizador, sendo os seus ficheiros obtidos pelo porto 443 no servidor,

conforme representado no diagrama. Assim as chamadas ao *back-end* e Keycloak são efetuadas a partir da sua máquina, estando assim representadas no diagrama.

5.2.1 Testes

Durante o desenvolvimento do sistema e aquando da sua implantação, foram efetuados testes de forma a comprovar a funcionalidade do mesmo. Os componentes de *front-end* devido a sua componente gráfica de elaboração do teste Go/NoGo foram testados manualmente, seguindo o normal decorrer dos casos de uso. Mas os restantes componentes com serviços foram testados recorrendo a testes de integração desenvolvidos na aplicação Postman, uma aplicação muito utilizada na indústria para estes tipos de testes. O Código 1 representa um extrato de um teste nesta ferramenta, que valida a existência de vários atributos essenciais no serviço de obtenção de tokens do Keycloak, validando a sua configuração.

```
var data = pm.response.json();
const responseJson = pm.response.json();
pm.test("Status code is 200", function () {
  pm.response.to.have.status(200);
});
pm.test("Has access_token", () => {
  pm.expect(responseJson.access_token).not.null;
});
pm.test("Has refresh_token", () => {
  pm.expect(responseJson.refresh_token).not.null;
});
pm.test("Has access_token content", () => {
   pm.expect(responseJson.access_token).not.null;
   var parsed = JSON.parse(atob(responseJson.access_token.split('.')[1]));
    pm.expect(parsed.sub).to.be.a("string")
    pm.expect(parsed.preferred_username).to.be.a("string")
    pm.expect(parsed.gender).to.be.a("string")
    pm.expect(parsed.birthdate).to.be.a("string")
    pm.expect(parsed.lingua).to.be.a("string")
    pm.expect(parsed.rgpd).to.be.a("string")
});
```

Código 1 – Extrato de código de um teste no Postman

5.3 Breve sumário

Este capítulo expõem a solução final, descrevendo o sistema que é composto por cinco grandes componentes, um *frontend*, um *backend*, o software Keycloak de gestão de utilizadores, o modelo de fadiga e a base de dados de fadiga. Este encontra-se implantado em um único servidor online com dois processadores e 4 GB de memória RAM, acessível através de HTTPS em três portos distintos que disponibilizam acesso a serviços. A integração destes serviços pode ser testada com recurso a testes elaborados no software Postman.

6 Experimentação e Avaliação

Este capítulo aborda a experimentação e avaliação da solução obtida. Inicia por indicar qual a hipótese ou pergunta a investigar e que resposta se espera obter como resultado. Segue a indicação, racional, e explicação do método de investigação usado, a "Investigação Ação". É exposta a metodologia de validação da solução com base numa avaliação que recorre à *framework* de avaliação da qualidade (QEF) e na análise da taxa de acerto e matrizes de confusão dos modelos de previsão. Finalmente é indicada a forma de disseminação do trabalho, através da defesa e disponibilização desta tese de mestrado.

6.1 Hipótese e resposta esperada

Com base no problema explicado na introdução deste trabalho é possível resumir a premissa inicial numa única pergunta ou hipótese:

"Será possível criar um detetor de fadiga usando testes Go/NoGo e aprendizagem automática?"

Esta pergunta explica claramente o objetivo principal que se pretende atingir. E a resposta que melhor se adequa a esta pergunta é a criação de uma solução específica de software, que com base em testes Go/NoGo e aprendizagem automática, permite detetar a fadiga em pessoas.

6.2 Método Investigação Ação

O método de investigação escolhido para este trabalho foi o Action Research ou Investigação Ação, com base no trabalho de (Staron, 2020). Este é composto por ciclos consecutivos, que na sua forma simplificada englobam apena duas fases, a de avaliação da situação e a de intervenção no problema. Na sua forma canónica é composto por cinco fases, o diagnóstico, o planeamento de ações, a tomada de ação, a avaliação, e a aprendizagem.

A fase de diagnóstico aborda a questão de "Qual é exatamente o problema real?", sendo cada ciclo iniciado por esta identificação do problema a resolver no ciclo. Tendo em conta que este problema deve ser limitado na sua abrangência. A fase seguinte é a de planeamento de ações, onde os investigadores trabalham em conjunto para identificar quem faz o quê. Sendo planeado que dados vão ser recolhidos e como. Segue a fase de tomada de ação, onde são efetuadas ações e observadas as consequências das mesmas. Os efeitos dessas ações são recolhidos e utilizados na fase seguinte, a de diagnóstico, onde através de métodos estatísticos, os dados são analisados e apresentados à equipa. Caso os dados sejam inconclusivos, segue um planeamento de recolha de dados adicionais para os próximos ciclos. A fase final do ciclo é a de aprendizagem, onde a especificação da aprendizagem obtida no ciclo é efetuada. Podendo ser produzidos variados documentos como guias, publicações científicas, e *wikis* de boas práticas.

Uma das forças deste método é a sua flexibilidade e natureza iterativa, onde a mudança é abraçada, com abertura a ajustes do plano. Este plano existe em concreto, mas os intervenientes estão cientes que este pode mudar, fruto da disponibilidade dos dados, de alterações na companhia, ou de fatores externos. Outras das forças relevantes deste método é a sua ênfase na visualização de dados através de diagramas, gráficos e outras representações de visualização mais avançadas, que permitem perceber e diagnosticar o problema com maior eficácia. Esta ênfase nas visualizações permite dirigir a discussão do campo das especulações para uma discussão dos dados existentes.

Esta flexibilidade à mudança e natureza iterativa do método foram as principais razões que levaram a escolha do método Investigação Ação como método de investigação deste trabalho. Nota-se em parte uma semelhança ao processo iterativo de desenvolvimento de software lecionado no ISEP, que permite uma rápida adaptação em ambientes menos estáticos.

6.3 Atores

Um projeto de investigação segundo o método de Investigação Ação é composto por pessoas, denominadas atores, que compõem a equipa de ação, podendo tomar papeis como praticantes de ações ou de investigadores. Assim os atores deste projeto são os que seguem:

O investigador principal, responsável pela documentação do projeto, desenvolvimento do software, avaliação da solução, e pela geração de dados de fadiga.

A orientadora, com o papel de revisão da documentação do projeto, revisão da avaliação da solução e da orientação do projeto e do investigador principal.

Os sujeitos que colaboraram na obtenção de dados de fadiga, utilizando o teste Go/NoGo disponibilizado no software criado.

6.4 Validação e avaliação

A validação da solução obtida é efetuada recorrendo a sua avaliação. Para tal recorre-se à QEF e a uma análise da taxa de acerto dos algoritmos de aprendizagem automática na previsão de fadiga.

6.4.1 QEF

A QEF é uma *framework* de avaliação quantitativa de software (Escudeiro and Bidarra, 2008), (Escudeiro, Bidarra and Escudeiro, 2010) que permite medir a qualidade de um software, decompondo-o nas suas características mesuráveis. Através do preenchimento de uma tabela de requisitos e cálculos auxiliares é possível obter um valor único em percentagem correspondente à qualidade do software. Podendo este método ser aplicado em todas as fases dos ciclos de desenvolvimento de software. Os requisitos são agrupados por dimensões e por fatores, possibilitando a obtenção de métricas intermedias automaticamente através do uso de uma folha de cálculo.

q	Desvio global	Qi	Dimensão	Qj	Wij (Peso do fator j na Dim i) [0,1]	Fator	rwjk (peso do requisito k no Fator j) {2, 4, 6, 8, 10}	Requisito	wfk % cumprimento do requisito k) [0,100]
95.82%	0.166667	100	Funcional	100	0.50	Casos de uso	8	RF1 - UC1 Gerir utilizador	100
							6	RF2 - UC2 Consultar histórico	100
							10	RF3 - UC3 Avaliar fadiga	100
				100	0.50	Interação	6	RS1 - O sistema deve suportar a língua Inglesa e Portuguesa.	100
							4	RU1 - Deve existir uma preocupação estética com o sistema percecionado pelos utilizadores.	100
							8	RU2 - O sistema deve ser simples e fácil de usar	100
		83.33333	Performance	75	0.67	Modelo	10	RC1 - O detetor deve apresentar uma taxa de acerto superior a 50%.	50
							10	RD1 - A deteção deve ser efetuada num intervalo de tempo inferior a um minuto.	100
				100	0.33	Sistema	6	RD2 - O sistema deve apresentar um consumo moderado de recursos de forma a reduzir o seu custo de alojamento.	100
		100	Adicional	100	0.50	Interface	10	RA1 - O sistema necessita ser disponibilizado online, devendo existir essa consideração a nível da implementação e	100
							8	RA2 - A interface entre os utilizadores e o sistema deve ser efetuada por https.	100
				100	0.50	Legal	8	RA3 - O sistema deve cumprir o princípio da minimização dos dados pessoais.	100
							6	RA4 - O sistema deve cumprir o RGPD tanto quanto possível.	100

Figura 27 – Visão da ferramenta de calculo QEF

Na Figura 27 é possível ver a folha de cálculo Excel implementada para esta validação de qualidade, sendo notório a distribuição de casos de uso por dimensão, fator e peso do fator. Uma das dimensões é a funcional que agrupa o fator referente aos casos de uso com um peso consoante a sua importância no sistema. Nesta dimensão existe também o fator de interação associado à interação do utilizador com o sistema. A segunda dimensão é a performance, um ponto fulcral do sistema, dividida no fator relacionado com o modelo de deteção e outro com a performance do sistema. A terceira dimensão é ocupada por requisitos adicionais, subdividida nos fatores de interface do sistema e dos requisitos legais relacionados com o RGPD.

As métricas de avaliação do QEF para o preenchimento do quadro são visíveis na Tabela 11 .

Tabela 11 - Métricas de avaliação do QEF

Descrição	Métrica	0 %	50 %	100 %
RF1 – UC1 Gerir utilizador	Uma pessoa pode criar e editar o seu utilizador. Permite fazer login e logout.	Sem implementação	Parcialmente implementado	Totalmente implementado
RF2 – UC2 Consultar histórico	Um utilizador consegue ver o histórico de testes de fadiga efetuados.	Sem implementação	Parcialmente implementado	Totalmente implementado
RF3 – UC3 Avaliar fadiga	O utilizador consegue efetuar teste Go/Nogo, indicar fadiga e receber previsão.	Sem implementação	Parcialmente implementado	Totalmente implementado
RS1 - O sistema deve suportar a língua Inglesa e Portuguesa.	Percentagem de frases não traduzidas em Português e Inglês no frontend.	10+	10 a 3	3 a 0
RU1 - Deve existir uma preocupação estética com o sistema percecionado pelos utilizadores.	Percentagem de resposta negativa de utilizadores a questionário com pergunta: “O sistema é feio?”.	< 33%	>= 33%	>= 50%
RU2 - O sistema deve ser simples e fácil de usar	Percentagem de resposta positiva de utilizadores a questionário com pergunta: “O sistema é de fácil utilização?”.	< 33%	>= 33%	>= 50%
RC1 - O detetor deve apresentar uma precisão superior a 50%.	O modelo de previsão deve ter uma taxa de acerto superior a 50% na classificação usando todas as classes da escala de fadiga. Ou superior a 50% com classes combinadas.	<50%	>=50% com classes combinadas	>= 50% com todas as classes
RD1 - A deteção deve ser efetuada num intervalo de tempo inferior a um minuto.	O tempo que um utilizador demora a fazer o teste de fadiga e a obter um resultado.	> que 2 minutos	<= a 2 minutos	<= a 1 minuto
RD2 - O sistema deve apresentar um consumo moderado de recursos.	O número de CPUs e memoria RAM que o servidor necessita.	8 CPUs ou 32 GB RAM	4 CPUs ou 16 GB RAM	2 CPUs e 8 GB RAM
RA1 - O sistema necessita ser disponibilizado online.	Um utilizador consegue aceder ao sistema via internet.	Não	-	Sim
RA2 - A interface entre os utilizadores e o sistema deve ser efetuada por HTTPS.	Percentagem das interfaces usadas pelo utilizador que suportam HTTPS.	0%	%	100%
RA3 - O sistema deve cumprir o princípio da minimização dos dados pessoais.	Existem dados pessoais, não absolutamente necessários ao funcionamento do sistema.	mais de 3 dados	1 a 3 dados	Não
RA4 - O sistema deve cumprir o RGPD tanto quanto possível.	Número de artigos do RGPD em falha pelo sistema.	mais de 10 falhas	3 a 10 falhas	1 a 2 falhas

Optou-se pela definição de um critério de três valores, 0, 50 e 100% que são suficientes para detalhar a qualidade de cada requisito sem introduzirem demasiada complexidade. Para os casos funcionais RF1, RF2 e RF3, a métrica utilizada é relacionada com a implementação total

ou parcial dos mesmos, ou com a sua não implementação. Visto todos os casos de uso terem sido totalmente implementados, estes três critérios obtêm um valor de 100%.

No RS1 relacionado com o suporte da língua portuguesa e inglesa, a métrica é relacionada com o número de frases sem tradução sendo esperado que frases simples como Go e NoGo não sejam traduzidas, razão pela qual o valor de 100% apresenta 3 frases por traduzir. Uma análise da aplicação revela que as únicas palavras ou frases por traduzir são de facto as expressões Go e NoGo, pelo que este critério obtém um valor de 100%.

Os RU1 e RU2 são medidos com base em questionários, tendo o cuidado de elaborar a pergunta binaria sobre a beleza na negativa, de forma a ser mais notório a percentagem de utilizadores com opinião desfavorável à estética do sistema. O resultado deste questionário anónimo efetuado a todos os voluntários que participaram e utilizaram a aplicação revela que a maioria considera que o sistema não é feio (87%) e que é de fácil utilização (100%). Assim ambos os critérios obtêm um valor de 100%.

Na avaliação da precisão do modelo RC1, a métrica é a taxa de acerto do mesmo, sendo dado a possibilidade de não ser possível adequadamente classificar todas as classes do questionário de fadiga. Caso seja necessário combinar classes de forma a melhorar a performance do sistema, o sistema de avaliação já contempla esta necessidade. De facto, não foi possível classificar com uma taxa de acerto superior a 50% na previsão de todas as classes, mas um agrupamento em três classes distintas permite atingir este valor. Assim o valor para este critério é de 50%.

O requisito RC1 sobre o tempo que demora a avaliar a fadiga, é medido simplesmente contabilizando o tempo que demora a correr o caso de uso na sua plenitude. Visto ser possível efetuar o teste e obter a previsão em menos de um minuto, o valor para este critério é de 100%.

No caso do RD2, a medição de recursos é efetuada através do número de processadores e memoria RAM que o servidor necessita. Possuindo o servidor dois processadores e apenas 4 GB de memoria RAM, o valor para o critério é de 100%.

O RA1, é medido binariamente caso o sistema esteja ou não disponível online. Visto o sistema estar disponível online, este critério obtém um valor de 100%.

No caso RA2, a métrica é a percentagem de interfaces existentes que o utilizador utiliza, que implementam o protocolo HTTPS. Existem três interfaces que o utilizador utiliza para interagir com o sistema implementado e todas elas implementam o protocolo HTTPS, assim o valor para este critério é de 100%.

Relativamente aos requisitos legais, RA3 e RA4, no primeiro é medido se existem dados pessoais desnecessários no sistema. No segundo o número de falhas que o sistema apresenta relativamente ao regulamento RGPD, sendo permitido 1 a 2 falhas. Visto o RA4 não ser um requisito central do sistema e os investigadores não serem peritos legais, é possível a

existência de algum detalhe legal despercebido na construção do mesmo. Uma análise à criação de um novo utilizador no sistema revela que os únicos dados pedidos são a data de nascimento, um nome de utilizador e o género. Estes três critérios não permitem identificar uma pessoa, tendo em conta que o nome de utilizador não é o nome da pessoa, assim não podem ser considerados dados pessoais. De qualquer forma, mesmo assim os requisitos do GRPD foram cumpridos, tendo em conta o pedido de autorização de tratamento de dados para fins científicos e as especificidades do artigo 31. Assim, apesar de o autor não ser um especialista legal, é possível afirmar que ambos os critérios obtêm um valor de 100%.

O resultado do QEF indica um valor de cerca de 95% para a qualidade da solução desenvolvida, sendo a principal falha a nível de performance, nomeadamente do modelo de classificação.

6.4.2 Considerações para a avaliação dos modelos

A avaliação do modelo a nível da sua precisão, conforme indicado no requisito RC1, é efetuada recorrendo à taxa de acerto (accuracy) que deve ser superior a 50%. Mas a avaliação de um único modelo não garante uma solução forte, pelo que é prática corrente o recursos a comparação de vários modelos com algoritmos de aprendizagem automática diferentes de forma a garantir uma elevada taxa de acerto. De igual forma, neste trabalho são criados e avaliados vários modelos de forma a selecionar o melhor.

Da literatura sobressaem vários algoritmos de aprendizagem automática usados na classificação de fadiga, o KNN, SVM e ANN. Assim serão usados pelo menos três algoritmos diferentes, o KNN, o SVM e um terceiro. Este terceiro poderá ser o ANN, mas devido a este ser uma rede neuronal, não é adequado o seu uso com poucos dados ou baixos recursos. Assim o ANN poderá ser substituído por outro algoritmo como o *Random Forest*, ou outro que se considere mais adequado.

A comparação entre os vários modelos criados deve ser efetuada recorrendo à métrica taxa de acerto e à comparação das matrizes de confusão. Caso apenas seja possível efetuar classificação binária, será vantajoso utilizar a métrica da área abaixo da curva (AUC) e característica de operação do recetor (ROC) como termo de comparação. Selecionando o modelo que apresente a melhor combinação destes dois ou três fatores como o modelo final.

6.5 Planeamento da recolha de dados

De forma a poder contruir o *dataset* que permite treinar o modelo de aprendizagem automática é necessário recolher dados do uso do sistema, nomeadamente referentes ao teste Go/NoGo e autoavaliação de fadiga. Esta recolha apesar de poder ser aleatória (teoricamente usando uma grande população de acordo com a lei dos grandes números), na prática não é realista esperar que exista um grande número de voluntários a participar nesta recolha de forma a garantir uma distribuição viável. Assim urge, tendo em conta os fatores ou

variáveis indicados pela literatura, identificar um grupo mínimo de pessoas capaz de providenciar dados que inspirem confiança. Apesar de esta recolha de dados não ser uma experiência, é possível aplicar parte do método de design de experiências 2K fatorial para identificar este grupo mínimo ideal. Os três fatores indicados pela literatura com influência nos resultados de testes Go/NoGo são a fadiga, a idade e o género. Tendo em conta que apesar da idade e o género serem atributos relativamente constantes de uma pessoa, a fadiga pode variar facilmente com o tempo. A Tabela 12 representa o design 2K fatorial para esta situação, com os fatores fadiga e idade a variar em dois possíveis valores, + para elevado e – para reduzido, e o género em masculino e feminino. Assim é possível afirmar que o número mínimo de pessoas necessárias para cobrir estes três fatores é de quatro pessoas.

Tabela 12 – Design 2K fatorial

Pessoa	Fadiga	Idade	Género
A	+	+	Masculino
A	-	+	Masculino
B	+	-	Masculino
B	-	-	Masculino
C	+	+	Feminino
C	-	+	Feminino
D	+	-	Feminino
D	-	-	Feminino

Um homem “A” com idade avançada, um jovem masculino “B” mais novo, uma mulher mais sénior “C”, e uma jovem rapariga “D”. Abstemo-nos de identificar os intervalos exatos de idade visto o importante é que exista uma separação clara de idades entre jovem e sénior, nos participantes da recolha de dados.

Relativamente à variação do fator fadiga, apesar de existirem experiências na literatura onde é pedido aos sujeitos que efetuem uma tarefa fatigante, esta não é garantia que o sujeito fique fatigado. Sendo a existência deste género de tarefa, um claro fator de desmotivação à participação voluntária. Assim o ideal será pedir aos participantes que efetuem o teste e autoavaliação durante a manhã após acordarem e outra vez à noite após o trabalho de forma a tentar capturar dois estados de fadiga, um fresco e outro fatigado. Idealmente, três testes em cada caso (manhã e noite) durante cerca de duas semanas.

É impossível dado a natureza voluntária deste trabalho escolher o grupo ideal de sujeitos, pelo que existe a necessidade de abordar várias pessoas de forma a tentar formar um grupo que englobe estas quatro pessoas ideais.

Os dados recolhidos necessitam de ser sujeitos a uma análise exploratória de dados, de forma a os caracterizar e compreender a correlação das diferentes variáveis no nível de fadiga.

6.6 Disseminação

A disseminação e a apresentação dos resultados deste trabalho ao publico em geral passa pela criação deste documento, a tese de mestrado. E pela sua apresentação e defesa junto a um painel de avaliação constituído por júris do Mestrado de Engenharia Informática do ISEP. Este trabalho será posteriormente disponibilizado publicamente, ficando acessível online no repositório científico do Instituto Politécnico do Porto.

7 Dataset e Modelos

Este capítulo descreve o processo de recolha de dados, a sua limpeza e transformação em *dataset*, e a exploração dos mesmos. Segue a descrição dos modelos obtidos e do processo de seleção do modelo mais adequado. É também avaliado o impacto da idade e género na performance do algoritmo, assim como o efeito da extensão da duração do teste Go/NoGo. No final é apresentada alguma informação sobre a implantação do modelo.

7.1 Descrição do processo de obtenção dos dados

De forma a obter os dados necessários à criação dos modelos, foram convidadas várias pessoas a participar como voluntários no uso do sistema. Tendo em conta os mínimos identificados no design 2K fatorial, foram convidadas oito pessoas. Quatro homens de cerca de 27, 43, 44 e 73 anos e quatro mulheres de 15, 32, 48 e 71 anos, garantindo pessoas jovens, de meia-idade e seniores. Foi pedido aos voluntários que durante duas semanas utilizassem o sistema para efetuar dois momentos de teste diários. Um de manhã, quando estivessem mais frescos, e outro ao final do dia, quando estivessem mais cansados. Em cada momento deveriam efetuar três testes seguidos (permitindo uma pequena visão da repetibilidade dos mesmos), totalizando 84 entradas por pessoa. Cada um destes sujeitos foi acompanhado no seu primeiro momento, com uma explicação de como fazer o teste, de como responder ao questionário de fadiga, e respondendo a todas as dúvidas levantadas. Todos os sujeitos foram treinados efetuando testes Go/NoGo sob supervisão, até apresentarem uma proficiência e autonomia mínima necessária à participação autónoma no processo de recolha de dados. A duração esperada de quinze dias foi largamente ultrapassada, atingindo um mês de duração. Consequência da existência de dias de indisponibilidade dos sujeitos, uma situação compreensível visto o aspeto voluntario da recolha.

7.2 Preparação dos dados

Os dados recolhidos foram analisados, revelando a existência de algumas incoerências. Várias entradas continham um número diferente que os três testes pedidos para o mesmo período do dia ou o seu nível de fadiga autoavaliado era diferente em um dos testes, alguns sujeitos apresentavam mais que os 84 testes requeridos, e um dos sujeitos pediu para algumas das entradas serem removidas. Uma breve conversa com este sujeito deixou a suspeição que os seus registos estavam a ser escolhidos. Procedeu-se à limpeza dos dados, removendo ou corrigindo as incorreções encontradas, incluindo a remoção de todas as entradas deste último sujeito de 44 anos. Dado a natureza do processo de recolha de dados não foram encontrados *outliers*.

Seguiu-se a criação de atributos auxiliares derivados da fadiga, através de uma generalização de valores categóricos. Tendo em contas os níveis de fadiga existentes na autoavaliação de fadiga do teste, visível na Figura 28, foram criados três novos atributos. conforme visível na Tabela 13.

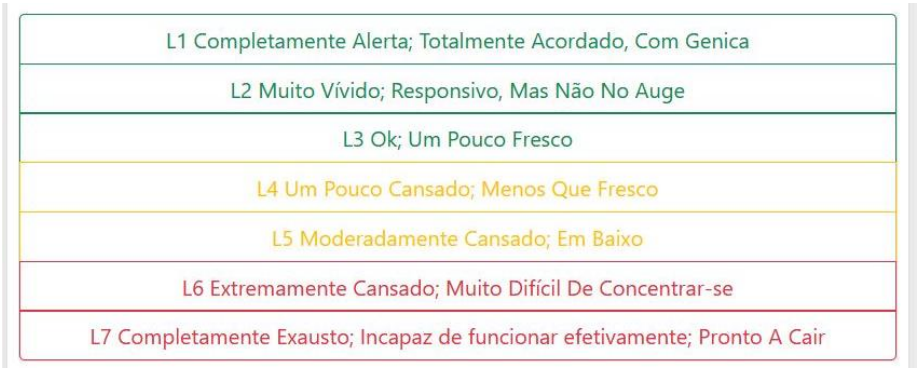

Figura 28 – Níveis de fadiga usados na autoavaliação

O atributo fadiga3 agrupou os níveis de fadiga em três categorias, redefinindo os níveis 2 e 3 como 1, o nível 5 como 4, e o nível 6 como 7. Obtendo-se um atributo capaz de discriminar em fresco, pouco cansado e muito cansado. O atributo fadiga2 fez a separação em fresco e cansado, agrupando o nível 2 e 3 ao nível 1, e os níveis 4, 5 e 6 ao 7. Foi criado outro atributo alternativo de duas categorias, subdividindo os dados em muito cansado e não muito cansado, o fadiga22 que agrupou o nível 6 ao 7 e juntou todos os outros ao nível 3.

Tabela 13 – Atributos auxiliares relativos a fadiga

Atributo	**1**	**2**	***3***	***4***	**5**	**6**	**7**
fadiga original	1	2	3	4	5	6	7
fadiga3	1	1	1	4	4	7	7
fadiga2	1	1	1	7	7	7	7
fadiga22 (alternativo)	3	3	3	3	3	7	7

De forma a preparar os dados a serem alimentados aos modelos e garantir uma melhor performance os dados foram uniformizados para a escala de 0 a 1, sendo por exemplo a idade dividida por 100 (idade máxima expectável) e o género masculino substituído por 0 e o feminino por 1. Assim foram gerados dois *datasets*, um com os dados originais para registo, e outro com os dados após a mudança de escala para alimentar aos modelos.

Visto existirem apenas sete idades diferentes nos dados, o que representa menos de 10% das idades do ser humano, foi criado um novo atributo para idade baseado em quatro classes: menos de vinte anos, menos de quarenta anos, menos de sessenta anos e mais de sessenta anos. Este atributo permite transformar qualquer idade, evitando previsões com base em valores para os quais o modelo não foi treinado.

7.3 Análise exploratória

Seguiu-se uma análise exploratória dos dados, através de ferramentas gráficas e estatísticas do software R, sendo visível abaixo as descobertas de maior relevância.

Uma das representações que melhor ajuda a visualização dos dados é a frequência de falhas NoGo por cada nível de fadiga. Na Figura 29, é possível ver essa informação para todos os níveis de fadiga, sendo notório o incremento do número de falhas com o cansaço e uma falta de dados em ambos os extremos dos níveis de fadiga. A distribuição do número de testes pelas classes é de 3%, 26%, 27%, 16%, 16%, 12% e 4% respetivamente.

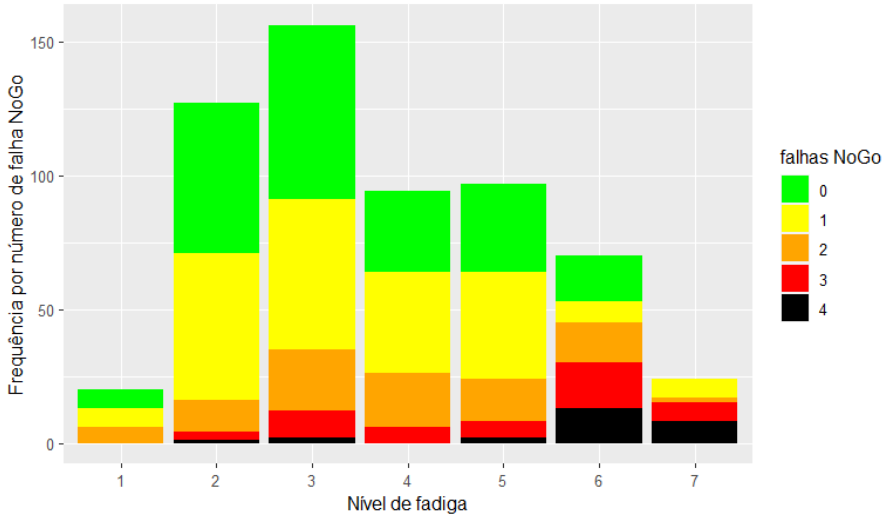

Figura 29 - Frequência de falhas NoGo para todos os níveis de fadiga

Outra das representações de interesse é a distribuição do tempo de reação de resposta ao estímulo Go, representado através do diagrama de extremos e quartis ou *boxplot*, visível na Figura 30. Apesar de não ser claro, existe uma ligeira tendência da diminuição do tempo de reação com a fadiga.

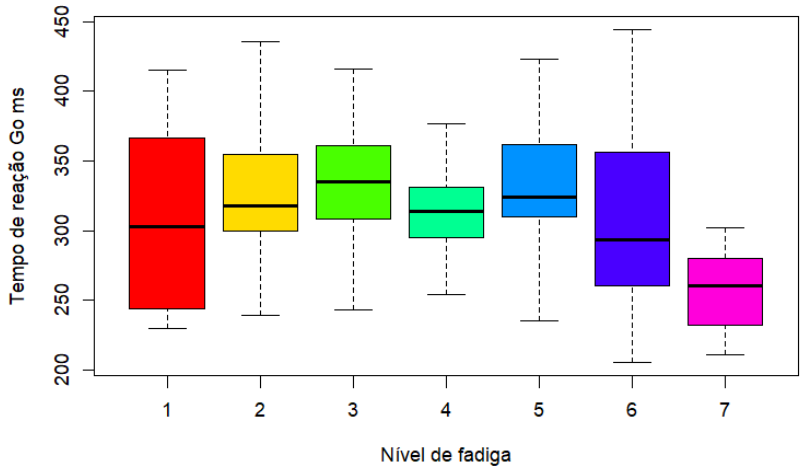

Figura 30 – Diagrama de extremos e quartis do tempo de reação para todos os níveis

A divisão dos dados de fadiga em três classes recorrendo ao atributo fadiga3, que representa os estados de fadiga fresco, pouco cansado e muito cansado, permite obter uma visão mais consolidada dos dados conforme visível na Figura 31. Agora é claramente notório uma concentração do número mais elevado de falhas na classe de maior fadiga, a 7 que inclui os níveis 6 e 7. Nota-se que as outras duas classes aparentam ser bastante semelhantes.

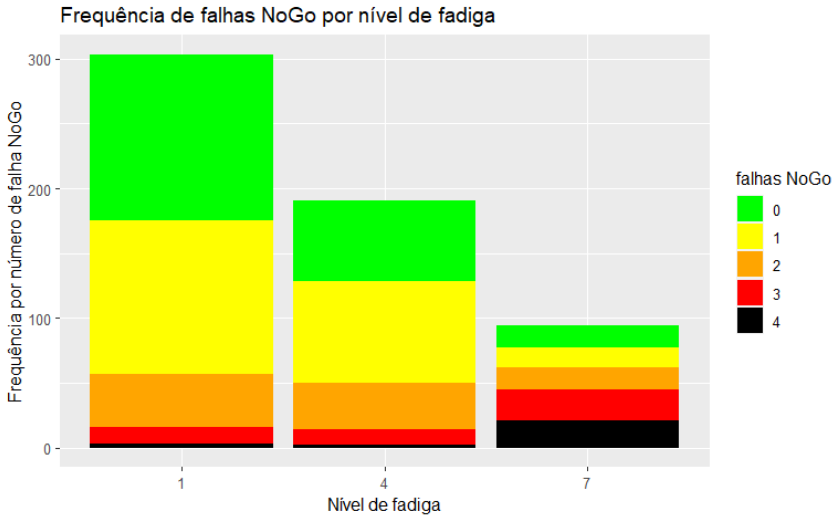

Figura 31 - Frequência de falhas NoGo para 3 classes de fadiga

A distribuição do número de testes por cada uma das classes é de 52%, 32% e 16% respetivamente. Lembrando que a recolha de dados foi efetuada ao início e ao fim do dia, é natural que metade das entradas representem uma ausência de fadiga. A visão da distribuição dos tempos de reação, visível na Figura 32, mostra que existe realmente uma tendência da diminuição do tempo de reação com o aumento de fadiga. De novo nota-se que as duas classes que não apresentam fadiga extrema são bastante semelhantes.

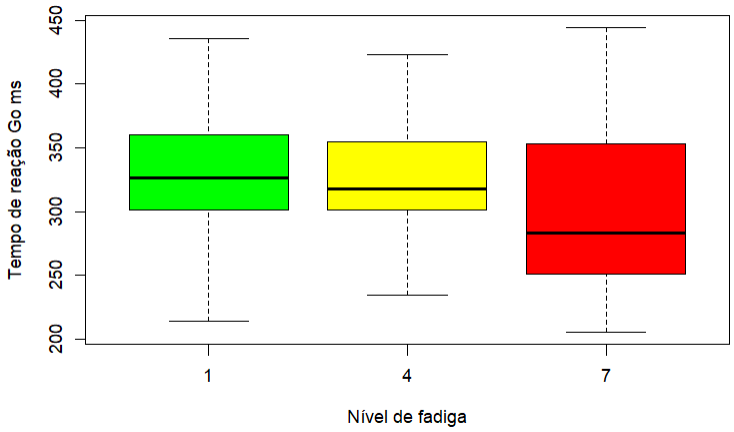

Figura 32 - Diagrama de extremos e quartis do tempo de reação para três classes

De facto, recorrendo a um teste de hipóteses, uma ferramenta estatística, é possível afirmar a um nível de confiança de 95% que o tempo de reação da classe 7 é inferior ao das classes 1 e 4. Não sendo possível distinguir estatisticamente entre a classe 1 e 4.

A investigação da influência de cada atributo com a fadiga, passa pela avaliação da correlação dos atributos existentes. A Figura 33 representa essas mesmas correlações, expostas de uma forma gráfica, correspondendo o valor zero a mais baixa correlação entre dois atributos, o valor 1 à mais alta correlação direta e o -1 a mais alta correlação inversa. Os atributos fadiga seguidos de um número referem-se a fadiga, sendo "fadiga 7" o original, "fadiga 3" o com três classes, o "fadiga 2" com duas classes, e o "fadiga 22a" o alternativo com duas classes que separa extremamente fatigado de não extremamente fatigado.

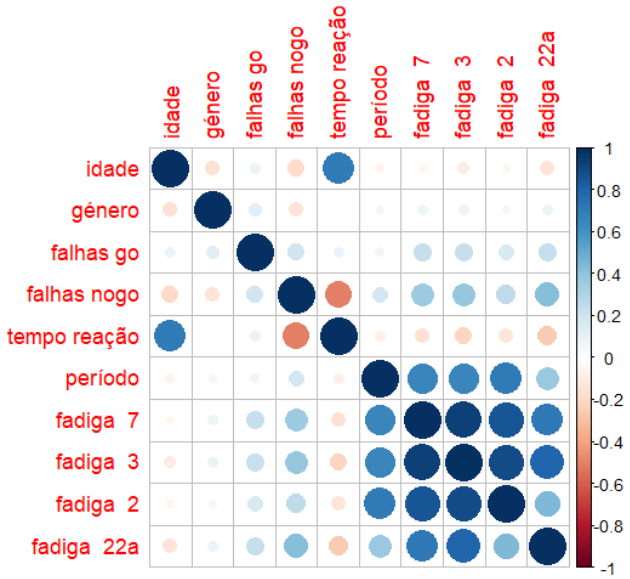

Figura 33 – Matriz de correlação entre os vários atributos

São compreensíveis várias correlações como a idade com o tempo de reação com um valor de 0.71, as falhas NoGo com o tempo de reação com valor de -0.50, a falta de correlação entre o género e o tempo de reação com valor de -0.01. As correlações entre o tempo de reação e a fadiga 7,3, 2 e 22a são baixas sendo respetivamente -0.16, -0.22, -0.14 e -0.25. As correlações entre as falhas Go e a fadiga também são também baixos, correspondendo aos valores 0.24, 0.23, 0.16 e 0.24 respetivamente. Mas as correlações da fadiga com as falhas NoGo apresentam valores ligeiramente mais elevados, sendo respetivamente 0.37, 0.38, 0.26 e 0.42. Com esta informação é possível indicar que o principal indicador de fadiga será o número de falhas do estímulo NoGo, seguido das falhas do estímulo Go e por fim o tempo de reação com uma menor importância. Também é possível afirmar que o género não influencia o tempo de reação de forma significativa, e que é visível uma relação entre tempos de reação mais baixos e mais falhas no estímulo NoGo.

7.4 Modelos

Os algoritmos de aprendizagem automática selecionados para avaliação foram o KNN, SVM e ANN. Devido ao facto de o processo de treino do ANN ser demorado, foi incorporado o algoritmo *Random Forest* no script automático de avaliação desenvolvido na linguagem R. O treino dos modelos foi efetuado com uma divisão dos dados em 70% para treino e 30% para teste, e recorrendo à estratégia *repeated cross validation* com 10 *folds*. Algumas das classes encontravam-se desequilibradas, especialmente no caso de apenas duas classes, pelo foi aplicada a técnica de *oversampling* aos dados de treino obtendo um equilíbrio. Os parâmetros dos algoritmos foram variados de forma a selecionar os que originavam modelos com a melhor taxa de acerto. Os resultados recolhidos são visíveis na Tabela 14, onde para cada combinação de algoritmo e número de classes de fadiga a classificar são apresentadas várias métricas, sendo a mais relevante a taxa de acerto (accuracy).

Os modelos para sete classes foram os que apresentaram os piores resultados, não conseguindo prever todas as classes em alguns dos casos, o que impede a obtenção de algumas métricas. Apesar de apenas conseguirem acertar em cerca de um terço dos casos, este resultado é superior ao resultado teórico de uma previsão aleatória (1/7). O melhor modelo para três classes atinge uma taxa de acerto de 0.57, superior ao objetivo, e ao esperado de um modelo aleatório (0.33). A melhor taxa de acerto para duas classes (fresco vs fatigado) apresenta um valor de 0.62, pouco animador face à probabilidade de acertar em metade dos casos com um modelo puramente aleatório. Os modelos com duas classes alternativas (não muito cansado vs muito cansado) apresentam resultados animadores, com uma taxa de acerto de 0.84. Observando os resultados de cada modelo é claro que o algoritmo *Random Forest* é o superior, obtendo em todas os casos e métricas sempre resultados superiores.

Tabela 14 – Comparação dos modelos

Algoritmo	**Classes**	**Taxa de acerto**	***Recall***	***Precision***	**F1-Score**	**Kappa**	**AUC**
Knn	7	0.33	0.33	0.32	0.31	0.16	-
Svm	7	0.29	0.29	-	-	0.06	-
Random forest	7	0.36	0.36	0.38	0.36	0.20	-
ANN	7	0.34	0.34	-	-	0.15	-
Knn	3	0.52	0.56	0.50	0.53	0.25	-
Svm	3	0.51	0.55	0.49	0.52	0.24	-
Random Forest	3	0.57	0.59	0.56	0.57	0.31	-
ANN	3	0.46	0.51	0.44	0.47	0.17	-
Knn	2	0.57	0.74	0.56	0.64	0.13	0.62
Svm	2	0.57	0.65	0.57	0.61	0.13	0.59
Random Forest	2	0.62	0.72	0.61	0.66	0.24	0.64
ANN	2	0.56	0.66	0.56	0.61	0.12	0.62
Knn	2a	0.77	0.77	0.94	0.85	0.38	0.83
Svm	2a	0.81	0.81	0.95	0.88	0.46	0.85
Random Forest	2a	0.84	0.86	0.94	0.90	0.51	0.88
ANN	2a	0.80	0.81	0.94	0.87	0.43	0.87

7.4.1 Seleção do modelo

O modelo selecionado para classificação é o *Random Forest* de três classes, visto ser o modelo com mais classes que ultrapassa o objetivo mínimo de taxa de acerto. Para três classes é o que apresenta a maior taxa de acerto 0.57 e uma comparação da sua matriz de confusão, representada na Tabela 15, com as restantes não evidencia anomalias classificando todas as classes.

Tabela 15 – Matriz de confusão do modelo de três classes

		Real		
	classe	**1**	**4**	**7**
Previsto	**1**	55	27	4
	4	23	25	3
	7	12	5	21

Dado os resultados dos modelos de classificação de duas classes alternativas serem bastante bons, com o *Random Forest* a atingir um AUC de 0.88 e uma taxa de acerto de 0.84 na identificação de sujeitos muito cansados, este modelo foi também escolhido para implementação como modelo alternativo. Uma comparação da sua matriz de confusão, visível na Tabela 16, com as dos outros algoritmos não revelou qualquer anomalia, classificando todas as classes.

Tabela 16 - Matriz de confusão do modelo de duas classes alternativo

		Real	
	classe	**1**	**7**
Previsto	**1**	128	7
	7	20	21

7.4.2 Impacto da idade e género

Da informação da matriz de correlação dos atributos correspondentes às variáveis do teste Go/NoGo sabemos que a influência da idade e género na fadiga é extremamente baixa, mas que a idade está correlacionada com o tempo de reação e que este se encontra inversamente correlacionado com a fadiga. Assim foi testado o impacto destas variáveis na taxa de acerto do algoritmo *Random Forest* de três classes. O modelo original apresenta uma taxa de acerto de 0.57, com a remoção da idade ele baixa para 0.54, com a remoção do género para 0.56 e com a remoção de ambos para 0.50. Mas note-se que se o modelo for treinado com as idades reais em vez das quatro classes de idade, a remoção do género implica uma redução de 0.57 para 0.51. Não sendo claro qual das variáveis possui um menor impacto real, apenas que a remoção de qualquer uma das variáveis origina uma degradação da capacidade de classificação do modelo.

7.4.3 Avaliação da extensão da duração do teste

Tendo em conta o facto de os testes Go/NoGo terem sido efetuados em grupos de três testes consecutivos, foi possível converter esses três testes em uma única entrada do *dataset*, reduzindo o seu tamanho em um terço. Esta entrada representa um teste com o triplo da duração, com a possibilidade de 12 falhas NoGo em vez das 4 do teste real. Uma análise de correlação a estes dados revela que a correlação das falhas NoGo com a fadiga aumenta de 0.38 para 0.45 no caso de três classes. Uma breve tentativa de comparação dos modelos de classificação revela ser possível obter com o algoritmo SVM uma taxa de acerto de 0.65 para estas três classes, existindo uma degradação da performance do algoritmo *Random Forest*. Visto estes dados representarem uma alteração grosseira do teste real efetuado, representando um teste mais longo, esta informação não foi usada para classificação real.

7.4.4 Detalhes da implantação do modelo

Os dois modelos de fadiga foram implementados em Python com ajuda da biblioteca "scikit-learn" de forma a estarem disponíveis ao servidor. O modelo do algoritmo *Random Forest* com três classes continuou a apresentar uma taxa de acerto de 57% (F1-Score de 54% e Kappa de 29%). O alternativo de duas classes conseguiu aumentar a sua taxa de acerto de 84% para um valor de 89% (F1-Score de 79% e Kappa de 59%). A razão desse aumento deve-se a uma otimização da semente de aleatoriedade visível a vermelho no Código 2, responsável pela aleatoriedade da divisão do conjunto de dados de teste e treino.

```
# Split dataset into training set and test set
X2_train, X2_test, y2_train, y2_test = train_test_split(X2, y2,
test_size=0.3, random_state=656) # 70% training and 30% test

# Reagrupar as colunas de treino.
df2 = pd.DataFrame(X2_train)   #nova dataframe df2
df2.insert( 5 , "fadiga22", y2_train, True)  #re-inserir a coluna

df2_majority = df2[df2.fadiga22==3] # Dividir por classes.
df2_minority = df2[df2.fadiga22==7]

df2_minority_upsampled = resample(df2_minority, replace=True, n_samples=347,
random_state=123)

df2_upsampled = pd.concat([df2_majority, df2_minority_upsampled])

X2_train=df2_upsampled[['age', 'gender', 'gofailnumber',
'nogofailnumber','goavg'  ]]  # Features
y2_train=df2_upsampled['fadiga22'] # Labels

clf_2a=RandomForestClassifier(n_estimators=500, random_state=123)
clf_2a.fit(X2_train,y2_train)
```

Código 2 – Extrato de código da criação do modelo alternativo em Python

Este extrato de código representa a separação dos dados em teste e treino, o *oversampling* da classe minoritária e o treino do modelo. Apesar de existir um aumento do da taxa de acerto, esta é conseguida à custa de uma manipulação da separação teste/treino podendo significar um sobre ajuste do modelo.

7.5 Breve sumário

Este capítulo descreve o processo de recolha de dados referentes ao teste Go/NoGo efetuado por quatro mulheres e quatro homens, a sua limpeza, tratamento, análise e criação de modelos. Uma análise exploratória dos dados indica uma redução do tempo de reação para fadiga elevada, e que as falhas na tarefa NoGo são o fator de maior correlação com a fadiga. Dos modelos de avaliação de fadiga construídos, destaca-se os que recorrem ao algoritmo *Random Forest* pela sua maior taxa de acerto, nomeadamente na classificação de fadiga em três e duas classes. O modelo escolhido para previsão é o de três classes (fresco, pouco cansado e muito cansado) que classifica com uma taxa de acerto de 57%. Um modelo alternativo de duas classes (não muito cansado e muito cansado) foi também selecionado devido a sua elevada capacidade de previsão com uma taxa de acerto de 84%.

8 Conclusões

Este capítulo apresenta as principais conclusões do trabalho, iniciando com uma discussão de alguns dos aspetos mais relevantes. Prossegue com uma análise do caminho a tomar no futuro, e termina com as principais conclusões do trabalho, aproveitando para rever os objetivos atingidos.

8.1 Discussão

A solução proposta apresenta um sistema complexo que permite a deteção de fadiga, devido à existência de quatro grandes módulos de software independentes, o que origina uma certa complexidade. Apesar do sistema ser modular, e o componente do modelo de fadiga ser facilmente substituível, uma alteração do módulo de gestão de utilizadores implica possíveis alterações nos componentes de *frontend* e *backend* a nível do formato da informação do utilizador. A escolha do uso do *software* Keycloak como sistema de gestão de utilizadores apresenta várias vantagens a nível de segurança, de garantias do RGPD e de tempo de desenvolvimento. Mas a sua substituição por um módulo incorporado no *backend* e uma única base de dados (fadiga e utilizadores) poderia simplificar o sistema de forma significativa, especialmente agora que sabemos que é possível evitar o tratamento de dados pessoais.

A solução final encontra-se implementada num único servidor Linux, por motivos de performance e económicos. Uma alternativa seria a sua implementação através de um sistema de virtualização, como por exemplo o Docker, e recorrendo a bases de dados externas também virtualizadas. Isto, apesar de permitir vantagens a nível da facilidade de implantação e escalabilidade do sistema, origina custos económicos e de desenvolvimento maiores. Seria necessária uma máquina de desenvolvimento mais potente de forma a permitir a virtualização local durante o desenvolvimento. Relativamente a custos, o servidor único custa atualmente 33€ por mês na Google Cloud, mas a base de dados SQL mais simples no mesmo fornecedor custa entre 10 e 50€ mensais e o alojamento de um único componente sob a forma de Kubernetes custa a partir de 25€ mensais.

O processo de recolha de dados foi efetuado com sucesso obtendo-se um total de 588 entradas válidas de sete sujeitos voluntários. As condições mínimas indicadas pelo design 2K fatorial foram excedidas, existindo três faixas etárias diferentes. Infelizmente os registos de um dos oito participantes originais foram excluídos, um sacrifício necessário ao rigor dos dados. A interação com alguns dos sujeitos revelou a subjetividade da escala de fadiga, com muitos dos sujeitos a afirmarem que tinham dificuldade em diferenciar entre o nível de fadiga que sentiam. Apesar de sentirem que conseguiam avaliar a sua fadiga, estes sentiam alguma dificuldade em a diferenciar e a associar a um nível específico da escala. Este facto é reforçado pela existência de entradas com níveis diferentes de fadiga em testes separados por apenas alguns segundos. Uma das dificuldades apresentadas por alguns dos sujeitos foi o facto de terem de ligar os seus computadores pela manhã para efetuar o teste, lamentando a impossibilidade de efetuarem o teste no seu telemóvel.

A análise exploratória demonstra que a divisão da recolha num período de menor fadiga ao início do dia e outro de maior fadiga ao fim da noite foi um sucesso, visto metade dos dados representarem sujeitos claramente frescos. A baixa correlação do tempo de reação com a fadiga foi uma surpresa, dado a informação disponível na literatura, ficando claro que o fator de maior correlação foi o número de falhas NoGo. Mas mais importante que esta baixa correlação é o facto de o tempo de reação claramente diminuir com a fadiga, o que contraria a literatura encontrada. Uma possível explicação para esta situação é que as pessoas quando estão cansadas podem possuir menos paciência ao realizar o teste, originando uma dificuldade em inibir as reações originando tempos mais rápidos e um maior número de falhas. Esta informação leva-nos a repensar os parâmetros do teste, visto ser evidente que deve ser dada maior importância à tarefa NoGo. A alteração da razão entre as duas tarefas não será o ideal pois a tarefa NoGo deve ser rara o suficiente para induzir em erro pessoas cansadas, mas o aumento do número total de tarefas pode melhorar os resultados. Um aumento do número de taregas NoGo permite uma maior discriminação da fadiga levando a uma melhor sensibilidade, conforme sugerido pelo modelo criado através da junção de três testes em um único. Mas deve existir um cuidado neste aumento, pois testes longos criam insatisfação no utilizador e podem inviabilizar o uso da ferramenta no mundo real.

Da avaliação dos modelos é clara a inadequação do método à classificação em sete classes de fadiga (35% de acerto), sendo a previsão de três classes (fresco, pouco cansado e muito cansado) o mínimo aceitável (57% de acerto). Da análise da previsão de duas classes é possível afirmar que existe alguma dificuldade em diferenciar entre não fatigado e fatigado (61% de acerto) comparativamente à deteção de níveis elevados de fadiga (84% de acerto). O algoritmo de classificação com melhores resultados foi o *Random Forest*, mas ficou a impressão (gerada durante várias análises parciais durante o processo de recolha de dados) que esta liderança deste algoritmo apenas é aplicada a este conjunto de dados em particular, podendo outro ser o ideal para um conjunto de dados diferentes. Apesar dos resultados da precisão dos modelos serem inferiores aos que constam na literatura (cerca de 90% de taxa de acerto) para outros métodos de deteção, não devemos culpar a escala de fadiga escolhida de sete classes ou a subjetividade da mesma, mas aceitar que o teste Go/NoGo como ferramenta poderá apresentar uma dificuldade em diferenciar níveis baixos de fadiga.

Claramente existem aspetos a melhorar, tanto a nível da recolha de dados como dos parâmetros do teste. Por exemplo, dado o desequilíbrio do número de entradas com fadiga elevada, o uso de uma tarefa geradora de fadiga ou outro mecanismo capaz de garantir fadiga elevada, aquando da recolha de dados, pode gerar um conjunto de dados que origine uma melhor performance dos modelos.

Com base nos dados recolhidos não é possível efetuar com exatidão uma avaliação real do impacto da idade e do género na previsão com base em Go/NoGo. Dependendo do modelo, a remoção do género pode ter um impacto maior que a remoção da idade ou vice-versa. Apenas que, para o conjunto de dados recolhidos, a remoção de ambos os atributos implica uma redução da taxa de acerto em cerca de 7%. Sendo impossível afirmar se essa redução da performance é resultado de um sobre ajuste do modelo aos dados devido a um reduzido número de participantes, ou em alternativa, resultado de um impacto real do género e idade no teste Go/NoGo. Mas tendo em conta os dados referentes a correlações e do baixo impacto do género no modelo de previsão com idade sob a forma de classes, é possível que o género seja o atributo com menor influência e assim um candidato a remoção.

O uso da linguagem Python para a implantação dos modelos, relativamente à linguagem R utilizada para avaliação dos diferentes modelos, originou uma pequena diferença a nível da taxa de acerto do modelo de previsão de duas classes alternativas (não muito cansado e muito cansado). O uso de R para a análise inicial, permitiu uma rápida visualização de resultados, uma das suas forças indicadas na literatura, sendo a escolha do uso de Python para a criação do servidor extremamente adequada dada a facilidade de implementação do mesmo. O aumento da taxa de acerto do modelo de duas classes alternativos de 84% (R) para 89% (Python) levanta alguma suspeita, mas é razoável dado a natureza dos modelos de classificação. Algoritmos como o *Random Forest* são gerados aleatoriamente, sendo a sua consistência garantida pelo uso de uma semente de aleatoriedade constante. Dado que a implementação do algoritmo é feita em duas linguagens distintas, é natural que possam originar resultados ligeiramente diferentes. Note-se também, que o incremento da taxa de acerto foi atingido após a otimização dos parâmetros do modelo, através de uma otimização da semente de aleatoriedade associada à divisão dos dados em teste e treino. Indicando a possibilidade que que a obtenção de um maior volume de dados pode levar uma melhor classificação.

8.2 Trabalho futuro

No decorrer deste trabalho foram descobertas e apontadas várias limitações dos métodos utilizados e da solução criada, destacando-se:

- Uma menor performance dos modelos face a outros métodos de deteção de fadiga existentes na literatura
- Uma baixa correlação dos tempos de reação com a fadiga

- Um desagrado dos sujeitos voluntários em o detetor não ter sido disponibilizado em versão *smartphone*.

- Uma incapacidade do método em classificar adequadamente os sete níveis de fadiga originais

- O facto de teste apenas possuir quatro tarefas NoGo, o fator mais correlacionado com a fadiga

- O conjunto de dados apresentar um desequilíbrio a nível do número de entradas com fadiga elevada, sendo a melhor performance para a diferenciação deste tipo de fadiga

- A implantação da solução ser ligeiramente complexa

Tendo em conta estas observações, existem vários caminhos para o desenvolvimento deste tema no futuro.

O primeiro é a aquisição de um novo conjunto de dados, agora com um teste Go/NoGo mais longo, de forma a incluir mais tarefas NoGo com o intuito de aumentar a capacidade de diferenciação dos modelos, visto o NoGo ser o fator mais correlacionado com a fadiga. Tendo em consideração que um aumento da duração do teste pode originar insatisfação e a sua inviabilidade no mundo real. Deve também ser explorada a hipótese de incluir nesta nova recolha de dados uma tarefa originadora de fadiga, com o intuito de diminuir o desequilíbrio do número de dados de fadiga elevada, alavancando a capacidade de classificação deste tipo de fadiga.

Apesar de a maioria dos sujeitos indicarem que o sistema é de fácil utilização, estes sugeriram a sua disponibilização via *smartphone*. A razão pela qual não se considerou inicialmente o seu uso deve-se a latência associada com a recolha do tempo de reação aos estímulos. Sabendo agora que a correlação do tempo de reação com a fadiga é baixa, sendo o principal fator a tarefa NoGo, a exploração do uso deste meio torna-se viável. Com o benefício adicional, de que o uso de um objeto pessoal e individual como um telemóvel permite a eliminação dos dados de utilizador e a manutenção de um histórico apenas localmente.

Dada a relativa complexidade do sistema criado, a sua simplificação deverá ser um dos próximos passos. Atualmente, sacrificando o histórico, seria possível reduzir o sistema apenas ao componente de frontend e ao componente do modelo de fadiga. Criando um sistema anónimo (sem utilizadores) para previsão de fadiga online. Com a possível evolução para telemóvel, este será sem dúvida o caminho a seguir.

8.3 Conclusões finais

Tendo em conta a pergunta ou hipótese inicial, é possível responder que sim. Que é possível criar um detetor de fadiga recorrendo a testes Go/NoGo e à aprendizagem automática.

Apesar da performance dos modelos obtidos na solução serem inferiores aos descritos na literatura, que na sua generalidade apresentam taxas de acerto de 90%, os modelos gerados possuem a capacidade de prever a fadiga acima de uma mera previsão aleatória.

Com a solução e a colaboração de sete sujeitos voluntários foi possível criar um *dataset* com o resultado de 588 testes Go/NoGo e uma autoavaliação de fadiga. Uma análise dos dados revela que estes se dividem quase igualmente por frescos e fatigados. Que existe uma correlação entre a fadiga e os erros da tarefa NoGo de cerca de 0.4, entre a fadiga e a tarefa Go de cerca de 0.24. Sendo a correlação entre a fadiga e o tempo de reação inferior a cerca de 0.20. Assim o principal fator correlacionável com a fadiga é a tarefa NoGo. Também é possível afirmar com um nível de confiança de 95% que o tempo de reação ao teste Go é inferior para níveis de fadiga muito elevados comparativamente aos outros níveis.

Uma análise dos modelos, revela que o algoritmo de classificação mais indicado para os dado recolhidos é o *Random Forest* apresentando uma taxa de acerto de 36% para a classificação de sete classes, de 57% para a classificação de três classes (fresco, pouco cansado e muito cansado), de 62% para duas classes (fresco e cansado) e de 84% para duas classes alternativas (não muito cansado e muito cansado). O modelo escolhido para previsão foi o de três classes, tendo também sido implementado o modelo de deteção de elevada fadiga como alternativa. Demonstrando claramente a elevada afinidade do teste Go/NoGo para detetar níveis de fadiga elevados.

Revendo os objetivos iniciais propostos, foi possível criar um detetor de fadiga com base em testes Go/NoGo, apesar da sua capacidade de classificação ficar aquém de outros métodos referenciados na literatura. O detetor foi disponibilizado online, tendo todo o processo de recolha de dados sido efetuado pela internet. A deteção demora um intervalo de tempo claramente inferior a um minuto, demorando tipicamente perto de trinta segundos. O sistema é de fácil utilização, tendo 100% dos utilizadores afirmado através de um questionário que este é de fácil utilização. Comparativamente apenas 87% afirmaram que não era feio. A precisão do detetor é superior a 50%, atingindo uma taxa de acerto de 57% para três classes de fadiga (fresco, pouco cansado e muito cansado). Note-se que uma avaliação do software criado indica um índice de qualidade de cerca de 95%, sendo a principal falha o nível de performance do modelo.

Referências

Abiodun, O. I. *et al.* (2018) 'State-of-the-art in artificial neural network applications: A survey', *Heliyon*, 4(11), p. e00938. doi: 10.1016/j.heliyon.2018.e00938.

Åkerstedt, T. and Gillberg, M. (1990) 'Subjective and Objective Sleepiness in the Active Individual', *International Journal of Neuroscience*, 52(1–2), pp. 29–37. doi: 10.3109/00207459008994241.

Al-Libawy, H. *et al.* (2017) 'Fatigue Detection Method Based on Smartphone Text Entry Performance Metrics', *Proceedings - 2016 9th International Conference on Developments in eSystems Engineering, DeSE 2016*, pp. 40–44. doi: 10.1109/DeSE.2016.9.

Assembleia da República (2019) *Lei n.º 58/2019 de 8 de agosto da Assembleia da República*. Available at: https://dre.pt/dre/detalhe/lei/58-2019-123815982.

Breque, M., De Nul, L. and Petridis, A. (2021) *Industry 5.0 Towards a sustainable, humancentric and resilient European industry - Directorate-General for Research and Innovation*, *European Commission*. doi: 10.2777/308407.

Caldwell, J. A. *et al.* (2019) 'Fatigue and its management in the workplace', *Neuroscience and Biobehavioral Reviews*, 96(July 2018), pp. 272–289. doi: 10.1016/j.neubiorev.2018.10.024.

Carneiro, D. *et al.* (2017) 'A multi-modal architecture for non-intrusive analysis of performance in the workplace', *Neurocomputing*, 231(May 2016), pp. 41–46. doi: 10.1016/j.neucom.2016.05.105.

Cervantes, J. *et al.* (2020) 'A comprehensive survey on support vector machine classification: Applications, challenges and trends', *Neurocomputing*, 408, pp. 189–215. doi: 10.1016/j.neucom.2019.10.118.

Chandiwala, J. and Agarwal, S. (2021) 'Driver's real-time Drowsiness Detection using Adaptable Eye Aspect Ratio and Smart Alarm System', *2021 7th International Conference on Advanced Computing and Communication Systems, ICACCS 2021*, pp. 1350–1355. doi: 10.1109/ICACCS51430.2021.9441756.

Chapman, P. *et al.* (2000) 'Crisp-Dm', *SPSS inc*, 78, pp. 1–78. Available at: http://www.crisp-dm.org/CRISPWP-0800.pdf.

Escudeiro, P. and Bidarra, J. (2008) 'Quantitative Evaluation Framework', *RISTI – Revista Ibérica de STI*, pp. 16–27.

Escudeiro, P., Bidarra, J. and Escudeiro, N. (2010) 'Evaluating Educational Software', *Journal of Systemics, Cybernetics and Informatics*, 8(2), pp. 5–10.

Géron, A. (2019) *Hands-on Machine Learning with Scikit-Learn, Keras, and Tensorflow*. 2nd edn. O'Reilly Media.

Girish, I. *et al.* (2020) 'Driver Fatigue Detection', *2020 IEEE 17th India Council International Conference, INDICON 2020*. doi: 10.1109/INDICON49873.2020.9342456.

Grandini, M., Bagli, E. and Visani, G. (2020) 'Metrics for Multi-Class Classification: an Overview', pp. 1–17. Available at: http://arxiv.org/abs/2008.05756.

Guo, M. *et al.* (2016) 'Research on the relationship between reaction ability and mental state for online

assessment of driving fatigue', *International Journal of Environmental Research and Public Health*, 13(12). doi: 10.3390/ijerph13121174.

Guo, Z. *et al.* (2016) 'The impairing effect of mental fatigue on visual sustained attention under monotonous multi-object visual attention task in long durations: An event-related potential based study', *PLoS ONE*, 11(9), pp. 1–14. doi: 10.1371/journal.pone.0163360.

Guo, Z. *et al.* (2018) 'The impairing effects of mental fatigue on response inhibition: An ERP study', *PLoS ONE*, 13(6), pp. 1–18. doi: 10.1371/journal.pone.0198206.

IBM (2021) *Python vs. R: What's the Difference?* Available at: https://www.ibm.com/cloud/blog/python-vs-r (Accessed: 7 February 2022).

Jiang, T., Gradus, J. L. and Rosellini, A. J. (2020) 'Supervised Machine Learning: A Brief Primer', *Behavior Therapy*, 51(5), pp. 675–687. doi: 10.1016/j.beth.2020.05.002.

Kato, Y., Endo, H. and Kizuka, T. (2009) 'Mental fatigue and impaired response processes: Event-related brain potentials in a Go/NoGo task', *International Journal of Psychophysiology*, 72(2), pp. 204–211. doi: 10.1016/j.ijpsycho.2008.12.008.

Keycloak (2022) *Keycloak*. Available at: https://www.keycloak.org/ (Accessed: 1 February 2022).

Koen, P. *et al.* (2001) 'Providing clarity and a common language to the "fuzzy front end"', *Research Technology Management*, 44(2), pp. 46–55. doi: 10.1080/08956308.2001.11671418.

Kolodziej, M. *et al.* (2020) 'Fatigue Detection Caused by Office Work with the Use of EOG Signal', *IEEE Sensors Journal*, 20(24), pp. 15213–15223. doi: 10.1109/JSEN.2020.3012404.

Kruchten, P. B. (1995) 'The 4+1 View Model of architecture', *IEEE Software*, 12(6), pp. 42–50. doi: 10.1109/52.469759.

Kumar, V. (2019) *Python Vs R: What's Best for Machine Learning*. Available at: https://towardsdatascience.com/python-vs-r-whats-best-for-machine-learning-93432084b480 (Accessed: 7 February 2022).

Kuratsune, D. *et al.* (2012) 'Changes in reaction time, coefficient of variance of reaction time, and autonomic nerve function in the mental fatigue state caused by long-term computerized Kraepelin test workload in healthy volunteers', *World Journal of Neuroscience*, 02(02), pp. 113–118. doi: 10.4236/wjns.2012.22016.

Laouz, H., Ayad, S. and Terrissa, L. S. (2020) 'Literature Review on Driver's Drowsiness and Fatigue Detection', *2020 International Conference on Intelligent Systems and Computer Vision, ISCV 2020*. doi: 10.1109/ISCV49265.2020.9204306.

Lee, K. A., Hicks, G. and Nino-Murcia, G. (1991) 'Validity and reliability of a scale to assess fatigue', *Psychiatry Research*, 36(3), pp. 291–298. doi: 10.1016/0165-1781(91)90027-M.

Li, W., Chen, Y. and Song, Y. (2020) 'Boosted K-nearest neighbor classifiers based on fuzzy granules', *Knowledge-Based Systems*, 195, p. 105606. doi: 10.1016/j.knosys.2020.105606.

Magnuson, J. R., Doesburg, S. M. and McNeil, C. J. (2021) 'Development and recovery time of mental fatigue and its impact on motor function', *Biological Psychology*, 161(February), p. 108076. doi: 10.1016/j.biopsycho.2021.108076.

Martinez-Plumed, F. *et al.* (2021) 'CRISP-DM Twenty Years Later: From Data Mining Processes to Data Science Trajectories', *IEEE Transactions on Knowledge and Data Engineering*, 33(8), pp. 3048–3061. doi: 10.1109/TKDE.2019.2962680.

Matthews, G. *et al.* (2012) *The handbook of operator fatigue*. ASHGATE. doi: 10.1080/00140139.2013.819674.

Natnithikarat, S. *et al.* (2019) 'Drowsiness Detection for Office-based Workload with Mouse and Keyboard Data', *BMEiCON 2019 - 12th Biomedical Engineering International Conference*, 1975. doi: 10.1109/BMEiCON47515.2019.8990236.

Osisanwo, F. Y. *et al.* (2017) 'Supervised Machine Learning Algorithms: Classification and Comparison', *International Journal of Computer Trends and Technology*, 48(3), pp. 128–138. doi: 10.14445/22312803/ijctt-v48p126.

Osterwalder, A. *et al.* (2014) *Value proposition design: How to create products and services customers want*. John Wiley & Sons.

Parekh, V., Shah, D. and Shah, M. (2020) 'Fatigue Detection Using Artificial Intelligence Framework', *Augmented Human Research*, 5(1), pp. 1–17. doi: 10.1007/s41133-019-0023-4.

Patel, H. H. and Prajapati, P. (2018) 'Study and Analysis of Decision Tree Based Classification Algorithms', *International Journal of Computer Sciences and Engineering*, 6(10), pp. 74–78. doi: 10.26438/ijcse/v6i10.7478.

Phaladisailoed, T. and Numnonda, T. (2018) 'Machine Learning Models Comparison for Bitcoin Price Prediction', in *2018 10th International Conference on Information Technology and Electrical Engineering (ICITEE)*. IEEE, pp. 506–511. doi: 10.1109/ICITEED.2018.8534911.

Pimenta, A. *et al.* (2016) 'A neural network to classify fatigue from human-computer interaction', *Neurocomputing*, 172, pp. 413–426. doi: 10.1016/j.neucom.2015.03.105.

Qin, H. *et al.* (2021) 'Detection of mental fatigue state using heart rate variability and eye metrics during simulated flight', *Human Factors and Ergonomics In Manufacturing*, 31(6), pp. 637–651. doi: 10.1002/hfm.20927.

Ribeiro, A. P. (2018) *Detecting and monitoring mental fatigue: a non-invasive approach*. PhD thesis, Universidade do Minho. Available at: http://repositorium.sdum.uminho.pt/handle/1822/55851.

Rich, N. and Holweg, M. (2000) *Value Analysis Value Engineering*. Cardiff, United Kingdom. Available at: https://www.urenio.org/tools/en/value_analysis.pdf.

Saaty, T. L. (2004) 'Decision making — the Analytic Hierarchy and Network Processes (AHP/ANP)', *Journal of Systems Science and Systems Engineering*, 13(1), pp. 1–35. doi: 10.1007/s11518-006-0151-5.

Sakovich, N. (2021) *Top Most Popular Frontend Frameworks 2022*. Available at: https://www.sam-solutions.com/blog/best-frontend-framework/ (Accessed: 7 February 2022).

Samn, S. and Perelli, L. (1982) *Estimating Aircrew Fatigue: A Technique with Application to Airlift Operations*. USAF school of aerospace medicine. Available at: http://oai.dtic.mil/oai/oai?verb=getRecord&metadataPrefix=html&identifier=ADA125319.

Schröer, C., Kruse, F. and Gómez, J. M. (2021) 'A Systematic Literature Review on Applying CRISP-DM Process Model', *Procedia Computer Science*, 181, pp. 526–534. doi: 10.1016/j.procs.2021.01.199.

Sennovate (2022) *Best Open Source Single Sign-On Solutions*. Available at: https://sennovate.com/best-open-source-single-sign-on-solutions/ (Accessed: 1 February 2022).

Shahid, A. *et al.* (2011) 'Karolinska Sleepiness Scale (KSS)', in *STOP, THAT and One Hundred Other Sleep Scales*. Springer Link, pp. 209–210. doi: 10.1007/978-1-4419-9893-4_47.

Shen, Weijie *et al.* (2012) 'Effective driver fatigue Monitoring through pupil detection and yawing analysis in low light level environments', *International Journal of Digital Content Technology and its Applications*, 6(17), pp. 372–383. doi: 10.4156/jdcta.vol6.issue17.41.

Stancin, I. and Jovic, A. (2019) 'An overview and comparison of free Python libraries for data mining and big data analysis', in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, pp. 977–982. doi: 10.23919/MIPRO.2019.8757088.

Staron, M. (2020) 'Action Research as Research Methodology in Software Engineering', in *Action Research in Software Engineering*. Cham: Springer International Publishing, pp. 15–36. doi: 10.1007/978-3-030-32610-4_2.

Sur, S. and Sinha, V. (2009) 'Event-related potential: An overview', *Industrial Psychiatry Journal*, 18(1), p. 70. doi: 10.4103/0972-6748.57865.

Technostacks (2018) *React vs Angular: Which Is A Better JavaScript Framework?* Available at: https://technostacks.com/blog/react-vs-angular (Accessed: 7 February 2022).

Technostacks (2021) *Top Frontend Frameworks of 2022 for Web Development*. Available at: https://technostacks.com/blog/best-frontend-frameworks/ (Accessed: 7 February 2022).

The R Foundation (2021) *What is R?* Available at: https://www.r-project.org/about.html (Accessed: 7 February 2022).

Ulinskas, M. *et al.* (2018) 'Recognition of human daytime fatigue using keystroke data', *Procedia Computer Science*, 130, pp. 947–952. doi: 10.1016/j.procs.2018.04.094.

Woods, D. L. *et al.* (2015) 'Factors influencing the latency of simple reaction time', *Frontiers in Human Neuroscience*, 9(MAR), pp. 1–12. doi: 10.3389/fnhum.2015.00131.

Zhang, X. *et al.* (2021) 'Fatigue Detection with Covariance Manifolds of Electroencephalography in Transportation Industry', *IEEE Transactions on Industrial Informatics*, 17(5), pp. 3497–3507. doi: 10.1109/TII.2020.3020694.