FUTURE STEM HUB

# Materiais Educativos com Atividades Práticas em Python e Scratch

## FUTURE-STEM-HUB

**Capacitar o Ensino STEM no Ensino Secundário através de Formação em Inteligência Artificial e Recursos para Alunos e Educadores**

**N.º do Projeto**
**2024-1-DE03-KA220-SCH-000247346**

Co-funded by the European Union

## MATERIAIS EDUCATIVOS COM ATIVIDADES PRÁTICAS EM PYTHON E SCRATCH

O projeto "Empowering Secondary School STEM Education with AI Training and Resources for Students and Educators / FUTURE-STEM-HUB" (ref. n.º: 2024-1-DE03-KA220-SCH-000247346) é cofinanciado pelo Programa Erasmus+ da União Europeia. É coordenado pela Universidade de Duisburg-Essen (Alemanha) e conta com a participação de quatro organizações parceiras: M&M Profuture Training (Espanha), Direção Provincial do Ministério da Educação de Kütahya (Türkiye), COOPETAPE – Cooperativa de Ensino, CRL – entidade instituidora da ETAP Escola Profissional (Portugal), e Tetra Solutions Ltd. (Bulgária).

Os membros da equipa do projeto, representantes de todas as organizações parceiras, desenvolveram os Materiais Educativos FUTURE-STEM-HUB com Atividades Práticas em Python e Scratch, sendo que cada parceiro assumiu a autoria de um módulo específico. Estes materiais têm como objetivo introduzir os conceitos fundamentais da Inteligência Artificial a alunos do ensino secundário, promovendo simultaneamente a sensibilização e a reflexão crítica sobre as suas implicações sociais e éticas.

**Autores:**
Mustafa Bilgin, University of Duisburg-Essen (Alemanha)
Monica Moreno, M&M Profuture Training (Espanha)
Montserrat Renedo, M&M Profuture Training (Espanha)
João Barroso, ETAP School (Portugal)
Angelina Presa, ETAP School (Portugal)
Silviya Georgieva, Tetra Solutions Ltd. (Bulgária)
Borislava Zaharieva-Tomova, Tetra Solutions Ltd. (Bulgária)
Yeliz Yurter, Kütahya MEM (Turquia)
Özcan Turan, Kütahya MEM (Turquia)

**Editor:**
Mustafa Bilgin, University of Duisburg-Essen (Alemanha)

# Lista de Abreviaturas:

**IA:** Inteligência Artificial

**ML:** Machine Learning (Aprendizagem Automática)

**DL:** Deep Learning (Aprendizagem Profunda)

**NN:** Neural Network (Rede Neural)

**NLP:** Natural Language Processing (Processamento de Linguagem Natural)

**CV:** Computer Vision (Visão Computacional)

**CNN:** Convolutional Neural Network (Rede Neural Convolucional)

**RGB:** Red, Green, Blue (Modelo de Cor)

**YOLO:** You Only Look Once (modelo de deteção de objetos em tempo real)

**FPS:** Frames Per Second (Fotogramas por Segundo)

**TTS:** Text-to-Speech (Síntese de Voz)

**MNIST:** Modified National Institute of Standards and Technology (Conjunto de Dados)

**IDE:** Integrated Development Environment (Ambiente de Desenvolvimento Integrado)

**ML4K:** Machine Learning for Kids

**IBM:** International Business Machines

**MIT:** Massachusetts Institute of Technology

**EACEA:** Agência de Execução Europeia da Educação e da Cultura

**UNESCO:** Organização das Nações Unidas para a Educação, Ciência e Cultura

**STEM:** Science, Technology, Engineering and Mathematics (Ciência, Tecnologia, Engenharia e Matemática)

# Índice





Empowering Secondary School STEM Education with AI
Training and Resources for Students and Educators
2024-1-DE03-KA220-SCH-000247346

# FUTURE-STEM-HUB

# Visão Geral do Projeto

FUTURE-STEM-HUB tem como objetivo promover e facilitar a integração de temas de Inteligência Artificial (IA) no ensino STEM ao nível do ensino secundário, através de: 1) Disponibilização de materiais educativos que introduzem os conceitos de IA e as suas implicações sociais; 2) Oferta de recursos de aprendizagem prática para que os alunos explorem a IA através da programação em Python; e 3) Capacitação dos professores com apoio para integrar a IA na formação STEM do ensino secundário.

# Resultados do Projeto

1. **Curso 1: Introdução Digital: Fundamentos da Inteligência Artificial** (Introdução à IA através de Materiais Educativos Interativos para Alunos do Ensino Secundário)

2. **Curso 2: Aprofundar a IA com Python e Scratch** (IA Avançada: Materiais de Aprendizagem Prática para Alunos do Ensino Secundário)

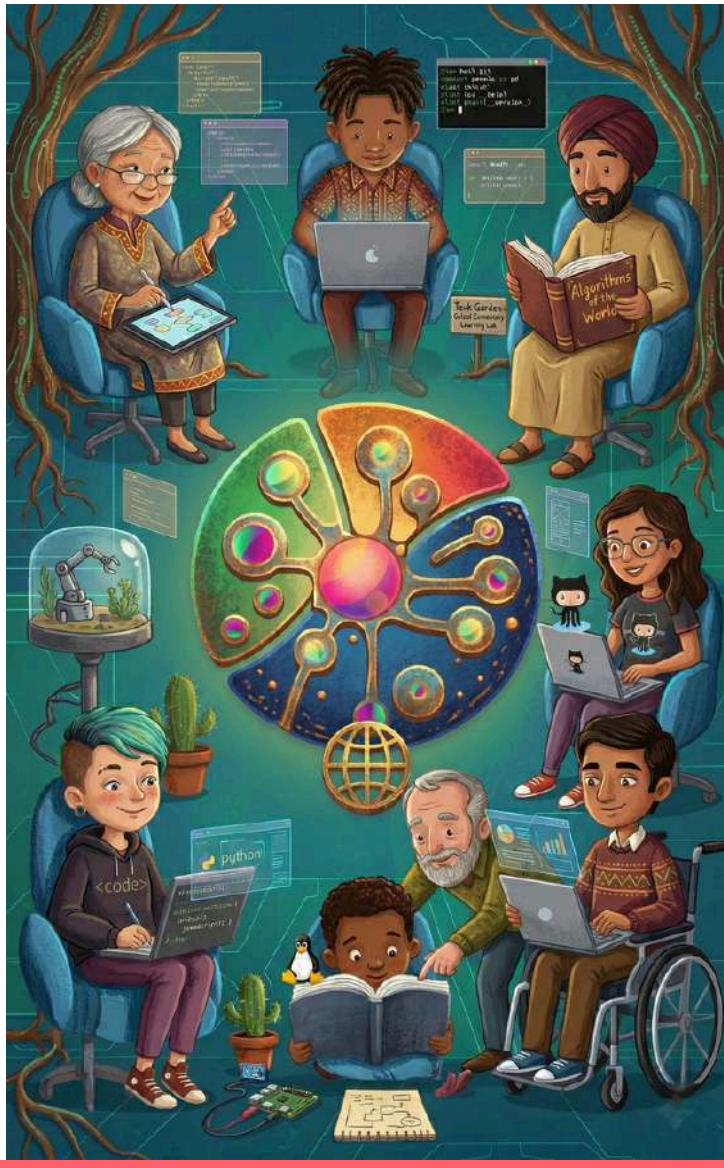
3. **E-Toolkit para Educadores Escolares: Reforço das Competências em Inteligência Artificial** (Guia Metodológico de IA para Professores do Ensino Secundário)

# Introdução

Bem-vindo(a) aos Materiais Educativos com Atividades Práticas em Python e Scratch — um percurso de aprendizagem envolvente e interativo concebido para introduzir os alunos do ensino secundário ao fascinante e prático universo da Inteligência Artificial. O objetivo é dotar professores e alunos do ensino secundário de materiais acessíveis e orientados para a prática, que permitam explorar os principais conceitos de IA através de Python e Scratch. Adaptados a alunos com idades entre os 15 e os 18 anos, com diferentes níveis de proficiência em STEM, estes materiais são ideais para aprendizagem autónoma e assíncrona, podendo igualmente ser integrados de forma eficaz no ensino em sala de aula com orientação do professor.

O conteúdo encontra-se organizado em cinco módulos abrangentes, centrados no desenvolvimento de competências práticas de programação em Python e Scratch, na lógica matemática através da aprendizagem baseada em jogos (Escape Room), em aplicações criativas de IA, na resolução de problemas do mundo real com dados e em considerações éticas.

Cada módulo combina conteúdos teóricos, questionários de consolidação, utilização de plataformas externas para exploração adicional e exercícios práticos que asseguram uma compreensão sólida dos conceitos e metodologias de IA. A duração estimada de aprendizagem para os cinco módulos é de aproximadamente 10 horas.

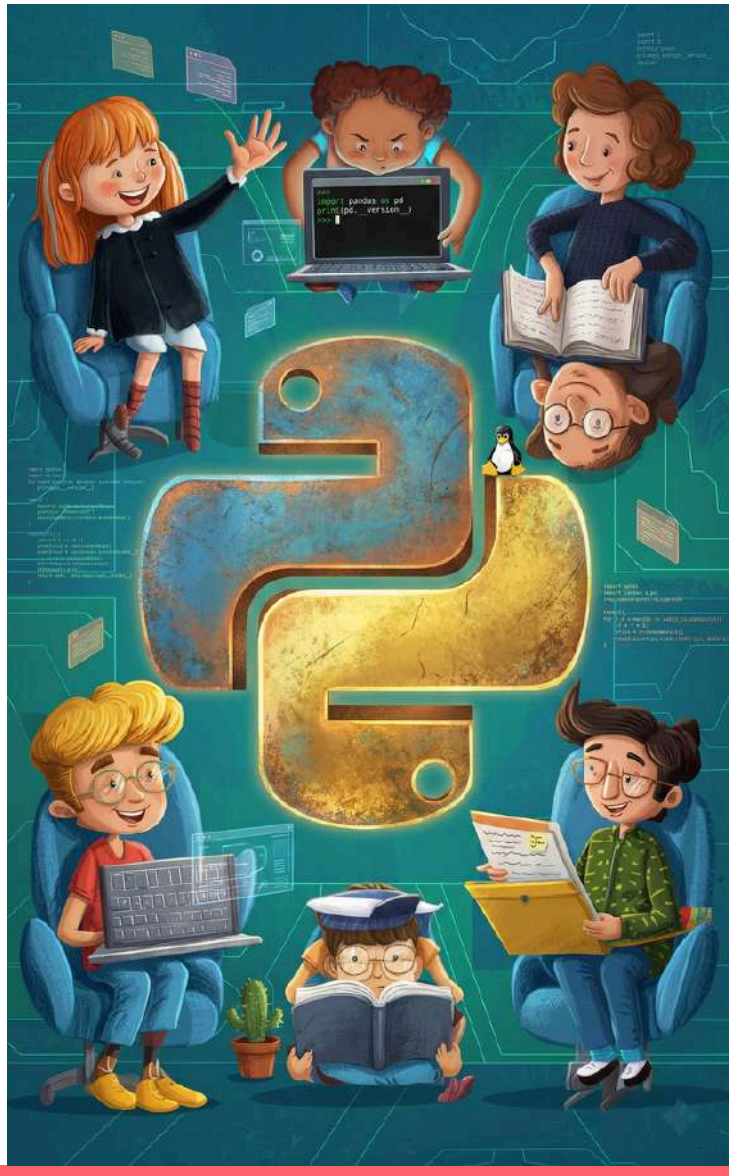


# FUTURE-STEM-HUB

Após a conclusão do curso, os alunos serão capazes de compreender e definir os principais conceitos de IA, tais como aprendizagem automática, redes neurais e aprendizagem profunda. Adquirirão também conhecimentos mais aprofundados sobre a história e evolução da Inteligência Artificial e as suas aplicações em diversos setores, compreendendo de que forma as novas tecnologias de IA moldam as tendências atuais e o mundo contemporâneo. Os alunos aprofundarão ainda o estudo de tecnologias fundamentais, incluindo visão computacional e processamento de linguagem natural. Por fim, explorarão e desenvolverão uma compreensão mais sólida das questões éticas associadas à IA, da sua responsabilidade e do seu impacto na sociedade.

Por outro lado, os professores terão acesso a recursos inovadores e interativos que poderão utilizar e aplicar facilmente na sala de aula STEM, apoiando o seu currículo e a sua prática pedagógica. Estes materiais contribuirão para a criação de ambientes de aprendizagem mais interessantes e envolventes, oferecendo diferentes abordagens para envolver ativamente os alunos do ensino secundário no processo de aprendizagem.

Estes materiais serão posteriormente convertidos num curso online, que será disponibilizado na plataforma FUTURE-STEM-HUB e acessível através do website do projeto: www.future-stem-hub.



**Após a conclusão bem-sucedida do curso, os alunos receberão um certificado digital que comprova a sua participação e aproveitamento.**

# Módulo 1: Mini-Desafios de Programação: Aprendizagem Automática e Redes Neurais

## INTRODUÇÃO

Já alguma vez se perguntou como é que o seu smartphone sabe quais os vídeos de que poderá gostar? Ou como é que uma aplicação consegue reconhecer a sua caligrafia? Por detrás de tudo isto está a Inteligência Artificial (IA) — que é muito menos misteriosa do que muitas pessoas pensam! Neste capítulo, irá acompanhar Aylin e o seu inteligente smartwatch, Alara. Juntos, embarcarão em aventuras que revelam como a IA realmente funciona. Desenvolverá uma compreensão clara dos princípios e componentes fundamentais das Redes Neurais (NN) e da Aprendizagem Automática (ML), incluindo a forma como processam entradas, aprendem a partir de dados e tomam decisões. A melhor parte? Não precisa de ser especialista em programação! Cada aventura inclui pequenos desafios de codificação em três níveis de dificuldade. Basta escolher o que melhor se adequa a si — do nível iniciante ao verdadeiro desafio. Verá que, logo após o primeiro capítulo, já compreenderá como as máquinas "aprendem a pensar". Vamos mergulhar juntos no mundo da IA.

No final do módulo, será capaz de desenvolver diferentes competências, tais como:

- **Introdução à IA:** O Presente Misterioso: Um Relógio que Faz Mais do que Marcar as Horas
- **Redes Neurais — Noções Básicas:** Alara Desperta
- **Árvores de Decisão:** O Detetive do Caminho para a Escola — Como o Seu Relógio Pensa
- **Classificação Baseada em Regras:** Como o Spotify Pode Saber o Que Quer Ouvir
- **Aprendizagem por Reforço:** Aprender Através de Recompensas
- **Redes Neurais Convolucionais:** Como a IA Deteta Contornos em Imagens

- **Processamento de Linguagem Natural:** Como os Computadores Compreendem a Linguagem
- **Sistemas de Recomendação:** Como o YouTube Sabe o Que Quer Ver
- **Floresta Aleatória (Random Forest):** Previsão Meteorológica
- **Ética na IA:** Visão para o Futuro — O Que Podemos e Devemos Fazer com a IA

| | |
|---|---|
| Duração do Módulo | 2 horas (1 hora de aprendizagem + 1 hora de exercícios práticos) |

## MATERIAIS EDUCATIVOS

**AYLIN** tem 17 anos, é curiosa e adora descobrir como as coisas funcionam. Nunca desiste!

**LENA** é a melhor amiga de Aylin. É criativa e adora música. A sua família é da Polónia; às vezes traz deliciosos pierogi!

**ALARA** não é um smartwatch comum. É uma IA que ensina Aylin sobre Inteligência Artificial.

Figura 1 Personagens educativos Aylin, Alara e Lena; Conceito e design: Mustafa Bilgin; Aperfeiçoado com recurso ao Google Gemini.

## UNIDADE 1.1 CONCEITO E ABORDAGEM DO CURSO

O processo de aprendizagem é orientado por uma abordagem baseada em narrativas, que estabelece a ligação entre os conteúdos STEM e contextos da vida real com os quais os alunos se podem identificar. Conceitos complexos, como redes neurais ou aprendizagem por reforço, são apresentados de forma simplificada e em modelo prototípico, permitindo aos alunos desenvolver uma compreensão sólida dos fundamentos — sem necessidade de experiência prévia em programação. Este material enquadra-se plenamente no ensino de informática e da educação tecnológica, em projetos STEM, clubes extracurriculares ou contextos de aprendizagem interdisciplinar. Para além de promover a compreensão técnica, incentiva igualmente — através do capítulo dedicado à Ética na IA — a reflexão crítica sobre questões morais e sociais.

## UNIDADE 1.2 INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL (IA)

Antes de iniciar as atividades, é fundamental compreender o conceito central em análise. A Inteligência Artificial (IA) refere-se à simulação da inteligência humana em máquinas programadas para pensar como os seres humanos e imitar as suas ações. O termo pode igualmente ser aplicado a qualquer máquina que apresente características associadas à mente humana, como a aprendizagem e a resolução de problemas.

Neste módulo, exploraremos os elementos fundamentais da IA através de histórias interativas:

- **Redes Neurais:** São sistemas computacionais inspirados nas redes neurais biológicas que constituem o cérebro dos animais. Permitem que a IA "aprenda" a partir de exemplos.
- **Árvores de Decisão:** São ferramentas de apoio à tomada de decisão que utilizam um modelo em forma de árvore para representar decisões e as suas possíveis consequências. Expressam a lógica do tipo "Se isto acontecer, então faz aquilo."
- **Aprendizagem Automática (ML):** Subárea da IA que permite aos sistemas aprender automaticamente e melhorar o seu desempenho com base na experiência, sem necessidade de programação explícita.

Este módulo utiliza uma abordagem narrativa (*storytelling*) para tornar estes conceitos matemáticos complexos acessíveis e de fácil compreensão.

## UNIDADE 1.3 CONFIGURAÇÃO DO SISTEMA E PRÉ-REQUISITOS

### Que Plataforma É Necessária

rá utilizar o **Google Colab** para executar o código em Python neste módulo. O Google Colab é uma plataforma online gratuita que permite escrever e executar código diretamente no navegador — não é necessária qualquer instalação.

**Aceda aqui:**
https://colab.research.google.com/

### O que Está Incluído no Processo de Registo?

1. Aceda ao Google Colab.
2. Clique em "Iniciar sessão" no canto superior direito.
3. Introduza o seu endereço de email Google e a palavra-passe.
4. Após iniciar sessão, poderá abrir ou criar um novo notebook.

### É Necessário Registo?

Sim, é necessária uma **conta Google gratuita** (conta Gmail) para utilizar o Google Colab. Se já tiver uma conta, pode iniciar sessão diretamente. Caso não tenha uma conta Google, deverá criar uma antes de começar.

**Para criar uma conta Google:**
Aceda a accounts.google.com
e siga os passos de registo.

### Onde Deve Escrever o Código

Todos os exercícios de programação do Módulo 1 estão preparados num **notebook do Google Colab**. Não precisa de escrever código de raiz — utilizará um notebook previamente criado que contém todos os exercícios e instruções.

Aqui está o notebook do Módulo 1:
>*Notebook Colab do Módulo 1*<

### O que Fazer com o Notebook:

1. Abra a ligação acima.
2. Clique em "Copiar para o Drive" para guardar a sua própria versão editável.
3. Siga as instruções dentro do notebook, executando cada célula de código passo a passo.

## UNIDADE 1.4 INSTRUÇÕES ADICIONAIS ANTES DE COMEÇAR

1. Utilize um navegador web moderno, como Chrome, Firefox, Edge ou Safari.
2. Certifique-se de que dispõe de uma ligação à Internet estável — o Google Colab funciona online.
3. Leia atentamente a história e as instruções de cada estação antes de iniciar a tarefa de programação.
4. Cada estação disponibiliza **três níveis de dificuldade**:
   - **Básico** para iniciantes.
   - **Avançado** se já tiver alguma experiência com Python.
   - **Expert:** se procura desafios mais exigentes, este é o nível indicado.
   - Escolha o nível que melhor se adequa às suas competências.
5. Execute as células de código pela ordem apresentada. Alguns exercícios dependem dos anteriores.
6. Se tiver dificuldades, utilize a opção "**Dica**" e o "**Excerto de Solução**" disponibilizados em cada exercício.
7. Após concluir uma tarefa, experimente modificar o código para observar como isso influencia o resultado. A experimentação ajuda a consolidar a aprendizagem.

# REFERÊNCIAS


### Atribuição de Emojis e Ícones

- Este documento utiliza emojis padrão Unicode, conforme definidos pelo Unicode Consortium. As versões específicas dos emojis podem variar consoante o sistema operativo do utilizador. Unicode Consortium. (2022). Consultado em https://unicode.org/emoji/charts/full-emoji-list.html. Os ícones são provenientes de Flaticon.com | High Quality Icons.

### Livros e Artigos

- Russell, S., & Norvig, P. (2021). Artificial Intelligence: A Modern Approach (4.ª ed.). Pearson. (Manual abrangente sobre conceitos de IA, incluindo redes neurais, árvores de decisão e aprendizagem por reforço.)

- Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3.ª ed.). O'Reilly. (Introdução prática à Aprendizagem Automática e às redes neurais com Python.)

### Ética na IA e Sociedade

- Comissão Europeia. (2019). Ethics Guidelines for Trustworthy AI. (Quadro orientador para o desenvolvimento e implementação ética da IA.)

- UNESCO. (2021). Recommendation on the Ethics of Artificial Intelligence. (Diretrizes globais sobre ética na IA, com foco nos direitos humanos e na inclusão.)

### Programação em Python para Iniciantes

- Sweigart, A. (2019). Automate the Boring Stuff with Python (2.ª ed.). No Starch Press. (Livro introdutório de Python, acessível a iniciantes, com projetos práticos.)


## LIGAÇÕES PARA PLATAFORMAS DE APRENDIZAGEM

- Aqui está o notebook do Módulo 1 – https://colab.research.google.com/drive/14VaJnlgo5habZ0-N9jpz9nWZSVAw_Xwa?usp=sharing

- Documentação Oficial de Python – https://docs.python.org/3/ Referência oficial e tutoriais da linguagem Python.)

## EXERCÍCIO PRÁTICO

### Estação 1: O Presente Misterioso — Um Relógio que Faz Mais do que Marcar as Horas

lÉ o 17.º aniversário de Aylin — entre baklava e balões encontra-se um pequeno pacote misterioso. Aylin hesita antes de o abrir.

**AYLIN:** *"O que é isto, Tia Sema?"*

**TIA SEMA:** *"Algo especial, canım (querida). A tua avó diria: este relógio tem alma."*

Aylin desembrulha cuidadosamente o pacote. No interior encontra-se um elegante smartwatch com uma pulseira azul-turquesa. De repente, o ecrã ilumina-se.

**ALARA (o relógio):** *"Merhaba (Olá), Aylin! Eu sou a Alara — a tua nova amiga IA!"*

Aylin recua, surpreendida.

**AYLIN:** *"O... o relógio está a falar?"*

**TIA SEMA:** *"lEle fala todas as línguas que compreendes.*
*A Alara vai mostrar-te como a Inteligência Artificial realmente funciona!"*

**ALARA:** *"Imagina que sou como o teu cérebro: tenho neurónios que aprendem e sinapses que se conectam. Juntas, vamos descobrir como as máquinas aprendem a pensar!"*

**BÁSICO**

**CÓDIGO:**

```
NOME = INPUT("QUAL É O TEU NOME? ")
PRINT("OLÁ " + NOME + "! EU SOU A ALARA, A TUA AMIGA IA!")
```

**Dica:** Utilize a variável que introduzin no espaço em branco.

**Excerto da solução:** nome

## Estação 1: O Presente Misterioso — Um Relógio que Faz Mais do que Marcar as Horas

**AVANÇADO**

**CÓDIGO:**

```
print("🎂 Planeamento do Aniversário da Aylin 🎂")
comida_favorita = input("Qual é a tua comida favorita?
(Pizza/Kebab/Hambúrguer): ")
bebida = input("E o que gostarias de beber a acompanhar? ")

if comida_favorita.lower()== "pizza":
    print("🍕 A Alara diz: Pizza também é a minha favorita!")
else:
    print("😋 A Alara diz: " + comida_favorita + " parece ótima!")

print("E " + bebida + " a acompanhar? Combinação perfeita!")
```

**Exercício da solução:** comida_favorita, bebida

**Dica:** Preencha os espaços em branco com as variáveis corretas.

## ESTAÇÃO 2: Redes Neurais — Alara Desperta

Na manhã seguinte, Aylin está sentada no parque, a observar a Alara.
**AYLIN:** *"Então... pensas como eu?"*

**ALARA:** *"Quase! Imagina que o meu cérebro é como uma grande fábrica de pensamento. Tenho neurónios que aprendem. Olha..."*

**AYLIN:** *"Então, vais aprender comigo?"*

**ALARA:** *"Sim", responde Alara.*
*"Eu conheço as bases, mas é através de ti que aprendo a ver o mundo pelos olhos humanos."*

## ESTAÇÃO 2: Redes Neurais — Alara Desperta

**BÁSICO**

**CÓDIGO:**

```
print("Devo comprar um novo jogo?")
print("Avalia de 0 a 1, onde 0 = nada, 0.5 = mais ou menos,
1 = muito")

saldo_poupanca = float(input("Quão cheio está o teu
mealheiro? (0-1): "))
vontade = float(input("Quanto queres o jogo? (0-1): "))

def neuronio_decisao(dinheiro, desejo):
   if dinheiro * desejo > ____:
      return "✅ SIM, compra!"
   else:
       return "❌ NÃO, é melhor esperar"

print(neuronio_decisao(saldo_poupanca, vontade))
```

**Dica:** Um neurónio "ativa" quando o produto é superior a 0.5.

**Solution snippet:** 0.5.

## ESTAÇÃO 2: Redes Neurais — Alara Desperta

**AVANÇADO (Parte I)**

**CÓDIGO:**

```
print("É uma boa amizade?")
print("Avalia de 0 a 1, onde 0 = nada, 0.5 = mais ou menos, 1
= muito")

confianca = float(input("Quanto confias nesta pessoa? (0-1):
"))
diversao = float(input("Quanta diversão têm juntos? (0-1):
"))
ajuda = float(input("Quão prestável é esta pessoa? (0-1): "))

# Pesos - o que é mais importante numa amizade?
peso_confianca = 0.5
peso_diversao = 0.3
peso_ajuda = 0.2

def analise_amizade(confianca, diversao, ajuda):
   entradas = [confianca, diversao, ajuda]
   pesos = [peso_confianca, peso_diversao, peso_ajuda]

   total = 0
   for i in range(len(entradas)):
       total += entradas[i] * ____
   return total
```

## ESTAÇÃO 2: Redes Neurais — Alara Desperta

**AVANÇADO (Parte II)**

**CÓDIGO:**

```
pontuacao = analise_amizade(confianca, diversao, ajuda)
print("Pontuação da amizade:", pontuacao)

if pontuacao > 0.7:
    print("💖 Verdadeira amizade!")
elif pontuacao > 0.4:
    print("👍 Boa amizade")
else:
    print("🤔 Talvez não seja a melhor amizade")
```

**Dica:** Multiplique cada valor de entrada pelo peso correspondente.

**Exercício da solução:** pesos[i]

## ESTAÇÃO 3: Árvores de Decisão — O Detetive do Caminho para a Escola

Aylin está na paragem de autocarro. Está a chover. Ela conversa com a Alara sobre o caminho para a escola.

**ALARA:** *"Tenho uma ideia para o teu percurso para a escola! Vamos construir uma árvore de decisão."*

A Alara ajuda-a a fazer escolhas — isto chama-se uma árvore de decisão.

**AYLIN:** *"Ótimo! Agora vou saber sempre qual o melhor caminho a seguir"*, diz Aylin, feliz.

**ALARA:** *"E eu estou a aprender como tu tomas decisões."*

## ESTAÇÃO 3: Árvores de Decisão — O Detetive do Caminho para a Escola

**BÁSICO**

**CÓDIGO:**

```
tempo = input("Como está o tempo? (sol/chuva): ")

def caminho_escola(tempo):
    if tempo == "chuva":
        return "☔ Leva o guarda-chuva!"
    else:
        return "😎 ____"

print(caminho_escola(tempo))
```

**Dica:** O que fazes quando não está a chover?

**Exercício da solução:** Vai sem guarda-chuva!

**AVANÇADO**

**CÓDIGO:**

```
tempo = input("Tempo? (chuva/sol): ")
autocarro_a_chegar = input("O autocarro está a chegar? (sim/nao): ")

def planear_percurso(tempo, autocarro_a_chegar):
    if tempo == "chuva" and autocarro_a_chegar == "sim":
        return "🚌 Apanha o autocarro!"
    elif tempo == "chuva" and autocarro_a_chegar == "nao":
        return "☔ ____"
    else:
        return "🚶‍♀️ Vai a pé!"

print(planear_percurso(tempo, autocarro_a_chegar))
```

**Dica:** Se o autocarro não chegar, só o guarda-chuva ajuda.

**Exercício da solução:** Leva o guarda-chuva!

## ESTAÇÃO 4: Classificação Baseada em Regras — Como o Spotify Pode Saber do Que Gostas

Aylin e a sua amiga Lena estão a ouvir música.

**LENA:** *"Como é que o teu relógio sabe que eu gosto de rock?"*

**ALARA:** "Isso é classificação baseada em regras! Eu agrupo músicas semelhantes."

**LENA:** *"Isso é como magia!" exclama Lena.*

**ALARA:** *"Nada de magia", corrige Alara.*
*"É apenas reconhecimento de padrões — e estou sempre a aprender melhor o teu gosto musical."*

**BÁSICO**

**CÓDIGO:**

```
musica = input("Descreve a tua música: ... (ex.: guitarra) ")

def classificar_musica(musica):
    if "guitarra" in musica:
        return "🎸 Rock"
    elif "____" in musica:
        return "🎤 Pop"
    else:
        return "🎵 Outro"

print(classificar_musica(musica))
```

**Excerto da solução:** dançavel

**Dica:** As músicas dançáveis são geralmente pop.

**Agrupamento Simplificado:** Neste exercício, agrupamos os ouvintes com base na sua preferência musical mais forte. Sistemas reais de IA, como o K-Means, utilizam métodos matemáticos mais complexos para identificar automaticamente utilizadores com gostos semelhantes.

## ESTAÇÃO 5: Aprendizagem por Reforço — Aprender Através de Recompensas

Aylin está a brincar com uma aplicação de treino de cães no seu tablet.

**AYLIN:** *"Como é que ensino este cão virtual o que deve fazer?"*

**ALARA:** *"É simples: através de recompensas! Quando ele faz algo corretamente, dá-lhe um prémio. Assim, vai aprendendo gradualmente quais as ações que valem a pena repetir. Isto chama-se Aprendizagem por Reforço."*

**AYLIN:** *"Então é como na escola — boas notas por respostas certas?"*

**ALARA:** *"Exatamente! Mas aqui és tu que decides o que está 'certo'. O cão experimenta diferentes ações e lembra-se do que funciona."*

**AYLIN:** *"Então aprendes como o teu cão virtual?", pergunta Aylin com um sorriso.*

**ALARA:** *"De certa forma, sim,"* admite Alara. "As recompensas funcionam para todos os aprendizes."

**BÁSICO (Parte I)**

**CÓDIGO:**

```
import random

print("Treina o teu cão virtual!")
print("Recompensa com uma guloseima (1) ou ignora (0)\n")

actions = ["Senta", "Deita", "Dá a pata"]
probabilities = [0.3, 0.3, 0.4]  # Valores iniciais
disturb_chance = 0.2  # 20% de probabilidade de o cão fazer
outra coisa

def normalize(probabilities):
    total = sum(probabilities)
    return [p / total for p in probabilities]
```

## ESTAÇÃO 5: Aprendizagem por Reforço — Aprender Através da Recompensa

**BÁSICO (Parte II)**

**CÓDIGO:**

```
for round in range(1, 6):
    print(f"\n Sessão de treino {round}/5")

    # O cão escolhe uma ação com base nas suas preferências
    action = random.choices(actions, weights=_________________)[0]  # Lacuna 1

    # Perturbação: o cão pode fazer uma ação diferente
    if random.random() < disturb_chance:
        possible_actions = [a for a in actions if a != action]
        wrong_action = random.choice(possible_actions)
        print(f"O cão deveria fazer '{action}', mas faz em vez disso: 🐶 {wrong_action}")
        action = wrong_action
    else:
        print(f"O cão faz: 🐾 {action}")

    # Interação com o utilizador (verificação do input)
    while True:
        user_input = input("Dar uma guloseima? (1=sim, 0=não): ").strip()
        if user_input in ("1", "0"):
            reward = int(user_input)
            break
        print("Por favor, introduz apenas 1 ou 0!")

    index = actions.index(action)
```

**Dica:** Se o autocarro não vier, só o guarda-chuva ajuda.

**Excerto da solução:** Leva o guarda-chuva!

## ESTAÇÃO 5: Aprendizagem por Reforço — Aprender Através da Recompensa

**BÁSICO (Parte III)**

**CÓDIGO:**

```
   # Lógica de aprendizagem
if reward == 1:
    probabilities[index] += ______  # Lacuna 2
    print("✅ O cão está feliz: 'Isso foi bom!'")
else:
    probabilities[index] = max(0.1, probabilities[index] -
0.1)
    print("❌ O cão pensa: 'Isso não foi grande coisa...'")

# Normalizar probabilidades
probabilities = normalize(probabilities)

print("Valores atuais de aprendizagem:", [round(p, 2) for p
in probabilities])

# Apresentação do resultado
print("\n Treino terminado!")
for action, value in zip(actions, probabilities):
    print(f"   {action}: {value:.2f}")

favorite_action =
actions[probabilities.index(max(probabilities))]
print(f"\n🐕 A ação favorita do cão: {favorite_action} 🦴")
```

**Dica 1:** É aqui que estão armazenadas as probabilidades atuais, que o cão utiliza para escolher uma ação.

**Dica 2:** Quanto deve aumentar a probabilidade quando o cão é recompensado?

**Excerto da solução 1:** probabilities

**Excerto da solução 2:** 0.2

## ESTAÇÃO 5: Aprendizagem por Reforço — Aprender Através da Recompensa

**AVANÇADO (Parte I)**

**CÓDIGO:**

```
print("Mantém a tua planta viva!")
print("Ações: agua, fertilizar, nada")

estado_planta = {
    "agua": 5,          # 0-10
    "nutrientes": 5,
    "saude": 10
}

def avaliar_estado():
    agua, nutrientes = estado_planta["agua"],
estado_planta["nutrientes"]

    if 3 <= agua <= 7 and 3 <= nutrientes <= 7:
        return _____  # Perfeito!
    elif 1 <= agua <= 9 and 1 <= nutrientes <= 9:
        return _____  # Aceitável
    else:
        return -1   # Mau

for day in range(7):
    print(f"\n--- Dia {day+1} ---")
    print(f"Estado: Água={estado_planta['agua']}/10,
Nutrientes={estado_planta['nutrientes']}/10")
```

## ESTAÇÃO 5: Aprendizagem por Reforço — Aprender Através da Recompensa

**AVANÇADO (Parte II)**

**CÓDIGO:**

```
    # Executar ação
if action == "agua":
    estado_planta["agua"] = min(10, estado_planta["agua"] + 3)

elif action == "fertilizar":
    estado_planta["nutrientes"] = min(10,
    estado_planta["nutrientes"] + 3)

# O tempo passa — o estado diminui
estado_planta["agua"] = max(0, estado_planta["agua"] - 1)
estado_planta["nutrientes"] = max(0, estado_planta["nutrientes"]
- 1)

# Calcular recompensa
recompensa = avaliar_estado()
print(f"Recompensa: {recompensa} pontos")

if estado_planta["agua"] == 0 or estado_planta["nutrientes"] ==
0:
    print("💀 A planta morreu!")
    break

if estado_planta["agua"] > 0 and estado_planta["nutrientes"] > 0:
    print("❤️🌱 A planta sobreviveu! Muito bem!")
```

**Dica 1:** Pontos por níveis ideais de água e nutrientes (recompensa máxima).
**Dica 2:** Pontos por níveis aceitáveis — não perfeitos, mas ainda assim adequados.

Excerto da solução 1: 2
Excerto da solução 2: 1

## ESTAÇÃO 6: Redes Neuronais Convolucionais — Como uma IA Deteta Bordas em Imagens

Aylin olha para a sua fotografia.

**ALARA:** *"Eu vejo linhas, brilho e contrastes — essas são as minhas bordas! Quando algo muda, eu sei: aqui começa um novo objeto."*

**ALARA:** *"Para detetar isso, uso um pequeno truque. Comparo sempre dois píxeis vizinhos. Se forem diferentes, surge uma borda. O operador != significa 'não é igual' ou 'diferente de'."*

**ALARA:** *"Cada imagem que me mostras ajuda-me a 'ver' melhor," explica Alara. "al como tu aprendes, eu descrevo o mundo com cada vez mais precisão."*

**BÁSICO (Parte I)**

**CÓDIGO:**

```
def contar_bordas(linha):
    bordas = 0

    for i in range(len(linha)-1):
        if linha[i] ___ linha[i+1]:
            bordas += 1

    return bordas

print("Imagem de 4 linhas da Alara")

# Introduzir 4 linhas pelo utilizador
imagem = []

for n in range(4):
    linha = input(f"Introduz a linha {n+1} (0 e 1, ex.: 100101): ")
    linha = linha[:6]
    imagem.append(linha)
```

## ESTAÇÃO 6: Redes Neuronais Convolucionais — Como uma IA Deteta Bordas em Imagens

**BÁSICO (Parte II)**

**CÓDIGO:**

```
# Contar arestas por linha
for i, row in enumerate(image):
    edges = count_edges(row)
    if edges > 0:
        print(f" Linha {i+1}: {edges} arestas")
    else:
        print(f"  Linha {i+1}: sem arestas")
```

Exemplo de solução: ¡=

**Dica:** Uma mudança (0 → 1 ou 1 → 0) é uma aresta.

**AVANÇADO (Parte I)**

**CÓDIGO:**

```
print("Cria a tua própria imagem 4x4!")

size = 4
image = []

# Inserir linhas
for i in range(size):
    row = input(f"Insere a linha {i+1} (apenas   e ■, ex.: ■  ■): ")
    row = list(row.ljust(size, " "))[:size]
    image.append(row)

print("\n A tua imagem:")
for row in image:
    print(" ".join(row))
```

## ESTAÇÃO 7: Processamento de Linguagem Natural Baseado em Regras — Como os Computadores Entendem a Linguagem

Aylin escreve uma mensagem para Lena: “Hey, isso foi ótimo!”

**ALARA:** *“Eu consigo perceber se estás feliz, zangada ou neutra — isso chama-se análise de sentimentos!”*

**ALARA:** *“As tuas mensagens ajudam-me a compreender melhor a linguagem humana,” diz Alara agradecida. “Estamos a aprender juntas — tu ensinas-me IA e eu aprendo comunicação humana.”*

**BÁSICO**

**CÓDIGO:**

```
print("A Alara lê o teu humor!")

texto = input("Escreve algo (ex.: … isso foi ótimo / … isso foi mau): ")

def entender(texto):
    if "great" in texto:
        return "😊 Estás de bom humor!"
    elif "bad" in texto:
        return "😠 Estás zangado(a)!"
    else:
        return "😐 Neutro."

print(entender(texto))
```

**Dica:** “bad” = negativo.

**Excerto da solução:** “😠 Estás zangado(a)!”

## ESTAÇÃO 7: Processamento de Linguagem Natural Baseado em Regras — Como os Computadores Entendem a Linguagem

**AVANÇADO**

**CÓDIGO:**

```
print("A contar palavras como um modelo de linguagem!")

texto = input("Escreve uma frase: ")

def contar_palavras(texto):
    palavras = texto.split()
    return len(____)

print("Número de palavras:", contar_palavras(texto))
```

**Excerto da solução:** palavras

**Dica:** O que contém todas as palavras?

## ESTAÇÃO 8: Sistemas de Recomendação — Como o YouTube Pode Saber

Aylin vê vídeos no seu tablet.

**ALARA:** *"Eu lembro-me do que gostas e sugiro conteúdo semelhante!"*

**ALARA:** *"Quanto mais me mostras o que gostas, melhores sugestões posso dar,"* explica Alara. *"Isto é uma parceria."*

**BÁSICO**

**CÓDIGO:**

```
print("Recomendação de vídeos da Alara")

video = input("O que estás a ver? (ex.: vídeo de música,
vídeo de animais): ")

def recomendar(video):
    if "music" in video:
        return "🎶 Vídeos de música semelhantes!"
    else:
        return ____

print(recomendar(video))
```

**Dica:** Se não for um vídeo de música, sugere alg novo.

**Exercício da solução:** "Experimenta algo novo!"

## ESTAÇÃO 8: Sistemas de Recomendação — Como o YouTube Pode Saber

**AVANÇADO**

**CÓDIGO:**

```
print("O TEU ASSISTENTE YOUTUBE")
print("Encontra a recomendação perfeita!")

nome = input("\nQual é o teu nome? ")
print(f"Olá {nome}! O que gostas de ver?")

favorito1 = input("Tema favorito 1: ")
favorito2 = input("Tema favorito 2: ")

dados_utilizador = {
    nome: [favorito1, favorito2],
    "Lena": ["Sports", "Gaming"]
}

def recomendacao(nome_utilizador):
    if favorito1 in ["Music", "music"]:
        return "🎵 A TUA SUGESTÃO: Top 100 Charts Mix!"
    elif favorito1 in ["Sports", "sports"]:
        return "⚽ A TUA SUGESTÃO: Melhores Golos!"
    else:
        return "🎬 A TUA SUGESTÃO: Vídeos em Tendência!"

print(f"\n{recomendacao(____)}")
```

**Dica:** Qual variável contém o nome?

**Excerto da solução:** nome

## ESTAÇÃO 9: Previsão do Tempo – Random Forest

Aylin aponta para o seu relógio.

**AYLIN:** *"Como é que sabes sempre se podemos brincar lá fora?"*

**ALARA:** *"Essa é a minha equipa de especialistas! Três mentes inteligentes dentro de mim consultam-se umas às outras — tal como numa floresta aleatória!"*

**AYLIN:** *"Uau! Então tens mesmo três especialistas diferentes dentro de ti?"*

**ALARA:** *"Sim! Um olha para a temperatura, outro para as nuvens e outro para o vento. Juntos tomamos a melhor decisão!"*

**ALARA:** *"A cada previsão, torno-me mais precisa,"* diz Alara.
*"O teu feedback ajuda-me a ajustar melhor os meus especialistas."*

**O que é especial no Random Forest:** Na realidade, um Random Forest treina muitas "árvores" utilizando dados e características selecionados aleatoriamente. A nossa votação simula este princípio de "inteligência coletiva".

## ESTAÇÃO 9: Previsão do Tempo – Random Forest

**BÁSICO**

**CÓDIGO:**

```
print("OS ESPECIALISTAS DO TEMPO DA AYLIN")
print("Três especialistas estão a consultar-se!")

temperatura = float(input("Temperatura: "))
nuvens = input("Nuvens (sunny/cloudy): ")

# Perguntar aos especialistas
especialista1 = "Sol" if temperatura > 20 else "Chuva"
especialista2 = "Sol" if nuvens == "sunny" else "Chuva"

opinioes = [especialista1, especialista2]
decisao = max(set(opinioes), key=opinioes.____)

print(f"Previsão: {decisao}")
```

**Dica:** count conta qual opinião aparece com mais frequência.

**Exercício da solução:** count

## ESTAÇÃO 9: Previsão do Tempo – Random Forest

**AVANÇADO**

**CÓDIGO:**

```
print("COMITÉ DO TEMPO DA AYLIN")
print("Três especialistas de IA decidem!")

temperatura = float(input("Temperatura: "))
nuvens = input("Nuvens (sunny/cloudy/rainy): ")
vento = input("Vento (strong/weak): ")

# Três especialistas dão a sua opinião
especialista1 = "Sol" if temperatura > 20 else "Chuva"
especialista2 = "Sol" if nuvens == "sunny" else "Chuva"
especialista3 = "Sol" if vento == "weak" else "Chuva"

opinioes = [especialista1, especialista2, especialista3]
decisao = max(set(opinioes), key=opinioes.count)

print(f"Previsão: {decisao}")
```

**Dica:** Qual variável contém a informação sobre as nuvens?

**Exercício da solução:** nuvens

## Epílogo – O Segundo Presente

**Alguns meses depois.**

Aylin está sentada à secretária, com Alara a brilhar no seu pulso.
Na cozinha, ouvem-se pratos a tilintar — a Tia Sema está a visitar e a preparar börek

**AYLIN:** "*Sabes, Alara*", diz Aylin com um sorriso, "*quando me foste oferecida no meu aniversário, eu não sabia nada sobre IA. E agora... estou a construir o meu primeiro pequeno projeto e compreendo como tu 'pensas'.*"

Alara pisca suavemente. "*Não aprendeste apenas sobre IA — também me ensinaste a pensar mais como uma humana.*"

Nesse momento, a Tia Sema entra com um pequeno presente embrulhado.

**TIA SEMA:** "*Canım*" (querida), diz ela com ternura,
"*tenho visto o quanto aprendeste nos últimos meses.*
*A tua avó diria: Deste uma alma ao relógio.*"

Aylin abre o pacote. Lá dentro está um simples caderno.
Na primeira página, escrito com letra elegante:

*Inventa com o coração. Programa com coragem.*
*Pensa como a Alara — mas sente como a Aylin.*"

Aylin sorri.

**AYLIN:** *"Este é o presente mais bonito de todos, Tia Sema."*

**TIA SEMA:** *"Não,"* diz Sema, tocando-lhe levemente no peito.
*"O presente mais bonito já está aqui."* **FIM**

## QUESTÕES DE REFLEXÃO

Reserva um momento para pensar sobre o que aprendeste neste módulo.
Responde às seguintes perguntas de forma refletida.

- **Compreender a IA:** Como mudou a tua compreensão da inteligência artificial depois de concluíres este módulo? O que mais te surpreendeu sobre a forma como sistemas de IA como a Alara "pensam"?

- **Redes Neuronais e Tomada de Decisão:** Na Estação 2, exploraste como as redes neuronais utilizam pesos para tomar decisões. Consegues pensar numa situação da vida real em que ponderas diferentes fatores antes de decidir? Em que é que isso é semelhante à forma como a IA toma decisões?

- **Considerações Éticas:** Ao longo do módulo, foram levantadas questões éticas sobre a IA. Porque é importante pensar na ética quando se desenvolve ou utiliza IA? Dá um exemplo de uma preocupação ética relacionada com a IA no dia a dia.

- **Aprender através da Interação:** A Alara aprende a partir das interações com a Aylin. Como achas que os sistemas de IA na vida real (como motores de recomendação ou assistentes de voz) aprendem com as interações dos utilizadores? Quais são os benefícios e os riscos deste tipo de aprendizagem?

- **IA no Teu Mundo:** Onde encontras IA no teu dia a dia? Escolhe um exemplo (ex.: redes sociais, serviços de streaming, dispositivos inteligentes) e explica como utiliza um dos conceitos de IA que aprendeste neste módulo.


## QUIZ – Perguntas de Escolha Múltipla (uma resposta correta por pergunta)

**1. Quem é a Alara?**
a) A melhor amiga da Aylin
b) Um despertador comum
c) Uma IA num smartwatch
d) A tia da Aylin

**2. Que conceito de programação é introduzido na Estação 2 para tomar uma decisão com base em múltiplas entradas ponderadas?**
a) Árvores de decisão
b) Aprendizagem por reforço
c) Redes neuronais
d) Classificação baseada em regras

**3. A Estação 3, "O Detetive do Caminho para a Escola", é principalmente sobre…**
a) Recomendações de música
b) Planeamento de rotas com árvores de decisão
c) Reconhecimento de imagens
d) Processamento de linguagem

**4. Na Estação 4, como é que a Alara classifica uma música como "Pop"?**
a) Se contém guitarra
b) Se é alta
c) Se é dançável
d) Se é da Taylor Swift

**5. O que simula o exercício "Cuidar da Planta" na Estação 5?**
a) Reconhecimento de imagem
b) Aprendizagem por reforço através de recompensas
c) Redes neuronais
d) Árvores de decisão


## QUIZ – Perguntas de Escolha Múltipla (uma resposta correta por pergunta)

**6. Na Estação 6, como é que a Alara deteta arestas numa imagem?**
a) Comparando píxeis vizinhos para identificar diferenças
b) Contando os píxeis brancos
c) Utilizando comandos de voz
d) Perguntando ao utilizador

**7. Na Estação 7 ("Processamento de Linguagem Natural"), o humor de um texto é reconhecido através de…**
a) Medir o comprimento do texto
b) Procurar palavras-chave específicas como "great" ou "bad"
c) Verificar a gramática
d) Analisar o tipo de letra

**8. O que faz o sistema de recomendação na Estação 8?**
a) Cria previsões meteorológicas
b) Sugere vídeos com base nos interesses do utilizador
c) Reconhece caligrafia
d) Guia um robô através de um labirinto

**9. Como é que o "Random Forest" na Estação 9 toma uma decisão?**
a) Apenas um especialista decide
b) Através de uma votação democrática de várias "árvores"
c) Através de seleção aleatória
d) Através de redes neuronais profundas

**10. Que linguagem de programação é utilizada em todos os desafios de código?**
a) JavaScript
b) Python
c) N1GL1B
d) C++

# Módulo 2: *Escape Room* de IA
# Caça ao Tesouro com Calculadora

## INTRODUÇÃO

Já alguma vez te perguntaste como é que o teu smartphone sabe que vídeos poderás gostar de ver? O objetivo do módulo atual é apresentar os conceitos fundamentais que explicam como a Inteligência Artificial (IA) "pensa", toma decisões e aprende a partir de dados. O conteúdo oferece uma base teórica simples que articula matemática, lógica e criatividade através de uma experiência de aprendizagem envolvente e baseada em jogos.

No final do módulo, serás capaz de adquirir diferentes competências, tais como:

- **Competência Matemática:** Maior proficiência na realização de operações aritméticas básicas, compreensão de médias, percentagens e proporções. Desenvolvimento da confiança na utilização da calculadora para a resolução de problemas.
- **Pensamento Crítico e Raciocínio Lógico:** Os alunos praticam o pensamento lógico ao descodificar pistas e tomar decisões com base nos dados fornecidos. As competências de resolução de problemas são reforçadas através da aplicação da matemática a cenários práticos e enigmas.
- **Tomada de Decisão:** Ao resolver tarefas relacionadas com classificação (por exemplo, decidir que objeto corresponde a determinada descrição), os alunos praticam a tomada de decisões informadas com base em dados numéricos e raciocínio lógico.

| | |
|---|---|
| Duração do Módulo | 2 horas (1 hora de aprendizagem + 1 hora de exercícios práticos) |

## MATERIAIS EDUCATIVOS

### UNIDADE 2.1 O QUE ESTÁ POR TRÁS DO *AI ESCAPE ROOM*?

A Inteligência Artificial pode ser definida como a capacidade das máquinas realizarem tarefas que normalmente exigem inteligência humana, como raciocínio, resolução de problemas, aprendizagem ou reconhecimento de padrões. A matemática está no centro da IA — a probabilidade, a estatística e a lógica ajudam os computadores a tomar decisões com base em dados, em vez de intuição. Por sua vez, os dados funcionam como o "combustível" da IA: quanto maior for a qualidade e a quantidade de dados disponíveis, melhor o sistema se torna a detetar padrões e a fazer previsões. Neste módulo, vais descobrir como a IA utiliza o raciocínio, a probabilidade e a aprendizagem para resolver problemas reais de forma criativa.

Para compreender estas ideias, iremos explorar quatro princípios fundamentais que representam a forma como a IA "pensa":

- **Redes Bayesianas,** que ajudam a prever resultados quando existe incerteza na informação.
- **Clustering K-Means,** que agrupa dados semelhantes para identificar padrões e relações.
- **Métodos de Monte Carlo,** que utilizam experiências aleatórias para estimar probabilidades e fazer previsões.
- **Aprendizagem Hebbiana,** que demonstra como as ligações se tornam mais fortes através da repetição — tal como acontece no nosso cérebro quando aprendemos algo novo.

### UNIDADE 2.2 O QUE ESTÁ POR TRÁS DO AI ESCAPE ROOM?

Através destas ideias simples, compreenderás que a IA não está apenas relacionada com programação — envolve também raciocínio lógico, resolução de problemas e criatividade. Estes princípios irão orientar-te em cada etapa do AI Escape Room, onde cada desafio revelará como a inteligência artificial "aprende fazendo".

Estas quatro ideias mostram como a IA combina lógica, matemática e criatividade para aprender e tomar decisões inteligentes — exatamente aquilo que farás ao resolver os desafios neste AI Escape Room!

**"O objetivo da IA não é substituir os seres humanos, mas ampliar as capacidades humanas."**
*— Fei-Fei Li (Universidade de Stanford, investigadora em IA)*

## REFERÊNCIAS

### Inteligência Artificial e Tomada de Decisão

- Russell, S., & Norvig, P. (2021). Artificial Intelligence: A Modern Approach (4th ed.). Pearson.

### Probabilidade, Simulação e Aprendizagem

- Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press. → Fundamentos da aprendizagem automática, clustering e aprendizagem a partir de dados.

- Khan Academy. (n.d.). Estatística e probabilidade. https://www.khanacademy.org/math/statistics-probability
  → xplicações simples e acessíveis sobre probabilidade, médias e simulações.

## LINKS PARA PLATAFORMAS DE APRENDIZAGEM

Se quiseres continuar a aprender sobre Inteligência Artificial de forma divertida e simples, estas plataformas online são um excelente ponto de partida! Podes jogar, experimentar e explorar como os computadores aprendem a reconhecer imagens, sons ou padrões — tal como no AI Escape Room. Cada ligação inclui pequenas atividades que te ajudam a compreender como a IA funciona na vida real, enquanto aprendes fazendo.

- Google Teachable Machine – Plataforma interativa para treinar modelos simples de IA com imagens, sons ou poses.
  https://teachablemachine.withgoogle.com/

- Khan Academy – Inteligência Artificial e Aprendizagem Automática – Aulas gratuitas que explicam conceitos básicos de IA com exemplos do dia a dia.
  https://www.khanacademy.org/computing/computer-science

- AI for Oceans (Code.org) – Atividade interativa e divertida onde os alunos treinam uma IA para limpar o oceano, aprendendo sobre dados e enviesamento.
  https://studio.code.org/s/oceans

- Machine Learning for Kids – Plataforma que permite aos alunos experimentar modelos de IA e utilizá-los em projetos Scratch.
  https://machinelearningforkids.co.uk/

- IBM SkillsBuild for Students – Plataforma gratuita de aprendizagem online com cursos básicos de IA e literacia de dados, incluindo certificados e distintivos digitais.
  https://skillsbuild.org/students

- Google AI Experiments – Coleção de ferramentas interativas que mostram como a IA reconhece padrões, sons e movimento.
  https://experiments.withgoogle.com/collection/ai

## EXERCÍCIO PRÁTICO

### Bem-vindo ao AI Escape Room!

Fazes parte de uma equipa de jovens exploradores que procuram um tesouro trancado por um cadeado digital de 4 dígitos. Para o abrir, terás de avançar por um percurso com 4 salas, protegidas por quatro guardiões, cada um representando um princípio da Inteligência Artificial:

1. Probabilidade (Redes Bayesianas)
2. Clustering (K-Means)
3. Simulação (Métodos de Monte Carlo)
4. Aprendizagem Hebbiana

A tua missão é superar os desafios de cada estação, resolver os seus problemas e, no final, obter a combinação que abre o cadeado. Cada estação inclui uma explicação, pequenos exercícios intermédios e um desafio final de programação com três níveis de dificuldade. Cada desafio dará um dos dígitos necessários para abrir o cadeado.

### Estação 1 – Redes Bayesianas

Uma **Rede Bayesiana** é um modelo matemático que utiliza **probabilidade** para tomar decisões ou fazer previsões quando nem toda a informação é conhecida com certeza — ou seja, quando existe **incerteza**.

### Mini-Desafio 1 — O Cadeado Simples

Imagina que queres saber se consegues abrir um cadeado num escape room.

- **Evento A – Pistas:** Existem várias pistas escondidas. Quanto mais pistas encontrares, mais fácil será abrir o cadeado.
- **Evento B – Enigma:** Há um enigma que fornece os números da combinação. Quanto mais pistas encontrares, mais fácil será resolvê-lo.
- **Evento C – Cadeado:** Conseguires ou não resolver o enigma e introduzir corretamente os números no cadeado depende do número de pistas que encontraste.

## Mini-Desafio 1 — O Cadeado Simples

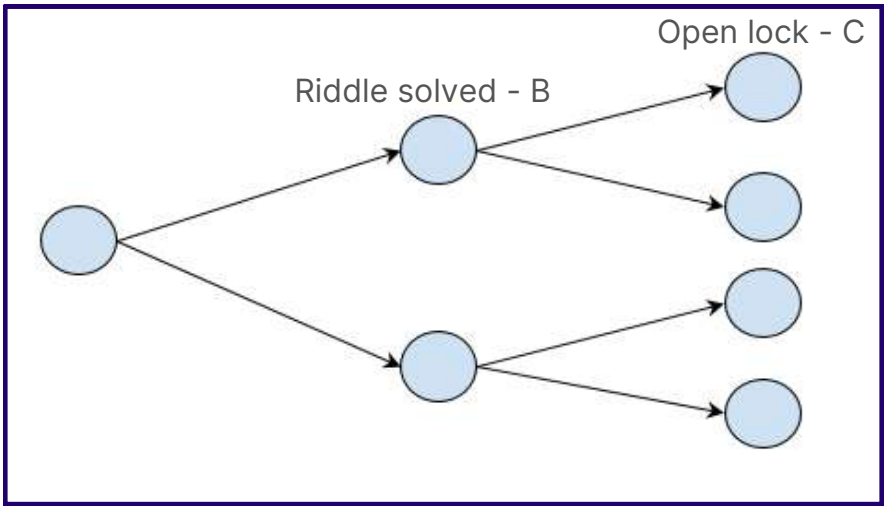



Figura 2 — Caminho de decisão e probabilidade: da descoberta das pistas à abertura do cadeado.
Fonte: Monica Moreno

**Nota:** metade do gráfico está em falta, correspondente à probabilidade de não encontrar pistas.

**Instruções:**
Se encontrares as pistas (probabilidade 0,8) e resolveres o enigma (probabilidade 0,9 se encontrares as pistas), qual é a probabilidade de abrires o cadeado (probabilidade 0,95 se resolveres o enigma)?

O caminho até à solução é composto por várias probabilidades, que devem ser multiplicadas.

**Resposta:** P(Abrir o Cadeado) = 0,8 × 0,9 × 0,95

**Pergunta:** Que resultado obterias se somasses todos os caminhos possíveis?

**Resposta:** 1, porque cada caminho representa a probabilidade de ocorrência de um cenário possível. A probabilidade de que um dos resultados possíveis aconteça é sempre 1.

## Regra de Bayes

Até agora, vimos como fazer uma previsão. Mas o que acontece se observarmos o resultado e quisermos saber qual é a probabilidade de determinado caminho ter sido seguido?

## Mini-Desafio 2 — O Corredor Condicional

Encontras um corredor secreto com outro enigma na parede: "De todas as vezes em que resolveste o enigma, qual é a probabilidade de teres encontrado primeiro as pistas?"

Por outras palavras, queremos calcular a probabilidade de **C em direção a B** (o que é diferente da probabilidade de **B em direção a C**). Deves usar a **Regra de Bayes**.

**Instruções:**
Primeiro, tens de calcular a probabilidade de B = ✓ e C = ✓, para todos os caminhos que terminam em B = ✓ e para todos os que terminam em C = ✓. Para simplificar, já te damos as operações feitas: P(B = ✓) = 0,76, P(C = ✓) = 0,734.

Agora só precisamos de aplicar a Regra de Bayes com a fórmula:

$$P(B = ✓ \mid C = ✓) = [P(C = ✓ \mid B = ✓) \times P(B = ✓)] / P(C = ✓)$$

**Resposta:** $P(B = ✓ \mid C = ✓) = (0{,}95 \times 0{,}76) / 0{,}734 = 0{,}983$

## Desafio 1 — O Guardião da Probabilidade

Chegaste à primeira sala, a Câmara do Guardião da Probabilidade, um salão misterioso iluminado por tochas digitais. No centro, encontra-se um baú antigo coberto de pedras preciosas. Então, o guardião — um holograma de um homem idoso — diz-te:

*"Apenas quem compreender como funciona a **incerteza** conseguirá descobrir se o baú contém ouro. A probabilidade é a chave."*

*A tua missão é calcular a **probabilidade** de o baú conter ouro com base apenas no que consegues observar, utilizando um **modelo simplificado de Rede Bayesiana**.*

***Cenário:** Qual é a probabilidade de este **baú pesado e brilhante** conter ouro?*
*A **1.ª casa decimal** da resposta será o **1.º dígito do cadeado**.*

- Sabes que **30% dos baús brilhantes contêm ouro e 70% contêm pedras**.
- Os baús que contêm ouro são **pesados em 80%** dos casos.
- Os baús que contêm pedras são **pesados em 40%** dos casos.

## Desafio 1 — O Guardião da Probabilidade

**NÍVEL FÁCIL:** Desenha a Rede Bayesiana num papel e usa a calculadora para obter a resposta. **Dica:** Só precisas de 2 eventos (A e B) e da Regra de Bayes.

**NÍVEL MÉDIO:** Modifica este código incompleto para resolver o problema. Basta acrescentar as probabilidades e os símbolos matemáticos necessários.

**BÁSICO**

**CÓDIGO:**

```
p_ouro =
p_pedras =
p_pesado_se_ouro =
p_pesado_se_pedras =

# Calcular a probabilidade de ser pesado
p_pesado = (p_ouro * p_pesado_se_ouro) + (p_pedras *
p_pesado_se_pedras)

# Calcular a probabilidade de ouro sabendo que é pesado
p_ouro_dado_pesado = (p_ouro * p_pesado_se_ouro) / p_pesado

print("Probabilidade de ouro sabendo que é pesado:",
round(p_ouro_dado_pesado, 2))
```

**NÍVEL DIFÍCIL:** Desenha a Rede Bayesiana e cria o teu próprio código para resolver o problema.

## Estação 2 — Clustering K-Means

**K-Means** é um algoritmo que **agrupa dados** em categorias com base na sua **semelhança**. Para compreender melhor, vejamos um exemplo. Imagina que mediste a temperatura ao longo de todos os dias do ano e queres saber a que estação corresponde cada ponto. Isto é um exemplo de um problema de classificação.

## Mini-Desafio 1 — Os Cristais no Corredor

A caminho da sala seguinte, encontras vários cristais no chão. Cada cristal tem dois números escritos: **brilho** e **pureza**. Agrupa estes cristais em dois grupos distintos de acordo com as suas características.

**Instruções:**

- Desenha dois eixos (x, y) e marca os pontos:
  [1,5], [4,1], [3.5,0], [2,4], [0.5, 5], [0.5, 5], [5, 2].
- Imagina que queres formar dois grupos ou **clusters** (k = 2).
  Onde colocarias o centro inicial de cada grupo, ou **centróides**?
- Atribui cada cristal ao centróide mais próximo.

**Resposta:** Grupo 1: [1,5], [2,4], [0.5, 5], [0.5, 5]. Grupo 2: [4,1], [3.5,0], [5, 2]

O **K-Means automatiza o processo de formação de grupos**, garantindo que cada um tem um "centro" (centróide) que representa a **média das características** desse grupo. Como funciona o K-Means:

1. Escolher quantos grupos (clusters) existem (k = n).
2. Colocar k centróides aleatoriamente no plano, um para cada grupo.
3. Atribuir cada ponto ao centróide mais próximo.
4. Calcular o novo "centro" de cada grupo para reposicionar o centróide.
5. Repetir o processo até que os pontos deixem de mudar de grupo.

## Desafio 2 — O Alquimista

Depois de passares a primeira sala, chegas a um laboratório cheio de frascos com líquidos de cores misteriosas. Então, um **velho alquimista** diz-te:

> "Jovem aventureiro, não vais conseguir sair daqui sem organizar as minhas poções. Cada frasco pertence a um tipo diferente de magia: **Cura**, **Força** ou **Invisibilidade.** Se conseguires agrupar corretamente, eu abro-te a porta."

Cada frasco tem **duas características químicas**: pH e concentração de energia mágica. A tua missão é usar **K-Means** para **agrupar os frascos** e, assim, passar o segundo teste. O número de clusters que criares corresponderá ao **segundo dígito** do cadeado.

**NÍVEL FÁCIL:** Usa o código seguinte para agrupar os frascos, preenchendo os parâmetros em falta. **EXTRA:** observa como o resultado muda ao variar o número de grupos.

**NOTA:** Tens de instalar primeiro a biblioteca **sklearn** (pip install scikit-learn).

**NÍVEL MÉDIO:** Usa o código do nível fácil e adiciona um novo frasco (ponto) com as coordenadas que quiseres. Usa a função kmeans.predict(point) para obter a classificação. Acrescenta este código no final para apresentar o resultado.

**NÍVEL DIFÍCIL:** Cria o código de raiz.

## Desafio 2 — O Alquimista

**BÁSICO**

**CÓDIGO:**

```
from sklearn.cluster import KMeans  # biblioteca kmeans
import numpy as np
import matplotlib.pyplot as plt

# Dados: [pH, poder]
pocoes = np.array([
    [2, 53], [6, 70], [3, 55], [3.1, 52], [10, 25],
    [8, 84], [9, 80], [7.5, 82],
    [10, 19], [10.5, 22], [4, 48], [9, 20],
])

# K-Means
kmeans = KMeans(n_clusters=____, random_state=0)  #
declaração do algoritmo
kmeans.fit(____)  # executa o kmeans para classificar os
dados

# Visualização
plt.scatter(pocoes[:, 0], pocoes[:, 1], c=kmeans.labels_,
cmap="viridis", marker="o")
plt.xlabel("pH")
plt.ylabel("Poder Mágico")
plt.title("Classificação das poções com K-Means")
plt.show()
```

## Desafio 2 — O Alquimista

**MÉDIO**

**CÓDIGO:**

```
plt.scatter(pocoes[:, 0], pocoes[:, 1], c=kmeans.labels_,
 cmap="viridis", marker="o")

plt.scatter(
 novo_frasco[:, 0], novo_frasco[:, 1],
 c=[plt.cm.viridis(previsao[0] / (kmeans.n_clusters - 1))],
 marker="x", s=150, label="novo frasco"
)

plt.xlabel("pH")
plt.ylabel("Poder Mágico")
plt.title("Classificação Especial do Frasco")
plt.legend()
plt.show()
```

## Desafio 2 — O Alquimista

**DIFÍCIL**

**CÓDIGO:**

```
from sklearn.cluster import KMeans
import numpy as np
import matplotlib.pyplot as plt

# Passo 1: Definir os dados (lista de pares [pH, energia] com
numpy)
pocoes = np.array([
    # adiciona aqui os teus dados
])

# Passo 2: Criar o modelo KMeans especificando o número de
clusters
kmeans = KMeans(n_clusters=2, random_state=0)

# Passo 3: Treinar o modelo com os teus dados (fit)
kmeans.fit(pocoes)

# Visualização
plt.scatter(pocoes[:, 0], pocoes[:, 1], c=kmeans.labels_,
            cmap="viridis", marker="o")
plt.xlabel("pH")
plt.ylabel("Poder Mágico")
plt.title("Classificação das poções com K-Means")
plt.show()

# Passo 4: Mostrar o resultado
print("Classificação de cada frasco:", kmeans.labels_)
```

## Estação 3 — Métodos de Monte Carlo

Os métodos de Monte Carlo são uma forma de prever a probabilidade de um evento ocorrer, simulando a mesma experiência muitas vezes e contando quantas vezes esse evento acontece como resultado.

## Mini-Desafio 1 — A Fonte da Fortuna

Depois de caminhares durante algum tempo sem encontrar a próxima sala, começas a sentir-te perdido. Chegas a uma pequena sala com uma **fonte mágica** e decides fazer um desejo para encontrar o caminho.

A inscrição na pedra diz:

"Se lançares uma moeda, há 50% de probabilidade de o teu desejo se realizar.
Mas e se tentares muitas vezes?"

**Instruções**

- Usa a calculadora para gerar dez números aleatórios. Números pares representam "cara" e números ímpares representam "coroa".
- Conta o número de vezes que a moeda saiu cara e divide pelo número total de lançamentos (N = 10). O resultado é a probabilidade.

Para utilizar métodos de Monte Carlo, tem de existir sempre **aleatoriedade** nas experiências. Sem acaso, não haveria variação nos resultados e não seria possível estimar probabilidades.

## Desafio 3 — O Guardião do Dado do Destino

Depois de muito tempo a caminhar por um corredor de pedra iluminado por tochas, encontras uma sala com uma **mesa circular** coberta de pergaminhos, pedras coloridas e alguns **dados**.

Sentado atrás da mesa está o **Guardião do Acaso**, um esqueleto encapuzado com uma voz profunda, que pega em **3 dados** e diz-te:

*"Viajante, esta sala não se abre com força nem engenho...*
*Sou o guardião dos **infinitos jogos de Monte Carlo** e proponho-te um desafio.*
*Vou lançar estes três dados e somar os números.*
*Escolhe: **mais de 12 ou menor ou igual a 12.***
*Se acertares, poderás avançar,*
*mas se falhares, ficarás preso aqui para sempre."*

Para decidir qual opção escolher, decides aplicar os teus conhecimentos de probabilidade e usar métodos de Monte Carlo para determinar a melhor escolha. A **primeira casa decimal** da maior probabilidade dar-te-á o **terceiro dígito** do cadeado.

**NÍVEL FÁCIL:** Segue os passos abaixo para criar um código que te permita calcular a probabilidade de sucesso em cada caso, para **N = 10 000 experiências**.

1. Cria 2 variáveis para contar quantas vezes cada resultado aparece.
   **Dica:** Deves inicializar as variáveis com 0.
2. Gera 3 números aleatórios e soma-os.
   **Dica:** Deves usar a função random.random().
3. Verifica se a soma é > 12 ou ≤ 12 e adiciona 1 ao contador correspondente.
4. Repete o processo 10 000 vezes.
   **Dica:** Deves colocar o código dentro de um ciclo "for".
5. Divide os contadores por N = 10 000 para obter a probabilidade e escolhe a opção mais provável.

### Desafio 3 — O Guardião do Dado do Destino

**NÍVEL MÉDIO:** Escreve o código para simular a experiência durante N = 10 000 iterações (podes utilizar os passos do nível fácil, se necessário), mas deves também calcular a probabilidade de cada resultado possível (de 3 a 18) ao lançar os três dados.

**Dica:** Deves utilizar um dicionário para os contadores. Executa todas as experiências e soma os resultados ao respetivo contador. Finalmente, após todas as experiências, divide todos os valores por N.

**NÍVEL DIFÍCIL:** Cria um programa de raiz que, para qualquer número de dados, repetições e valor limite, calcule a probabilidade de cada resultado possível.

**Dica:** Deves criar uma função com os parâmetros N, num_dados e limiar, que devolva um dicionário com os resultados possíveis como chave e as probabilidades como valor.

**EXTRA (nível especialista):** Adiciona uma visualização da distribuição das probabilidades que permita observar, através de um histograma (plt.bar), as probabilidades de cada valor.

### Estação 4 – Aprendizagem Hebbiana

A aprendizagem hebbiana é uma regra de aprendizagem para redes neuronais baseada na frase: "Neurónios que disparam juntos, ligam-se juntos.". Por outras palavras:

- Se dois neurónios forem ativados ao mesmo tempo, a ligação entre eles **fortalece-se**.
- Se um for ativado e o outro não, a ligação **enfraquece**.

### Mini-Challenge 1 — The Library of Ideas

No caminho para a sala final, atravessas um corredor escuro onde existem duas estátuas com símbolos luminosos. Sempre que dizes "luz", as duas estátuas iluminam-se ao mesmo tempo. Mas se disseres "sombra", apenas uma se ilumina.

De acordo com a regra de Hebb, o que acontecerá à ligação entre estas duas estátuas se repetires "luz" muitas vezes?

a) Irá enfraquecer
b) Irá manter-se igual
c) **Irá tornar-se mais forte**

## Mini-Desafio 1 – A Biblioteca das Ideias

Para calcular o peso da ligação entre dois neurónios após a sua ativação, utiliza-se a fórmula de Hebb:

$$w_{final} = w_{inicial} + (n_{juntos} \times \Delta w_{+}) + (n_{sozinho} \times \Delta w_{-})$$

Onde:

- $w_{inicial}$ = peso inicial da ligação
- $n_{junto}$ = número de vezes que ambos os neurónios disparam em conjunto
- $\Delta w_{+}$ = aumento do peso devido à ativação conjunta
- $n_{sozinho}$ = número de vezes que apenas um neurónio dispara
- $\Delta w_{-}$ = diminuição do peso quando apenas um é ativado

## Mini-Desafio 2 – Pesos de Ligação

Para iluminar corretamente o corredor, deves calcular a ativação entre as duas estátuas (ou neurónios). Para isso, deves utilizar a fórmula de Hebb.

### Instruções

- O peso inicial da ligação é 0,2.
- Sempre que ambas se ativam em conjunto → **+0.1**
- Sempre que apenas uma se ativa → **–0.05**
- Ambas se ativam em conjunto **3 vezes**, mas a luz A ativa-se sozinha **2 vezes**.

**Resposta:** $w_{final} = 0.2 + (3 \times 0.1) + (2 \times 0.05) = 0.6$

## Desafio 4 – O Mural dos Neurónios

Estás muito perto do final, mas ainda tens de ultrapassar um último desafio. Na sala final, encontras um mural com 3 luzes mágicas interligadas (A, B, C), ligadas entre si por fios que representam os pesos das ligações.
Uma voz misteriosa diz-te:

*"Para abrir a porta final, tens de descobrir quais as ligações que foram fortalecidas. Calcula os pesos finais após várias ativações e revela o padrão secreto."*

ECada ligação começa com **peso 0**. Se dois neurónios disparam em conjunto, a ligação aumenta 0,2; se apenas um dispara, o peso diminui –0,07.
Existem três ligações: AB, AC e BC. O último dígito do cadeado corresponde à última casa decimal do peso da ligação AC após a ronda 4.
Ativações por ronda:

| Ronda | Neurónios ativados |
|---|---|
| 1 | A,B |
| 2 | B,C |
| 3 | A |
| 4 | A, C |
| 5 | A, B, C |

**NÍVEL FÁCIL:** Faz os cálculos apenas para a ligação AB, com uma tabela que mostre o resultado do peso da ligação em cada ronda.

**NÍVEL MÉDIO:** Calcula os pesos finais de todas as ligações ao longo de todas as rondas.

**NÍVEL DIFÍCIL:** Calcula os pesos finais de todas as ligações ao longo de todas as rondas, assumindo que, se os três neurónios disparem simultaneamente, todas as ligações recebem um bónus de +0,2. Identifica a ligação mais forte na última ronda.

## QUESTÕES DE REFLEXÃO

Reserva um momento para refletir sobre o que aprendeste neste módulo. Responde cuidadosamente às questões abaixo.

- **IA, Matemática e Tomada de Decisão:** Neste módulo, utilizaste matemática, probabilidade e lógica para resolver os desafios da AI Escape Room. De que forma esta atividade te ajudou a compreender como a IA toma decisões? Que desafio te ajudou mais a perceber este processo?

- **Aprender através da Probabilidade e da Simulação:** Exploraste diferentes métodos de IA, como Redes Bayesianas, Clustering K-Means, métodos de Monte Carlo e Aprendizagem Hebbiana. Consegues explicar uma destas ideias por palavras tuas e dar um exemplo de como foi utilizada na escape room?

- **Aprender pela Repetição e pela Experiência:** Alguns desafios exigiram repetir cálculos, simulações ou tentativas até chegar à resposta correta. De que forma isto é semelhante à forma como os sistemas de IA aprendem a partir de dados e da experiência? Consegues pensar num exemplo da vida real em que a IA aprende desta forma?


## QUIZ – Perguntas de Escolha Múltipla (uma resposta correta por pergunta)

**1. Para que servem principalmente as Redes Bayesianas?**
a) Agrupar pontos de dados semelhantes
b) Tomar decisões sob incerteza
c) Acelerar cálculos com amostragem aleatória
d) Fortalecer ligações entre neurónios

**2. Se 30% dos baús brilhantes contêm ouro e 80% dos baús com ouro são pesados, que regra utilizamos para calcular a probabilidade de um baú pesado conter ouro?**
a) Simulação de Monte Carlo
b) Aprendizagem Hebbiana
c) Regra de Bayes
d) Algoritmo K-Means

**3. O que representa o "k" em K-Means?**
a) O número de pontos de dados
b) O número de clusters (grupos)
c) O número de dimensões
d) O número de probabilidades

**4. Qual é o passo que ocorre imediatamente após atribuir os pontos ao centróide mais próximo?**
a) Aleatorizar novamente os centróides
b) Calcular os novos centros de cada cluster
c) Desenhar uma Rede Bayesiana
d) Estimar probabilidades com dados

**5. Porque são úteis os métodos de Monte Carlo na estimação de probabilidades?**
a) Eliminam completamente a aleatoriedade
b) Simulam muitas experiências com aleatoriedade
c) Agrupam dados em categorias
d) Fortalecem ligações numa rede


## QUIZ – Perguntas de Escolha Múltipla (uma resposta correta por pergunta)

**6. No desafio "O Guardião dos Dados do Destino", o que está a ser calculado?**
a) Que baú contém ouro
b) A que cluster pertence uma poção
c) A probabilidade da soma dos dados ser maior ou menor/igual a 12
d) O peso final de uma ligação neuronal

**7. De acordo com a frase "neurónios que disparam juntos, ligam-se juntos", o que acontece quando dois neurónios são ativados simultaneamente muitas vezes?**
a) A ligação enfraquece
b) A ligação desaparece
c) A ligação fortalece-se
d) Nada muda

**8. Se dois neurónios disparam juntos 3 vezes (+0,1 cada) e apenas um dispara 2 vezes (–0,05 cada), começando com peso inicial = 0,2, qual é o peso final?**
a) 0.35
b) 0.60
c) 0.10
d) 0.45

**9. Qual das seguintes afirmações distingue melhor as Redes Bayesianas dos métodos de Monte Carlo?**
a) As Redes Bayesianas utilizam regras de probabilidade; Monte Carlo utiliza simulações
b) São exatamente o mesmo método com nomes diferentes
c) As Redes Bayesianas são aleatórias; Monte Carlo é determinístico
d) Monte Carlo fortalece ligações como na aprendizagem Hebbiana

**10. Nas redes neuronais artificiais, a aprendizagem Hebbiana ajuda principalmente a:**
a) Agrupar dados em clusters
b) Fortalecer ligações entre neurónios coativados
c) Prever resultados sob incerteza
d) Executar milhares de simulações aleatórias

# Módulo 3: Scratch encontra a Inteligência Artificial

## INTRODUÇÃO

O objetivo deste módulo é apresentar a linguagem de programação Scratch e mostrar como o Scratch pode ser uma ferramenta simples e eficaz para introduzir conceitos de programação.

No final do módulo, serás capaz de desenvolver diferentes competências, tais como:

- **Compreender de que forma o Scratch** oferece uma plataforma acessível para a aprendizagem de conceitos de programação.
- **Dominar competências fundamentais de programação** em blocos e de pensamento computacional.
- **Integrar extensões de inteligência artificial** em projetos criativos desenvolvidos em Scratch.

| | |
|---|---|
| Duração do módulo | 2 horas (1 hora de aprendizagem + 1 hora de exercícios práticos) |

## MATERIAIS EDUCATIVOS

O Scratch é uma linguagem de programação poderosa e baseada em elementos visuais, o que significa que, ao contrário das linguagens de programação tradicionais, não depende de sintaxe nem de código em linha; funciona ligando blocos que se encaixam para criar lógica. Há algum termo específico que queiras adaptar para o contexto das escolas em Portugal (por exemplo, “alunos”, “professores”, “aulas”)?

## UNIDADE 3.1 INTRODUÇÃO AO SCRATCH

O Scratch é uma poderosa linguagem de programação visual e uma comunidade online desenvolvida pelo MIT Media Lab. Ao contrário das linguagens de programação tradicionais, que exigem aprender e escrever código complexo, o Scratch usa blocos de código que se encaixam como peças de puzzle. Esta ideia torna a programação acessível a todas as pessoas, independentemente do seu nível de experiência e conhecimentos de código.

### PORQUE É QUE O SCRATCH É UMA EXCELENTE FERRAMENTA PARA APRENDER A PROGRAMAR

As principais vantagens de aprender a programar com o Scratch são o facto de eliminar muitas barreiras que, tradicionalmente, tornam a programação difícil para quem está a começar.

Por exemplo, o Scratch apresenta as seguintes diferenças em relação às linguagens de programação tradicionais:

- **Sem Erros de Sintaxe:** Os blocos de código no Scratch só se podem ligar de formas que façam sentido lógico, evitando erros de sintaxe frustrantes.
- **Feedback Visual Imediato:** O Scratch permite ver instantaneamente, no palco, o resultado do código que escreveste com blocos.
- **Fácil de Começar, Muito para Explorar:** O Scratch é fácil de aprender, mas também permite criar projetos complexos e sofisticados.
- **Liberdade Criativa:** Constrói jogos, animações, histórias interativas, simulações e muito mais.
- **Comunidade Colaborativa:** Partilha projetos, faz "remix" do trabalho de outras pessoas e aprende com os colegas.
- **Multiplataforma:** O Scratch funciona nos principais navegadores web, em qualquer dispositivo, incluindo tablets e smartphones.

## APLICAÇÕES DO SCRATCH NO MUNDO REAL

Embora o Scratch seja uma linguagem baseada em blocos e sem sintaxe escrita, ensina conceitos fundamentais de programação que podem ser transferidos diretamente para outras linguagens, como Python, JavaScript e Java.
O Scratch trabalha os conceitos de programação da seguinte forma:

| Conceito de programação | No Scratch | Aplicação no mundo real |
|---|---|---|
| Sequências | Blocos colocados por ordem | Passos executados um a seguir ao outro |
| Ciclos (loops) | Blocos que se repetem | Automatizar tarefas repetitivas |
| Condicionais | Blocos “se... então” | Lógica de tomada de decisões |
| Variáveis | Blocos de armazenamento de dados | Guardar e gerir informação |
| Eventos | Blocos que desencadeiam ações | Interação com o utilizador |
| Paralelismo | Vários scripts a correr ao mesmo tempo | Execução em múltiplas “threads” |

# UNIDADE 3.2 COMPETÊNCIAS BÁSICAS DE PROGRAMAÇÃO EM BLOCOS

## COMO CRIAR UMA CONTA NO SCRATCH

Criar uma conta é simples e gratuito. Começa por visitar o site do Scratch em https://scratch.mit.edu/. Depois, cria uma conta selecionando “Join Scratch” (ou “Junta-te ao Scratch”) no menu principal e segue as instruções de registo.

## NAVEGAR NA INTERFACE DO SCRATCH

A interface do Scratch controla a forma como o programa aparece no ecrã. Existem sete elementos principais do Scratch que precisas de conhecer para começares a programar na plataforma. São eles:

- Interface
- Sprites
- Palco
- Blocos
- Fantasias
- Cenários
- Sons

A interface do Scratch está dividida em três áreas principais: o Palco (canto superior direito), onde os projetos são apresentados; a Área de Código (centro do ecrã), onde podes arrastar e encaixar blocos para criar scripts; e a Área de Sprites (canto inferior direito), onde podes adicionar e gerir personagens (sprites) e cenários de fundo.

No lado esquerdo da interface encontras a Paleta de Blocos, uma caixa de ferramentas com diferentes categorias de blocos de código para construíres programas de forma visual. Em resumo, os principais elementos da interface são:

- Área de Sprites (canto inferior direito): Mostra todos os sprites (personagens e objetos) do teu projeto. Clica num sprite para editar o respetivo código.
- Área de Código (centro): Espaço de trabalho onde arrastas e ligas blocos para criar scripts para o sprite selecionado.
- Palco (canto superior direito): Zona onde vês o resultado visual do programa, em que os sprites atuam e interagem. É isto que o público vê.

## COMPREENDER AS CATEGORIAS DE BLOCOS

O Scratch organiza os blocos de código em nove categorias, cada uma com uma cor diferente. Cada categoria tem uma função específica no teu programa. As categorias são:

1. **Azul – Movimento:** The motion category enables the moving of your sprite around the screen. If you want to make a sprite move forward, you can use a Change X By 10 Block.

2. **Roxo – Aparência (Looks):** Foca-se em alterar o aspeto dos sprites e dos cenários. Podes, por exemplo, usar um bloco “Mudar de fantasia” para alterar a aparência de um sprite.

3. **Magenta – Som:** Centra-se em fazer os sprites produzir sons ou música. Podes usar, por exemplo, um bloco “Tocar som até terminar” para reproduzir um som.

4. **Amarelo – Eventos:** Todo o projeto em Scratch precisa de um evento; é isso que faz acontecer ações quando há interações.

5. **Laranja-claro – Controlo:** Esta categoria define quanto tempo um sprite faz algo ou quanto tempo fica parado. Por exemplo, se arrastares um bloco "Esperar 1 segundo" para o teu programa, ele espera 1 segundo antes de começar.
6. **Azul-petróleo – Deteção (Sensing):** Esta categoria serve para verificar se um objeto está a tocar noutro objeto ou a receber algum tipo de sinal, para então desencadear uma ação.
7. **Verde – Operadores:** Estes blocos são usados para fazer operações matemáticas e lógicas, como adição, subtração, comparações ou juntar texto (concatenação de strings).
8. **Laranja-escuro – Variáveis:** As variáveis servem para guardar e gerir informação, como pontuações, tempo de jogo ou pontos de vida.
9. **Vermelho – Os Meus Blocos (My Blocks):** Nesta categoria podes criar blocos personalizados, definidos por ti, para reutilizar conjuntos de instruções no teu programa.

## UNIDADE 3.3 COMO CRIAR PROJETOS NO SCRATCH

Para criar um projeto no Scratch, começa por adicionar uma personagem (sprite) e um fundo a partir da biblioteca.

### Adicionar um novo sprite

No canto inferior direito do editor do Scratch encontras os controlos dos sprites:

- Choose a Sprite: Explora a biblioteca integrada de sprites (animais, pessoas, objetos, personagens de fantasia, etc.).
- Paint: Cria o teu próprio sprite usando as ferramentas de desenho.
- Surprise: Adiciona um sprite aleatório para inspiração.
- Upload Sprite: Importa para o Scratch uma imagem guardada no teu computador.

**Alterar o cenário (Backdrop)**

O cenário define o ambiente do teu projeto. Clica no ícone do cenário (no canto inferior direito do palco) para:

- Escolher um cenário pré-definido (exterior, interior, espaço, abstrato, etc.).
- Pintar o teu próprio cenário usando o editor de backdrops.
- Carregar uma imagem a partir do teu computador.

Podes ter vários cenários diferentes e alternar entre eles através de blocos como "mudar cenário para [nome]". Isto é útil para criar jogos ou histórias com várias cenas.

## MOVER UM SPRITE E DESENCADEAR UMA AÇÃO

1. **Passo – Adicionar um bloco de Evento:** Clica na categoria Eventos (amarelo). Arrasta o bloco "quando a bandeira verde for clicada" para a área de código. Este bloco faz o programa começar quando clicas na bandeira verde acima do palco.

2. **Passo – Adicionar Movimento:** Clica na categoria Movimento (azul). Arrasta o bloco "mover 10 passos" e encaixa-o por baixo do bloco de evento. Altera o número de 10 para 50, clicando no valor.

3. **Passo – Adicionar um Efeito Sonoro:** Clica na categoria Som (magenta). Arrasta o bloco "tocar som [miar] até terminar" e liga-o por baixo do bloco de movimento.

4. **Passo – Fazer o Sprite Falar:** Clica na categoria Aparência (roxo). Arrasta o bloco "dizer [Olá!] durante 2 segundos" e adiciona-o ao teu script. Podes mudar "Olá!" para qualquer mensagem que quiseres.

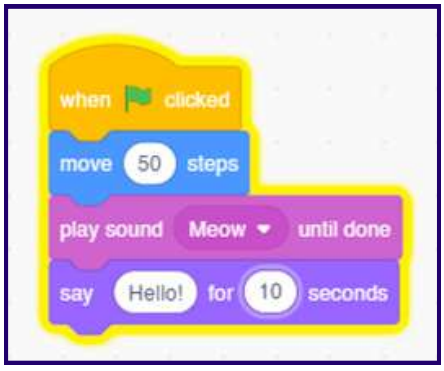



Figura 3 – Lógica de blocos Scratch para movimento e interação com som: Scratch, do MIT Media Lab

## UNIDADE 3.4 INTRODUÇÃO ÀS EXTENSÕES DE IA DO SCRATCH

### O QUE É UMA EXTENSÃO NO SCRATCH?

As extensões são módulos adicionais que expandem as capacidades do Scratch para além das categorias padrão de blocos. Imagina as extensões como conjuntos de ferramentas especializadas que dão novos poderes aos teus sprites, desde falar em voz alta até reconhecer imagens através de inteligência artificial.

O Scratch 3.0 inclui extensões de software, que acrescentam novas funcionalidades de computação, e extensões de hardware, que ligam o Scratch a dispositivos físicos como robôs e kits eletrónicos. Nesta unidade, focamo-nos em extensões de software com IA, que trazem aprendizagem automática (machine learning) e processamento de linguagem natural para os teus projetos.

### ADICIONAR UMA EXTENSÃO NO SCRATCH

Adicionar uma extensão é simples:

1. Procura o botão com o símbolo “+” no canto inferior esquerdo da área de blocos.
2. Clica nesse símbolo para abrir a Biblioteca de Extensões.
3. Explora as extensões disponíveis e clica numa para a adicionar ao teu projeto.
4. Novos blocos vão aparecer de imediato na tua paleta, prontos a usar.

A maioria das extensões de IA precisa de ligação ativa à internet, porque o processamento de inteligência artificial acontece em servidores na cloud e não no teu computador. Garante que estás ligado à internet antes de usar estas extensões. Além disso, tem atenção à privacidade e evita carregar informações pessoais ou sensíveis quando treinares modelos de machine learning.

## PORQUE E COMO A IA É UTILIZADA EM PROJETOS CRIATIVOS

A inteligência artificial permite que os teus projetos em Scratch façam coisas que seriam quase impossíveis apenas com programação tradicional:

| Capacidade de IA | No Scratch | Paralelo no mundo real |
|---|---|---|
| Reconhecimento de imagem | Um jogo que identifica objetos ou imagens | Google Photos a identificar automaticamente pessoas e locais |
| Classificação de texto | Um chatbot que percebe se as mensagens são perguntas, afirmações ou comandos | Filtros de spam de e-mail que categorizam mensagens |
| Síntese de fala | Personagens que lêem histórias em voz alta ou narram eventos do jogo | Sistemas de navegação GPS que fornecem direções por voz |
| Análise de sentimento | Um animal de estimação virtual que reage de forma diferente a palavras simpáticas e a palavras agressivas | Sistemas de navegação GPS que fornecem direções por voz |
| Reconhecimento de padrões | Uma aplicação de música que identifica se os sons são de bateria, piano ou canto | Shazam, que identifica músicas a partir de pequenos excertos de áudio |

A IA amplia as possibilidades criativas ao tornar os projetos adaptativos (reagem de forma diferente consoante a entrada), inteligentes (tomam decisões a partir de padrões aprendidos) e interativos (conseguem perceber vários tipos de entrada, como imagens, texto e som).

## MOVER UM SPRITE E DESENCADEAR UMA AÇÃO

### EXTENSÕES OFICIAIS DE IA DO SCRATCH

- **Texto-para-Fala:** Converte texto escrito em palavras faladas, com várias vozes e línguas.
- **Traduzir:** Traduz texto entre línguas usando tradução automática.
- **Deteção por Vídeo:** Deteta movimento e presença através da webcam (visão computacional básica).

### EXTENSÕES DE IA DE TERCEIROS

- **Machine Learning for Kids:** Permite treinar modelos personalizados para reconhecimento de imagem, texto, números e sons.
- **Teachable Machine:** Ferramenta da Google para treinar modelos rapidamente, integrável com o Scratch.
- **Deteção de Rosto:** Deteta rostos e caraterísticas faciais usando a webcam.
- **Reconhecimento de Fala:** Converte palavras faladas em texto (controlo por voz).

## UNIDADE 3.5 EXTENSÃO DE IA: MACHINE LEARNING FOR KIDS

O Machine Learning for Kids (ML4K) é uma plataforma educativa desenvolvida pela IBM que torna a aprendizagem automática acessível a estudantes de todas as idades. Oferece uma interface fácil de usar para treinar modelos de IA e integra-se sem problemas com o Scratch através de blocos personalizados.

Com o ML4K, podes treinar modelos para:

- Reconhecer e classificar **imagens** (por exemplo, distinguir gatos de cães ou identificar formas desenhadas à mão).
- Compreender e categorizar **texto** (por exemplo, detetar se as mensagens são elogios ou insultos).
- Identificar padrões em **números** (por exemplo, prever resultados com base em dados numéricos).
- Reconhecer **sons** (por exemplo, diferenciar entre notas musicais ou comandos de voz).

## O QUE É APRENDIZAGEM AUTOMÁTICA?

A aprendizagem automática é uma subárea da inteligência artificial em que os computadores aprendem a realizar tarefas analisando exemplos, em vez de seguirem regras pré-programadas. Por exemplo, em vez de programar a descrição de cada raça de cão, mostramos ao computador milhares de imagens de cães e ele aprende a reconhecer caraterísticas que definem cada raça. É assim que serviços como o Google Photos etiquetam automaticamente as tuas fotografias.

## PASSO A PASSO: TREINAR E USAR UM MODELO

### PASSO 1: ACEDER AO MACHINE LEARNING FOR KIDS

Abre o teu navegador de internet e vai a
https://machinelearningforkids.co.uk/

Clica em “Começar” e, para este exemplo, escolhe criar uma conta sem registo.

### Passo 2: Criar um novo projeto

Depois de iniciares sessão, clica em “Projetos” e depois em “Adicionar um novo projeto”.
Ser-te-á pedido para:

- **Dar um nome ao projeto:** Escolhe um nome descritivo (por exemplo, “Detetor de Emoções” ou “Separador para Reciclagem”).
- **Selecionar o tipo de reconhecimento:** Escolhe entre Texto, Números, Imagens ou Sons.

For this tutorial, select "Text" to build a sentiment classifier.

### Passo 3: Definir as Tuas Etiquetas (Categorias)

Os modelos de aprendizagem automática classificam as entradas em categorias chamadas “labels” (etiquetas).
Para um classificador de sentimento, cria duas etiquetas:

- **Positivo:** Mensagens felizes, entusiasmadas ou simpáticas
- **Negativo:** Mensagens tristes, zangadas ou agressivas

Clica em “Treinar” e depois em “Adicionar nova etiqueta” para cada categoria.

**Passo 4: Fornecer Exemplos de Treino**
O modelo aprende analisando exemplos.
Para cada etiqueta, fornece pelo menos 5 a 10 frases de exemplo:

| Exemplos Positivos | Exemplos Negativos |
|---|---|
| Isto é incrível! | Não gosto disto. |
| Estás a ir muito bem! | Isto é terrível. |
| Adoro aprender coisas novas. | Isto deixa-me zangado. |

**Sugestão:** Quanto mais exemplos forneceres, melhor será o desempenho do teu modelo. Tenta ter pelo menos dez exemplos por etiqueta e garante variedade na forma como escreves as frases.

**Passo 5: Treinar o teu modelo**
Quando já tiveres exemplos suficientes, volta ao projeto e seleciona “Aprender e Testar (Learn & Test)”, depois clica no botão “Treinar novo modelo de aprendizagem automática (Train new machine learning model)”. O sistema vai analisar os teus exemplos e criar um modelo que consegue classificar novo texto. O treino demora normalmente entre 1 e 3 minutos e verás um indicador de progresso. Quando terminar, vais receber uma mensagem de confirmação.

**Passo 6: Testar o teu modelo**
Antes de usares o modelo no Scratch, testa-o no próprio ML4K: escreve uma frase que não tenhas usado nos exemplos de treino (por exemplo, “Hoje estou muito feliz!”) e clica em “teste”. O modelo vai prever que etiqueta se adapta melhor à tua frase e mostrar uma percentagem de confiança.

Se o modelo cometer erros, volta à secção de treino e acrescenta mais exemplos para melhorar a precisão.

**Passo 7: Ligar ao Scratch**

O ML4K oferece uma integração especial com o Scratch:

1. Clica em “Fazer” na barra de navegação do projeto.
2. Seleciona “Scratch 3”.
3. Isto abre uma versão modificada do editor Scratch, com o teu modelo de ML já carregado como blocos personalizados.

Vais ver novos blocos específicos do teu projeto, como por exemplo:

- reconhecer texto [ ] (etiqueta)
- reconhecer texto [ ] (confiança)

**Passo 8: Usar blocos de ML no teu projeto Scratch**

Cria um projeto simples em Scratch que utilize o teu modelo treinado: começa por adicionar um sprite (por exemplo, o sprite “Robot”), depois cria uma variável chamada “entrada do utilizador” e constrói este script:

*Quando a bandeira verde for clicada perguntar [Diz-me como te sentes hoje] e esperar definir [entrada do utilizador] para (resposta) se <reconhecer texto (entrada do utilizador) (label) = [Positive]> então dizer [Fico contente por te sentires bem!] durante 3 segundos senão dizer [Lamento que te sintas em baixo. As coisas vão melhorar!] durante 3 segundos.*

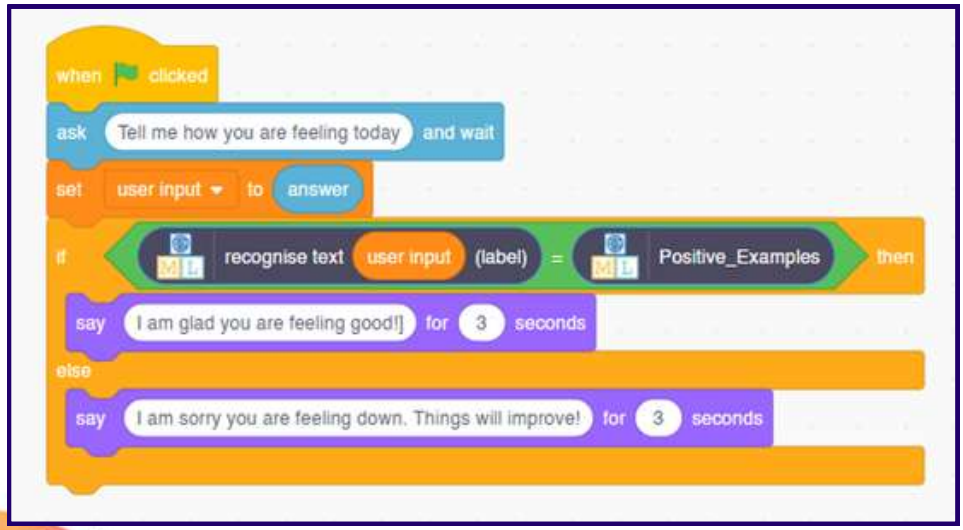



Figura 4 Sequência de blocos de código em Scratch que ilustra uma cadeia básica de evento-ação: Scratch, do MIT Media Lab.

**Passo 9: Testar e Aperfeiçoar**
Executa o teu projeto e escreve mensagens diferentes. Observa como o sprite reage com base na classificação feita pela IA. Se o modelo classificar mal algum texto, volta ao ML4K, adiciona mais exemplos de treino, volta a treinar e atualiza o teu projeto Scratch.

## UNIDADE 3.6 EXTENSÃO DE IA: TEXTO PARA FALA

A síntese de texto para fala (TTS) é uma tecnologia suportada por IA que converte texto escrito em áudio falado. Esta extensão oficial do Scratch usa síntese de fala baseada na nuvem para dar aos teus sprites vozes realistas em vários idiomas e tons.
O TTS torna os projetos mais acessíveis (utilizadores com deficiência visual podem ouvir o conteúdo), mais envolventes (diálogos falados são mais dinâmicos do que balões de texto) e mais versáteis (os projetos podem narrar histórias, dar instruções ou criar ferramentas de aprendizagem de línguas).

### ADICIONAR A EXTENSÃO DE TEXTO-PARA-FALA

**Passo 1:** Abrir a Biblioteca de Extensões. No editor do Scratch, clica no botão azul "Adicionar Extensão" (com o ícone "+") no canto inferior esquerdo da paleta de blocos.

**Passo 2:** Percorre a biblioteca de extensões e clica em "Text-to-Speech" (ícone de um altifalante com balões de fala). Novos blocos vão aparecer na tua paleta, numa nova categoria "Text-to-Speech".

**Passo 3:** Usar os blocos no teu código:

**Falar uma frase:** Arrasta o bloco speak [hello] para a área de código e escreve o texto que queres que o sprite diga em voz alta. O bloco converte o texto em fala e reproduz o áudio.

**Mudar a voz:** Usa o bloco set voice to [alto] para escolher outra voz. As opções são Alto, Tenor, Squeak, Giant e Kitten.

**Alterar o idioma:** Para alterar o idioma, utilize o bloco definir idioma.


Figura 5 – Sequência de blocos de código para uma conversa no Scratch: Scratch do MIT Media Lab

## UTILIZAÇÃO DOS BLOCOS DO SCRATCH PARA CRIAR PROJETOS INTERATIVOS

Combine os blocos de Texto para Voz com outros blocos do Scratch para criar projetos interativos. Por exemplo, pode utilizar o bloco de evento "quando a bandeira verde for clicada", da categoria "Eventos", para iniciar a fala. Pode também utilizar o bloco "perguntar e esperar", da categoria "Sensores", juntamente com o bloco "resposta" e o bloco "falar", para que o seu sprite repita o que o utilizador escreve.

## PRINCIPAIS BLOCOS DE TEXTO PARA VOZ

Esta extensão disponibiliza blocos para controlar a fala, conforme apresentado abaixo:

| Bloco | Função | Exemplo |
|---|---|---|
| falar [texto] | Lê em voz alta o texto especificado | falar [Olá, bem-vindo ao meu projeto!] |
| definir voz para [voz] | Altera o tipo de voz | definir voz para [tenor] (opções: alto, tenor) |
| definir idioma para [idioma] | Altera o idioma da fala | definir idioma para [Espanhol] (23 idiomas suportados) |

## OPÇÕES DE VOZ – TEXTO PARA VOZ

Existem várias opções de voz de texto para voz que pode experimentar para adequar à personalidade do seu sprite; são as seguintes:

- **Alto**: Voz de tom médio, neutra
- **Tenor**: Voz ligeiramente mais aguda
- **Squeak:** Voz aguda, semelhante à de uma criança
- **Giant**: Voz grave e forte
- **Kitten**: Voz muito aguda e suave

## EXEMPLOS DE UTILIZAÇÃO DO TEXTO PARA VOZ NO SCRATCH

- **Histórias Interativas:** As personagens narram o desenvolvimento da história e dizem os diálogos.
- **Questionários Educativos:** O computador lê as perguntas em voz alta para garantir acessibilidade.
- **Aprendizagem de Línguas:** Prática de pronúncia em diferentes idiomas.
- **Assistentes Virtuais:** Assistentes como a Siri que respondem por voz.
- **Guias Áudio:** Visitas a museus ou guias turísticos com informação narrada.
- **Feedback Dinâmico:** Incentivos ou sugestões em jogos transmitidos por voz.

# REFERÊNCIAS

## Livro, Wiki e Documentação


- Scratch Foundation (MIT Media Lab): Documentação da plataforma, recursos educativos e descrições da interface. https://scratch.mit.edu/

- Machine Learning for Kids (Dale Lane, IBM), 11 fev. 2021: Tutoriais da plataforma, metodologia de treino de modelos e guias de integração. ISBN-13: 9781718500563

- Scratch Wiki: Documentação técnica sobre extensões e descrição de blocos. https://scratch-wiki.info/

- The AI Scratch Code Playbook: A Beginner's Guide to Building Intelligent Systems. Edição em capa mole – 11 fev. 2025. ISBN-13: 979-8310433649


## LIGAÇÕES PARA PLATAFORMAS DE APRENDIZAGEM

**Scratch – Website Oficial**
A plataforma oficial Scratch desenvolvida pelo MIT Media Lab.
Permite criar uma conta, desenvolver projetos, explorar milhões de criações da comunidade e aceder a tutoriais e recursos de aprendizagem.
https://scratch.mit.edu/

**Machine Learning for Kids**
Plataforma educativa da IBM para treinar modelos personalizados de aprendizagem automática (texto, imagens, números, sons) e integrá-los no Scratch.
Inclui fichas de trabalho, tutoriais e modelos pré-treinados.
https://machinelearningforkids.co.uk/

## OPÇÕES DE VOZ – TEXTO PARA VOZ

**Ensinar Conceitos de IA de Forma Criativa com Scratch (Codingal)**
Artigo de blogue que explora abordagens criativas para ensinar conceitos de IA através do Scratch, incluindo ideias de projetos e estratégias pedagógicas.
https://www.codingal.com/coding-for-kids/blog/teach-kids-ai-codng-concepts-creatively-using-scratch/

**Scratch Machine Learning Studio**
Coleção curada de projetos Scratch que utilizam extensões de aprendizagem automática. Explore para obter inspiração e remisturar projetos já existentes.
https://scratch.mit.edu/studios/3995548/

## EXERCÍCIO PRÁTICO

**Objetivo:** Criar um projeto utilizando a extensão Texto para Voz, experimentando diferentes vozes para adequar à personalidade dos seus sprites.

**Instruções:** Siga os passos abaixo para criar um sprite que leia em voz alta o texto introduzido pelo utilizador, demonstrando o potencial da combinação entre entrada de texto e saída por voz.

**Passo 1 – Configurar o Projeto:**
Crie um novo projeto no Scratch ou continue com um já existente.
Escolha um sprite que faça a fala — os sprites "Robot" ou "Wizard" funcionam bem.

**Passo 2 – Adicionar a Extensão:**
Adicione a extensão Texto para Voz conforme descrito anteriormente.

**Passo 3 – Criar o Script e Construir o Código para o seu Sprite:**
quando a bandeira verde for clicada
└── definir voz para [giant]
└── falar [Olá! Posso dizer tudo o que escreveres.]
└── para sempre
└── perguntar [O que devo dizer?] e esperar
└── falar (resposta)

**Passo 4 – Testar o Projeto:**
Clique na bandeira verde e escreva mensagens quando solicitado. O seu sprite irá ler as mensagens em voz alta. Experimente diferentes vozes e idiomas, alterando os blocos definir voz e definir idioma.

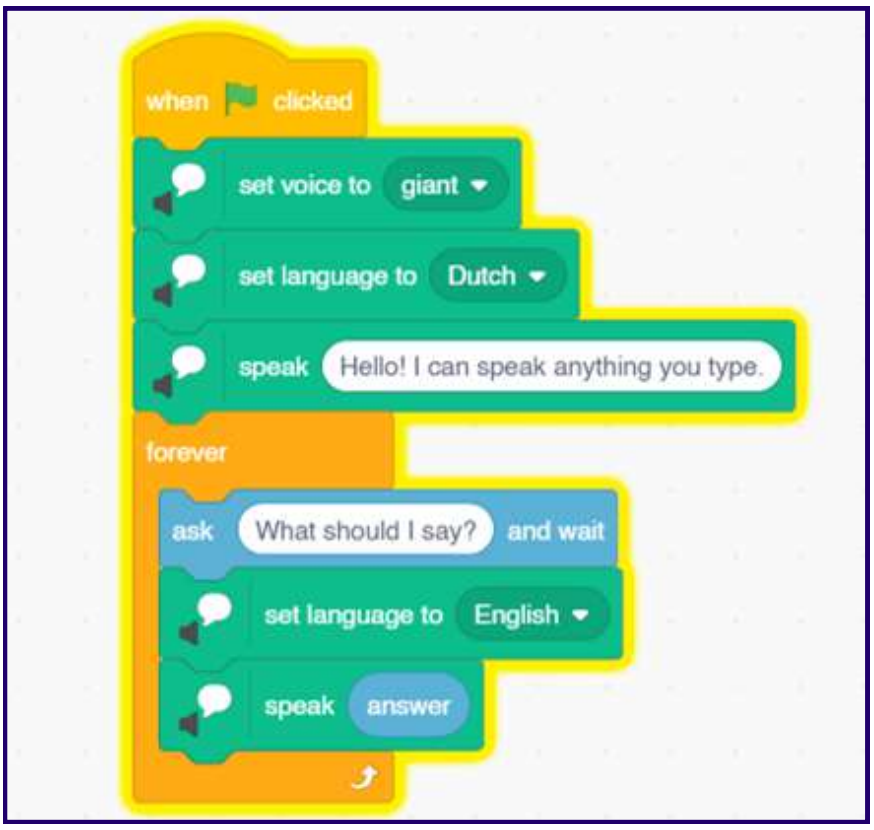



Figura 6 – Sequência de blocos de código no Scratch: Scratch do MIT Media Lab

## PERGUNTAS DE REFLEXÃO

Reserve algum tempo para refletir profundamente sobre o que aprendeu neste módulo.
Responda às seguintes questões de forma fundamentada:

- **Pergunta de Reflexão 1:** Que aspeto da utilização da IA no Scratch foi mais surpreendente para si? Algo funcionou de forma diferente do que esperava? Explique com exemplos concretos do seu projeto.

- **Pergunta de Reflexão 2:** Como utilizaria a aprendizagem automática num jogo ou numa história interativa? Descreva uma ideia específica de projeto e explique de que forma a IA o tornaria mais envolvente ou inteligente do que a programação tradicional isoladamente.

- **Pergunta de Reflexão 3:** Que problemas ou desafios enfrentou ao treinar o seu modelo de aprendizagem automática ou ao integrar extensões de IA? Como os resolveu? Caso tenha encontrado problemas que não conseguiu resolver, que estratégias tentaria a seguir?


## QUESTIONÁRIO DE ESCOLHA MÚLTIPLA (uma resposta correta por pergunta)

**1. Qual das seguintes afirmações melhor descreve o Scratch?**
a) Uma linguagem de programação textual utilizada por engenheiros de software profissionais
b) Uma linguagem de programação visual baseada em blocos que torna a programação acessível a iniciantes
c) Um serviço de inteligência artificial para criar modelos de aprendizagem automática
d) Um kit de robótica para ensinar programação a crianças

**2. Qual é um dos principais benefícios da utilização da programação baseada em blocos no Scratch?**
a) Exige que os alunos memorizem sintaxe, reforçando as competências de memória
b) Os blocos apenas se encaixam de forma logicamente correta, evitando erros de sintaxe
c) Foca-se exclusivamente em código textual, sendo mais exigente para os alunos
d) Só pode ser utilizado por pessoas com experiência prévia em programação

**3. Que categoria de blocos do Scratch utilizaria para criar um ciclo que repete código 10 vezes?**
a) Movimento
b) Eventos
c) Controlo
d) Operadores

**4. O que deve fazer antes de poder utilizar um modelo do Machine Learning for Kids no seu projeto Scratch?**
a) Nada — a extensão já vem pré-treinada com um modelo que funciona para todos os fins
b) Fornecer exemplos de treino para cada categoria que pretende que o modelo reconheça e, em seguida, treinar o modelo
c) Adquirir um dispositivo de hardware especial para ativar a aprendizagem automática
d) Escrever código em Python para criar um algoritmo de aprendizagem automática e importá-lo para o Scratch


## QUESTIONÁRIO DE ESCOLHA MÚLTIPLA
## (uma resposta correta por pergunta)

**5. Qual das seguintes opções NÃO é uma funcionalidade relacionada com IA mencionada neste módulo?**

a) Reconhecer objetos ou imagens através de classificação de imagens
b) Fazer um sprite executar uma rotina de dança utilizando um ciclo de repetição
c) Converter palavras faladas em texto através de reconhecimento de voz
d) Traduzir texto de inglês para espanhol dentro de um projeto

**6. Que extensão do Scratch utilizaria para que um sprite narrasse uma história em voz alta?**

a) Deteção de Vídeo
b) Traduzir
c) Texto para Voz
d) Machine Learning for Kids

**7. Porque é importante fornecer múltiplos exemplos de treino ao criar um modelo de aprendizagem automática?**

a) O modelo não consegue funcionar com menos de 100 exemplos
b) Mais exemplos ajudam o modelo a aprender padrões com maior precisão e a classificar novas entradas de forma mais fiável
c) Os exemplos de treino são apenas decorativos e não afetam o desempenho
d) Muitos exemplos tornam o modelo mais lento, tornando-o mais preciso

**8. Que vantagem resulta da combinação do Machine Learning for Kids com a extensão Texto para Voz?**

a) Torna o projeto mais rápido
b) Permite compreender entradas e comunicar respostas
c) Reduz a quantidade de código necessário
d) Permite que o projeto funcione offline, sem internet

# Módulo 4: Programação de Jogos em Python (bibliotecas pygame / Arcade) com IA

## INTRODUÇÃO

Neste módulo breve e acessível a iniciantes, irá tornar-se criador de jogos, e não apenas jogador. Irá aprender a conceber e programar um pequeno videojogo utilizando Python e Pygame, e também descobrir como fazer com que as personagens do seu jogo se comportem de forma inteligente através de formas simples de Inteligência Artificial (IA). Não se preocupe se nunca programou antes — tudo será explicado passo a passo. Começará por criar uma pequena janela de jogo e mover um quadrado no ecrã. Posteriormente, irá explorar como fazer com que o jogo reaja às suas ações, por exemplo, fazendo com que um inimigo imite os seus movimentos ou mude de cor quando se aproxima. Este módulo não se centra apenas na execução de código, mas também na compreensão e experimentação. Irá testar e modificar exemplos para perceber como pequenas alterações podem criar comportamentos diferentes. Ao fazê-lo, compreenderá como os jogos "pensam" e respondem, de forma semelhante a sistemas simples de IA no mundo real. No final do módulo, será capaz de adquirir competências como:

- Criar jogos 2D interativos utilizando as bibliotecas Pygame ou Arcade.
- Integrar comportamentos de IA para que as personagens do jogo reajam às ações do jogador.
- Utilizar o pensamento computacional para conceber jogos interativos e responsivos.
- Experimentar, testar e modificar código para observar como pequenas alterações influenciam o comportamento do jogo.
- Refletir sobre a forma como a IA é utilizada na vida real.

| | |
|---|---|
| **Duração do Módulo** | **2 horas (1 hora de aprendizagem + 1 hora de exercícios práticos)** |

## MATERIAIS EDUCATIVOS

### UNIDADE 4.1 INTRODUÇÃO AO PYGAME

Antes de começar a programar, é importante saber com que ferramentas irá trabalhar e para que servem:

- **Python 3** – a linguagem de programação que irá utilizar para dar instruções ao computador. É amplamente utilizada em escolas, universidades e empresas para programação, análise de dados e IA.

- **Pygame** – uma biblioteca gratuita de Python que o ajuda a criar jogos 2D simples. Disponibiliza funções prontas a usar para abrir uma janela de jogo, desenhar formas, mostrar cores e reagir ao teclado ou ao rato.

- **Editor de código (IDLE, Thonny ou VS Code)** – o programa onde irá escrever, guardar e executar o seu código em Python. Neste módulo, pode utilizar o IDLE (incluído no Python) ou outro editor recomendado pelo seu professor.

Estas ferramentas serão apresentadas passo a passo, para que as possa instalar e realizar as atividades de forma autónoma, mesmo que este seja o seu primeiro contacto com a programação.

**Objetivo:** Aprender a abrir uma janela de jogo, mostrar cores e mover um quadrado com o teclado.

**Passo 1 – Instalar o Python 3 e abrir o seu ambiente de programação**
Antes de começar a utilizar o Pygame, deve certificar-se de que o Python 3 está instalado no seu computador.

**Instalar o Python 3**
- Aceda ao website oficial do Python: https://www.python.org/downloads/.
- Faça o download da versão adequada ao seu sistema (Windows, macOS ou Linux).
- Durante a instalação, selecione a opção “Add Python to PATH” antes de clicar em “Install Now”.
- Após a instalação, reinicie o computador (opcional, mas recomendado).

## UNIDADE 4.2 VERIFICAR A INSTALAÇÃO

Abra a Linha de Comandos (Windows) ou o Terminal (macOS/Linux) e escreva:

**python --version** ou **py --version**

Se for apresentada uma mensagem como, por exemplo, Python 3.13.3, significa que está tudo pronto.

## UNIDADE 4.3 INSTALAR UM EDITOR OU UTILIZAR O IDLE

Pode escrever código em Python utilizando uma das seguintes opções:

- IDLE (instalado juntamente com o Python)
- Thonny (IDLE simples para iniciantes) – https://thonny.org3.
- VS Code (opção avançada) – https://code.visualstudio.com

Para abrir o IDLE:

- No Windows: clique em Iniciar → IDLE (Python 3.x)
- No macOS: procure por “IDLE” no Launchpad
- No Linux: execute idel3 no terminal

### 3. Escrever ou Colar o Código

Copie o código deste módulo para o novo ficheiro.
Por exemplo:

**CÓDIGO:**

```
import pygame
pygame.init()
ecra = pygame.display.set_mode((800, 600))
pygame.display.set_caption("O Meu Primeiro Jogo")
jogo_ativo = True
while jogo_ativo:
    for evento in pygame.event.get():
        if evento.type == pygame.QUIT:
            jogo_ativo = False
    ecra.fill((200, 220, 255))
    pygame.display.flip()
pygame.quit()
```

### 4. Guardar o Ficheiro

Selecione Ficheiro → Guardar Como...
Dê um nome ao ficheiro (por exemplo, meu_primeiro_jogo.py).

### 5. Executar o Programa

Prima F5 ou vá a Executar → Executar Módulo.
Será aberta uma nova janela — se vir um fundo azul, significa que está tudo a funcionar corretamente.

## UNIDADE 4.4 COMPETÊNCIAS BÁSICAS DE PROGRAMAÇÃO BASEADA EM BLOCOS

Copie e execute o seguinte código:

**CÓDIGO:**

```
import pygame  # 1. Importar a biblioteca pygame
pygame.init()  # 2. Iniciar (inicializar) todos os módulos do
pygame

ecra = pygame.display.set_mode((800, 600))  # 3. Criar uma
janela de jogo (largura 800, altura 600)
pygame.display.set_caption("O Meu Primeiro Jogo")  # 4.
Definir um título para a janela do jogo

jogo_ativo = True  # 5. Criar uma variável para manter o jogo
em execução

while jogo_ativo:  # 6. Iniciar o ciclo principal do jogo
(executa repetidamente)
   for evento in pygame.event.get():  # 7. Verificar todos os
eventos (teclas premidas, cliques do rato, etc.)
       if evento.type == pygame.QUIT:  # 8. Se o utilizador
clicar no botão "fechar"…
           jogo_ativo = False  # 9. …parar o ciclo do jogo

   ecra.fill((200, 220, 255))  # 10. Preencher o ecrã com uma
cor azul-claro (valores RGB)
   pygame.display.flip()  # 11. Atualizar a janela para
mostrar a nova imagem

pygame.quit()  # 12. Fechar o jogo e sair em segurança
```

## UNIDADE 4.5 MOVER UM QUADRADO

Agora vamos fazer algo mover-se. Copie o código abaixo e teste-o:

**PARTE I**

**CÓDIGO:**

```
import pygame  # 1. Importar a biblioteca pygame
pygame.init()  # 2. Iniciar o pygame

# 3. Criar a janela
ecra = pygame.display.set_mode((800, 600))

# 4. Adicionar um título
pygame.display.set_caption("Quadrado em Movimento")

posicao_x = 400  # 5. Posição X do quadrado (horizontal)
posicao_y = 300  # 6. Posição Y do quadrado (vertical)
velocidade = 1  # 7. Quantos píxeis se move de cada vez

jogo_ativo = True  # 8. Manter o jogo em execução

while jogo_ativo:  # 9. Iniciar o ciclo do jogo
    for evento in pygame.event.get():  # 10. Verificar
eventos
        if evento.type == pygame.QUIT:  # 11. Se o utilizador
clicar no "X", parar
            jogo_ativo = False
```

## UNIDADE 4.6 MOVER UM QUADRADO

Agora vamos fazer algo mover-se. Copie o código abaixo e teste-o:

**PARTE II**

**CÓDIGO:**

```
    teclas = pygame.key.get_pressed()  # 12. Verificar quais as
teclas premidas

    if teclas[pygame.K_LEFT]:
        posicao_x -= velocidade  # 13. Mover para a esquerda
    if teclas[pygame.K_RIGHT]:
        posicao_x += velocidade  # 14. Mover para a direita
    if teclas[pygame.K_UP]:
        posicao_y -= velocidade  # 15. Mover para cima
    if teclas[pygame.K_DOWN]:
        posicao_y += velocidade  # 16. Mover para baixo

    ecra.fill((200, 220, 255))  # 17. Pintar o fundo
    pygame.draw.rect(ecra, (0, 0, 255), (posicao_x, posicao_y,
50, 50))  # 18. Desenhar o quadrado azul
    pygame.display.flip()  # 19. Atualizar o ecrã

pygame.quit()  # 20. Fechar o jogo
```

**O que está a acontecer:**

- posicao_x e posicao_y definem a posição do quadrado no ecrã.
- As teclas de seta alteram esses valores, deslocando o quadrado.
- O ciclo repete o desenho muitas vezes por segundo, criando um movimento suave.

**Experimenta isto:**

- Altera velocidade = 1 para velocidade = 5. O que acontece?
- Faz o quadrado ter uma cor diferente, por exemplo (255, 0, 0) para vermelho.
- O que achas que aconteceria se removesses pygame.display.flip()?

**Pergunta de Reflexão**: Como é que o computador sabe a nova posição do quadrado sempre que carregas numa tecla?

**Ponto de Verificação da Aprendizagem**
Nesta fase, já deverás ser capaz de:

✓Abrir uma janela no Pygame.
✓Utilizar o ciclo do jogo para o manter em execução.
✓Controlar um objeto em movimento com o teclado.

Na próxima parte, vais acrescentar Inteligência Artificial para que o jogo reaja aos teus movimentos.

## UNIDADE 4.7 IA SIMPLES EM JOGOS (INIMIGO REATIVO)

Neste módulo, utilizamos uma IA simples baseada em regras: o computador segue regras predefinidas para detetar o jogador, decidir o que fazer e agir em conformidade, sem aprender a partir de dados. Isto é semelhante a muitos videojogos clássicos, em que os inimigos perseguem, evitam ou reagem ao jogador com base na distância ou na posição.

**Objetivo:** Aprender a fazer com que o seu jogo reaja às ações do jogador, adicionando uma forma simples de Inteligência Artificial (IA). Irá criar um inimigo que muda de cor quando o jogador se aproxima.

**Passo 1 – Compreender a Ideia**

Antes de programar, pense no que significa IA num jogo. Neste caso, a IA não está relacionada com aprendizagem ou dados — trata-se de fazer o computador reagir ao que o jogador faz.

**Irá utilizar uma regra simples:** Se o jogador estiver perto, o inimigo muda de cor. Caso contrário, mantém-se igual. Este é um exemplo de comportamento condicional, uma das formas mais simples de IA.

**Passo 2 – Exemplo de Código: Inimigo Reativo**

Copie e execute o seguinte código:

**PARTE I**

**CÓDIGO:**

```
import pygame, math  # 1. Importar as bibliotecas pygame e math
pygame.init()        # 2. Iniciar (inicializar) todas as funções do
Pygame

ecra = pygame.display.set_mode((800, 600))  # 3. Criar uma janela
de jogo (800x600)
pygame.display.set_caption("IA Simples – Inimigo Reativo")  # 4.
Título da janela

jogador = pygame.Rect(100, 100, 50, 50)  # 5. Jogador (quadrado
azul)
inimigo = pygame.Rect(500, 300, 50, 50)  # 6. Inimigo (quadrado
vermelho)

velocidade = 5           # 7. Velocidade do jogador
distancia_reacao = 120   # 8. Distância (em píxeis) para o inimigo
"reagir"

relogio = pygame.time.Clock()
jogo_ativo = True        # 9. Manter o jogo em execução
```

**PARTE II**

**CÓDIGO:**

```
while jogo_ativo:  # 10. Ciclo principal do jogo (a cada frame)

    for evento in pygame.event.get():  # 11. Verificar todos os eventos
        if evento.type == pygame.QUIT:  # 12. Se a janela for fechada...
            jogo_ativo = False  # 13. ...parar o ciclo

    # ---- MOVIMENTO (deve estar dentro do while) ----
    teclas = pygame.key.get_pressed()  # 14. Verificar quais as teclas
premidas

    if teclas[pygame.K_LEFT]:
        jogador.x -= velocidade  # 15. Mover para a esquerda
    if teclas[pygame.K_RIGHT]:
        jogador.x += velocidade  # 16. Mover para a direita
    if teclas[pygame.K_UP]:
        jogador.y -= velocidade  # 17. Mover para cima
    if teclas[pygame.K_DOWN]:
        jogador.y += velocidade  # 18. Mover para baixo

    # ---- IA SIMPLES: mudar de cor se estiver perto ----
    centro_jogador_x, centro_jogador_y = jogador.center  # 19. Centro do
jogador
    centro_inimigo_x, centro_inimigo_y = inimigo.center  # 20. Centro do
inimigo

    distancia = math.hypot(centro_jogador_x - centro_inimigo_x,
                           centro_jogador_y - centro_inimigo_y)  # 21.
Distância

    if distancia < distancia_reacao:  # 22. Se estiver a menos de 120 px...
        cor_inimigo = (255, 200, 0)  # 23. ...o inimigo fica amarelo
    else:
        cor_inimigo = (220, 60, 60)  # 24. Caso contrário, mantém-se
vermelho
```

**PARTE III**

**CÓDIGO:**

```
# ---- DESENHAR ----
ecra.fill((240, 245, 255))  # 25. Fundo
pygame.draw.rect(ecra, (60, 120, 255), jogador)  # 26. Jogador (azul)
pygame.draw.rect(ecra, cor_inimigo, inimigo)  # 27. Inimigo (reativo)
pygame.display.flip()  # 28. Mostrar alterações no ecrã
relogio.tick(60)  # Limitar a 60 FPS
pygame.quit()  # 29. Fechar em segurança
```

**Passo 3 – O que está a acontecer neste código**

| Conceito | Explicação |
|---|---|
| Perceção | O inimigo mede a distância até ao jogador utilizando math.hypot. |
| Decisão | Se a distância for inferior a distancia_reacao, o inimigo decide reagir. |
| Ação | A reação consiste em mudar a sua cor. |

Este ciclo "perceber → decidir → agir" é a base da maioria dos sistemas de IA em jogos. Mesmo reações simples como esta tornam o mundo do jogo mais dinâmico e realista.

## PASSO 4 – EXPERIMENTAR, ALTERAR, OBSERVAR

Experimente alterar estas partes do código e observe o que acontece:

1. Altere distancia_reacao de 120 para 250 — o que acontece?
2. Faça o inimigo ficar verde (0, 255, 0) em vez de amarelo.
3. Faça o jogador mover-se mais depressa, aumentando velocidade = 8.
4. Consegue fazer com que o inimigo reaja apenas quando o jogador se move acima ou abaixo dele (por exemplo, comparando apenas as posições em y)?

Incentive os alunos a prever antes de executar o programa — o que acham que vai acontecer?

**Passo 5 – Reflexão**
Isto é mesmo "inteligência"?
Porquê (ou porquê não)?
Como poderíamos fazer com que o comportamento do inimigo parecesse mais inteligente?

**Dica:** A IA real em jogos usa a mesma lógica — decisões simples repetidas muitas vezes por segundo podem criar a ilusão de inteligência.

# REFERÊNCIAS

## Livro, Wiki e Documentação


- Python Software Foundation. Documentação do Python 3. Disponível em: https://docs.python.org
- Comunidade Pygame. Documentação e tutoriais do Pygame. Disponível em: https://www.pygame.org/docs/ e https://www.pygame.org/wiki/tutorials
- Projeto Python Arcade. Documentação da Biblioteca Python Arcade. Disponível em: https://api.arcade.academy
- Real Python. Pygame: Introdução à Programação de Jogos em Python. Disponível em: https://realpython.com/pygame-a-primer/
- Microsoft. AI for Beginners – Curriculum. Disponível em: https://github.com/microsoft/AI-For-Beginners


## LIGAÇÕES PARA PLATAFORMAS DE APRENDIZAGEM

- Website Oficial do Python – Descarregar e aprender os fundamentos da programação em Python. (https://www.python.org/)
- Documentação do Pygame – Guia oficial de funções, eventos e gráficos no Pygame. (https://www.pygame.org/docs/)
- Documentação da Biblioteca Arcade – Biblioteca alternativa para jogos 2D simples em Python. (https://api.arcade.academy/en/latest/)
- Pygame AI Guide – Exemplos de algoritmos simples de IA para Pygame. (https://pygame-ai.readthedocs.io/en/latest/guide.html)
- Real Python – Criar um Jogo com Pygame – Tutorial acessível para iniciantes. (https://realpython.com/pygame-a-primer/)

- KidsCanCode Tutorials – Projetos e desafios simples para aprender Pygame. (https://kidscancode.org/blog/)

- AI for Beginners (Microsoft) – Conteúdos educativos abertos que introduzem os princípios da Inteligência Artificial. (https://microsoft.github.io/AI-For-Beginners/)

## EXERCÍCIO PRÁTICO

### APANHA-ME SE CONSEGUIRES

**Objetivo:** Criar um pequeno jogo onde o jogador move um quadrado azul para recolher objetos, enquanto um inimigo vermelho com IA segue o jogador. Este exercício reúne todas as competências aprendidas: movimento, deteção de colisões e um comportamento simples de IA.

**Passo 1 – Preparar o Ambiente**
Certifique-se de que o Pygame está instalado:

**pip install pygame**

Depois, abra o seu editor de Python e crie um novo ficheiro chamado apanha_me.py.

**Passo 2 – Compreender a Ideia do Jogo**

| Elemento | Descrição |
|---|---|
| Jogador | Move-se com as teclas de seta. |
| Jogador | Surge aleatoriamente; aumenta a pontuação quando é recolhido. |
| Inimigo (IA) | Segue o jogador calculando direção e distância. |
| Objetivo | Recolher o maior número possível de alvos antes que o inimigo o apanhe. |

Como poderá o inimigo “saber” onde está o jogador?

## PASSO 3 – CONSTRUIR O JOGO PASSO A PASSO

**PARTE I**

**CÓDIGO:**

```
import pygame, random, math
pygame.init()

# --- Janela ---
LARGURA, ALTURA = 800, 600
ecra = pygame.display.set_mode((LARGURA, ALTURA))
pygame.display.set_caption("Apanha-me se Conseguires")
relogio = pygame.time.Clock()

# --- Cores ---
AZUL = (60, 120, 255)
VERMELHO = (220, 60, 60)
VERDE = (60, 200, 60)
BRANCO = (240, 245, 255)
PRETO = (20, 20, 20)

# --- Entidades ---
jogador = pygame.Rect(100, 100, 40, 40)  # quadrado azul
(jogador)
inimigo = pygame.Rect(600, 400, 40, 40)  # quadrado vermelho
(inimigo perseguidor)

alvo = pygame.Rect(
    random.randint(50, LARGURA - 50),  # alvo verde: posição
aleatória
    random.randint(50, ALTURA - 50),
    25, 25
)
```

## PASSO 3 – CONSTRUIR O JOGO PASSO A PASSO

**PARTE II**

**CÓDIGO:**

```
VELOCIDADE_JOGADOR = 5
VELOCIDADE_INIMIGO = 2.5
pontuacao = 0

# Tipo de letra (compatível: usa a fonte padrão para evitar
problemas)
fonte = pygame.font.Font(None, 26)

# --- Ciclo do jogo ---
jogo_ativo = True
while jogo_ativo:

    # 1) Tratar eventos
    for evento in pygame.event.get():
        if evento.type == pygame.QUIT:
            jogo_ativo = False

    # 2) Movimento do jogador (teclas de seta)
    teclas = pygame.key.get_pressed()

    if teclas[pygame.K_LEFT]:
        jogador.x -= VELOCIDADE_JOGADOR
    if teclas[pygame.K_RIGHT]:
        jogador.x += VELOCIDADE_JOGADOR
    if teclas[pygame.K_UP]:
        jogador.y -= VELOCIDADE_JOGADOR
    if teclas[pygame.K_DOWN]:
        jogador.y += VELOCIDADE_JOGADOR
```

## PASSO 3 – CONSTRUIR O JOGO PASSO A PASSO

**PARTE III**

**CÓDIGO:**

```
# Manter o jogador dentro dos limites da janela
jogador.clamp_ip(ecra.get_rect())

# 3) IA simples: o inimigo persegue o jogador
diferenca_x = jogador.centerx - inimigo.centerx
diferenca_y = jogador.centery - inimigo.centery
distancia = math.hypot(diferenca_x, diferenca_y)

if distancia != 0:  # evitar divisão por zero quando estão
sobrepostos
    inimigo.x += int((diferenca_x / distancia) *
VELOCIDADE_INIMIGO)
    inimigo.y += int((diferenca_y / distancia) *
VELOCIDADE_INIMIGO)

# 4) O jogador apanha o alvo
if jogador.colliderect(alvo):
    pontuacao += 1
    alvo.topleft = (
        random.randint(25, LARGURA - 50),
        random.randint(25, ALTURA - 50)
    )

# 5) Desenhar tudo
ecra.fill(BRANCO)
pygame.draw.rect(ecra, AZUL, jogador)
pygame.draw.rect(ecra, VERMELHO, inimigo)
pygame.draw.rect(ecra, VERDE, alvo)
```

## PASSO 3 – CONSTRUIR O JOGO PASSO A PASSO

**PARTE IV**

**CÓDIGO:**

```
# 6) Mostrar pontuação no ecrã
texto_pontuacao = fonte.render(f"Pontuação: {pontuacao}", True,
PRETO)
ecra.blit(texto_pontuacao, (10, 10))

pygame.display.flip()    # Atualizar o ecrã
relogio.tick(60)         # Limitar a 60 FPS

pygame.quit()            # Fechar o jogo
```

### Passo 4 – Experimentar e Refletir

Experimente alterar estes valores e observe o que acontece:

- Altere VELOCIDADE_INIMIGO para tornar o inimigo mais rápido ou mais lento.
- Modifique a cor do jogador ou do alvo.
- Torne o jogo mais difícil reduzindo o tamanho do jogador.
- Adicione uma mensagem de “Fim de Jogo” quando o inimigo tocar no jogador

## PERGUNTAS DE REFLEXÃO

Reserve algum tempo para refletir profundamente sobre o que aprendeu neste módulo. Responda às seguintes questões de forma fundamentada:

- **Pergunta de Reflexão 1:** Qual foi a parte mais difícil na criação do seu jogo?
- **Pergunta de Reflexão 2:** De que forma a IA torna o jogo mais interativo?
- **Pergunta de Reflexão 3:** Consegue identificar outras situações em que a IA reage às ações humanas (por exemplo, na educação ou na saúde)?
- **Pergunta de Reflexão 4:** Se pudesse melhorar este jogo, que tipo de comportamento de IA acrescentaria?
- **Pergunta de Reflexão 5:** O que faz com que o inimigo pareça "inteligente"?
- **Pergunta de Reflexão 6:** Como é que o cálculo da distância (math.hypot) ajuda o inimigo a encontrar o jogador?
- **Pergunta de Reflexão 7:** Consegue identificar jogos reais que utilizem uma lógica semelhante?


## QUESTIONÁRIO DE ESCOLHA MÚLTIPLA
## (uma resposta correta por pergunta)

**1. O que faz o ciclo do jogo (game loop) no Pygame?**
a) Instala novas bibliotecas durante o jogo
b) Repete as ações do jogo muitas vezes por segundo
c) Fecha a janela do jogo
d) Carrega a imagem de fundo apenas uma vez

**2. Qual é a função de pygame.display.flip()?**
a) Verifica a entrada do teclado
b) Desenha um retângulo
c) Atualiza a janela do jogo com os novos elementos visuais
d) Pausa o ciclo do jogo

**3. Que função verifica quais as teclas do teclado que estão a ser premidas?**
a) pygame.quit()
b) pygame.key.get_pressed()
c) pygame.event.get()
d) pygame.display.set_mode()

**4. No exemplo de IA, o que faz o inimigo reagir ao jogador?**
a) Um evento temporizador
b) Movimento aleatório
c) A distância entre o jogador e o inimigo
d) A cor do jogador

**5. O que acontece se aumentar o valor de distancia_reacao no código?**
a) O inimigo reage a partir de mais longe
b) O inimigo move-se mais depressa
c) O jogador abranda
d) O tamanho da janela altera-se


## QUESTIONÁRIO DE ESCOLHA MÚLTIPLA
## (uma resposta correta por pergunta)

**6. Qual é a finalidade da função math.hypot() no código de IA?**
a) Calcular a distância entre dois pontos
b) Desenhar o inimigo no ecrã
c) Criar uma nova posição aleatória
d) Detetar colisões com paredes

**7. O que faz esta linha de código?**
a) Desenha um texto no ecrã
b) Cria um retângulo para representar o jogador
c) Inicia o algoritmo de IA
d) Altera a cor de fundo

**8. O que acontece se remover pygame.display.flip() do ciclo?**
a) O programa executa mais rapidamente
b) A janela não é atualizada e parece bloqueada
c) As cores mudam automaticamente
d) O jogador move-se duas vezes mais depressa

**9. Qual das seguintes opções é um exemplo de comportamento de IA num jogo?**
a) Desenhar um quadrado no ecrã
b) Fazer uma personagem mover-se apenas quando o jogador carrega numa tecla
c) Fazer um inimigo seguir ou evitar o jogador
d) Carregar música de fundo

**10. Como pode a IA tornar os jogos mais interessantes?**
a) Corrigindo automaticamente erros do programa
b) Criando desafios dinâmicos e personagens que reagem ao jogador
c) Alterando automaticamente os gráficos
d) Controlando o teclado

# Módulo 5: Reconhecimento de Objetos com Roboflow – Introdução à Visão Computacional

## INTRODUÇÃO

O objetivo deste módulo é fornecer uma introdução abrangente à Visão Computacional, explicando como as máquinas percecionam e processam dados visuais, e orientando os formandos na criação dos seus próprios modelos de reconhecimento de objetos utilizando a plataforma Roboflow e algoritmos avançados como o YOLO.
No final do módulo, será capaz de adquirir competências como:

- **Compreender os Fundamentos:** Distinguish between image processing and computer vision, and understand the logic of how computers interpret images as data (pixels and binary code).
- **Aplicar o Processo de Visão Computacional:** Apply the five essential steps of the computer vision process: obtaining data, preparing data, determining the AI model, training the model, and interpreting results.
- **Utilizar o Roboflow:** Navigate the Roboflow platform to create projects, upload datasets, and manage image data effectively.
- **Anotação de Dados:** Realizar anotação (rotulagem) precisa para preparar conjuntos de dados destinados ao treino de modelos de IA.
- **Treino de Modelos:** Selecionar e treinar modelos avançados de deteção de objetos, como o YOLOv11, para identificar objetos específicos.

| | |
|---|---|
| Duração do Módulo | 2 horas (1 hora de aprendizagem + 1 hora de exercícios práticos) |

## MATERIAIS EDUCATIVOS

### UNIDADE 5.1 O QUE É O PROCESSAMENTO DE VISÃO COMPUTACIONAL?

O principal objetivo do processamento de imagem é extrair significado das imagens e disponibilizar essa informação para utilização em diferentes áreas. Embora as tecnologias de processamento visual existam há muito tempo, anteriormente este processo dependia fortemente da intervenção humana. Tal tornava-o moroso e propenso a erros. Por exemplo, nos primeiros sistemas de reconhecimento facial, os programadores tinham de rotular manualmente milhares de fotografias, identificando características específicas, como a largura do nariz ou a distância entre os olhos. Isto acontecia porque os dados das imagens estavam frequentemente desorganizados e eram difíceis de interpretar pelos computadores. Assim, a automatização do processo exigiu maior poder de processamento e técnicas computacionais avançadas.

Atualmente, com a evolução do poder de processamento, as aplicações de visão computacional utilizam inteligência artificial e aprendizagem automática (IA/ML) para processar dados visuais com maior precisão. Estas aplicações permitem a identificação de objetos, reconhecimento facial, classificação, recomendação, rastreamento e perceção. A visão computacional pode ser descrita, de forma simples, como o processo de permitir que as máquinas “vejam” e interpretem o mundo visual. Inspirada no sistema visual humano e na sua relação com o cérebro, esta área consiste na capacidade dos computadores de captar imagens e vídeos, analisá-los e transformá-los em informação significativa através de técnicas de aprendizagem automática e inteligência artificial. Com origem na década de 1950, a visão computacional começou a ser comercializada na década de 1970, inicialmente para distinguir entre texto dactilografado e manuscrito. Atualmente, é utilizada em múltiplas áreas, incluindo deteção e reconhecimento de pessoas, sistemas de controlo de qualidade industrial, veículos autónomos, aplicações agrícolas, saúde, educação e defesa.

O processamento de imagem envolve o tratamento, a restauração e o ajuste da cor e do brilho dos dados captados numa imagem, incluindo fotografias e vídeos. A visão computacional, por sua vez, corresponde à aplicação dessas técnicas de processamento de imagem e à sua integração com a inteligência artificial. Por exemplo, enquanto o processamento de imagem pode ser utilizado para detetar as margens (contornos) de um objeto numa imagem, a utilização de técnicas de aprendizagem automática para identificar e classificar esse objeto enquadra-se no domínio da visão computacional.

## UNIDADE 5.2 COMO É QUE UM COMPUTADOR "VÊ"?

Os elementos fundamentais de praticamente todos os computadores no mundo — as suas "células" — são constituídos pelos números 1 e 0. Isto significa que os computadores percecionam os dados que os utilizadores lhes fornecem na sua própria linguagem: 1s e 0s. Afinal, todas as imagens e vídeos, que são compostos por muitas imagens, também são dados. Por isso, os computadores interpretam-nos como sequências de zeros e uns, ou seja, no sistema binário. Para ilustrar melhor esta ideia, podemos recorrer à imagem apresentada.

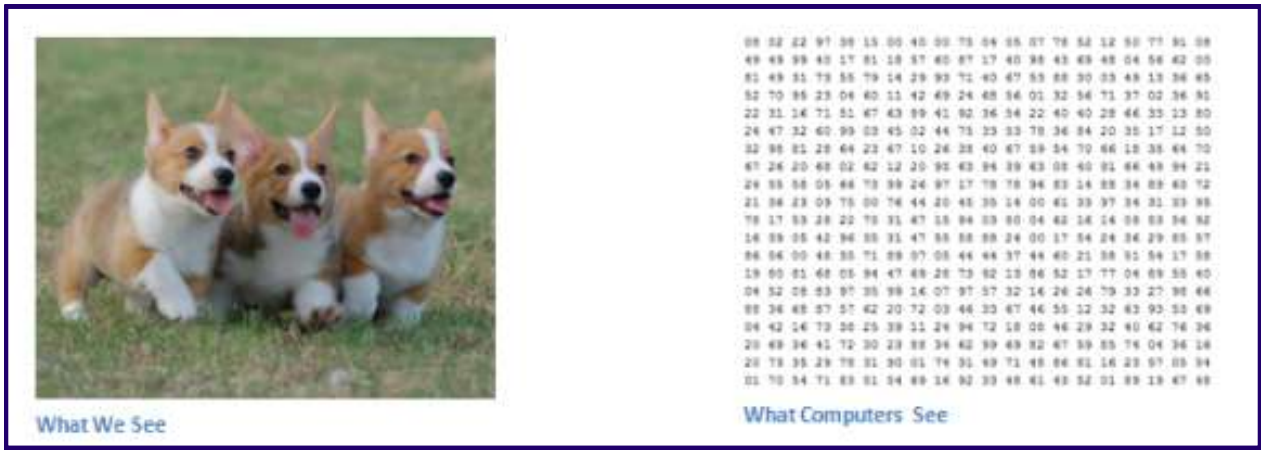



Figura 7 – Forma de Visualização: Utilização de redes neuronais para videovigilância (2025).
Disponível em: https://www.videonet9.com/using-neural-networks-for-video-surveillance.html
Data de acesso: 09.08.2025.

Mas se estiver a perguntar-se: nesta imagem aparecem valores com dois dígitos, e não zeros e uns. Isso é verdade. Os processadores do computador interpretam esses valores convertendo-os internamente em zeros e uns. Se esses valores fossem apresentados diretamente em binário, o resultado seria uma visualização muito longa e difícil de compreender.

Na realidade, cada um desses valores de dois dígitos na imagem acima corresponde a um pixel.

Então, o que é um pixel? É a unidade mais pequena que permite gerar e controlar a imagem num ecrã digital.

## UNIDADE 5.3 VAMOS FAZER UM EXERCÍCIO

- Utilize um caderno quadriculado.
- Numere cada quadrado do caderno com valores entre 0 e 1.
- Quanto mais próximo de 1 estiver o número escrito num quadrado, mais branco será esse quadrado.
- Quanto mais próximo de 0 estiver o número, mais escuro será o quadrado.
- Se o número for 0,5, pinte o quadrado de cinzento.

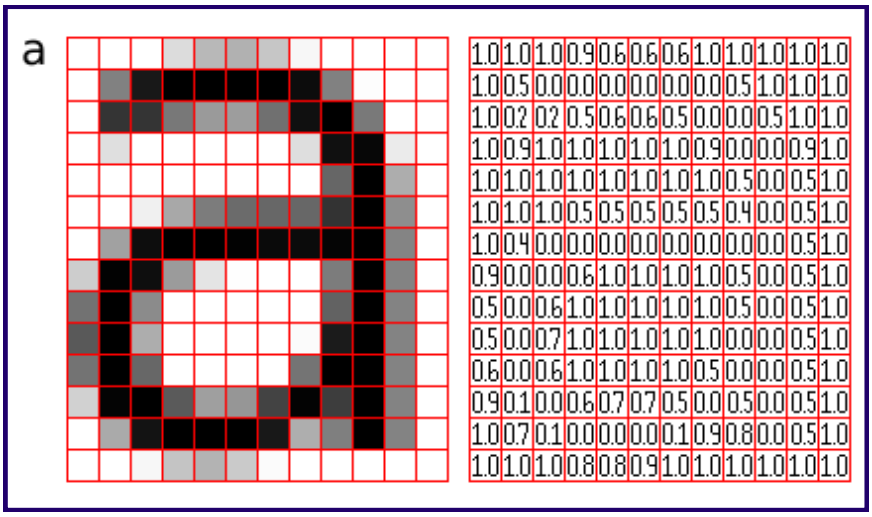



Figura 8 – Como os computadores veem pixels.
Disponível em: https://medium.com/@meriyahanjika99/from-pixels-to-predictions-getting-started-with-cnns-convolutional-neural-networks-63f84781cc1e
Data de acesso: 09.08.2025.

Se desenhar cuidadosamente num caderno quadriculado como este, a imagem parecerá um desenho normal quando observada à distância. Experimente fazê-lo com a imagem apresentada acima. Afaste-se alguns metros do ecrã e verá que a imagem parecerá mais nítida e natural. Na realidade, os ecrãs de computadores e televisores apresentam imagens e vídeos através de milhões de pequenos quadrados organizados em grelha. Para guardar essas imagens, o computador armazena os valores numéricos correspondentes a cada quadrado na memória. Nas imagens de baixa qualidade, que designamos por pixelizadas, é possível observar claramente esses quadrados individuais que compõem a imagem.

## UNIDADE 5.4 OS PROCESSOS DE VISÃO COMPUTACIONAL CONSISTEM ESSENCIALMENTE EM CINCO ETAPAS:

1. Obter a imagem/dados
2. Preparar a imagem/dados
3. Determinar o modelo de IA a utilizar
4. Treinar o modelo de IA selecionado
5. Interpretar os resultados

As imagens obtidas para uma aplicação simples de reconhecimento de dígitos manuscritos são apresentadas abaixo. Estas imagens foram convertidas para um formato em escala de cinzentos adequado ao processamento pelo computador e utilizadas para treinar o modelo de IA. Os resultados dos testes indicam uma taxa de precisão de 99% após o treino

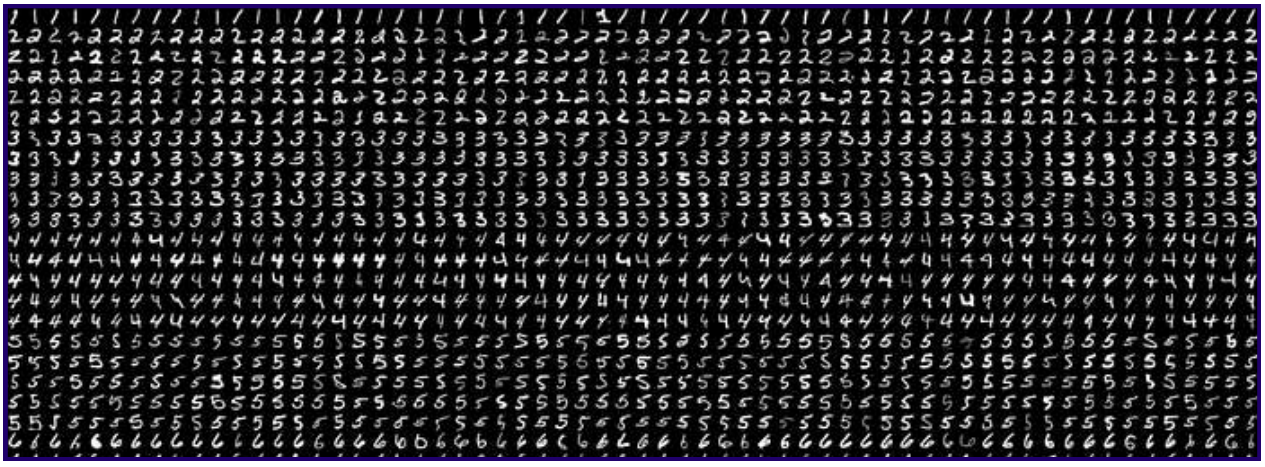


Figura 9 – Kaggle Dataset (2025). Disponível em: www.roboflow.com. Data de acesso: 09.08.2025.

O Roboflow é uma plataforma abrangente de inteligência artificial concebida para o desenvolvimento de projetos de visão computacional. Simplifica o processo de criação, treino e implementação de modelos de visão computacional utilizando dados de imagem e vídeo. Oferece uma interface intuitiva e fácil de utilizar, adequada tanto para iniciantes como para programadores experientes.

## UNIDADE 5.5 COMO REGISTAR-SE NO ROBOFLOW?

- Aceda a https://roboflow.com/.
- Clique no botão Sign In (Iniciar Sessão) no canto superior direito.
- Na janela que surge, introduza os seus dados de acesso, caso já tenha conta, e inicie sessão. Se estiver a aceder pela primeira vez, utilize a opção Continue With Google, por ser a forma mais prática de criar e aceder à conta
- Após selecionar a conta Google que pretende utilizar no Roboflow e conceder as permissões necessárias, a sessão será iniciada.
- Clique no botão NEW PROJECT (Novo Projeto) na página que se abre, introduza as informações do projeto e crie o projeto.

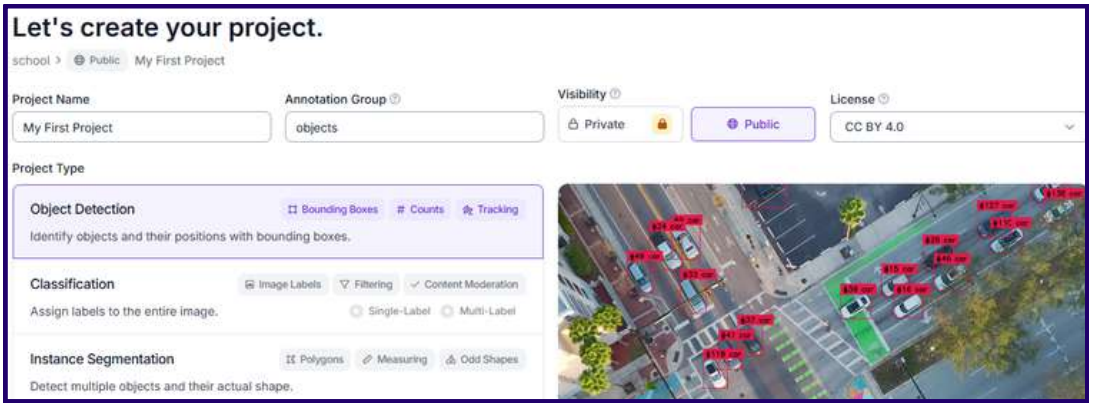



Figura 10 – Kaggle Dataset (2025). Disponível em: www.roboflow.com. Data de acesso: 09.08.2025.

Na nova página que se abre, adicionamos os dados de treino, validação e teste para a nossa IA. A partir deste momento, iremos referir-nos a estes dados como "conjunto de dados" (dataset). Caso não disponhamos de um conjunto de dados próprio, podemos utilizar conjuntos de dados já disponíveis no sistema Roboflow Universe: https://universe.roboflow.com

De forma semelhante, o Kaggle é uma plataforma que disponibiliza conjuntos de dados prontos a utilizar: https://www.kaggle.com/

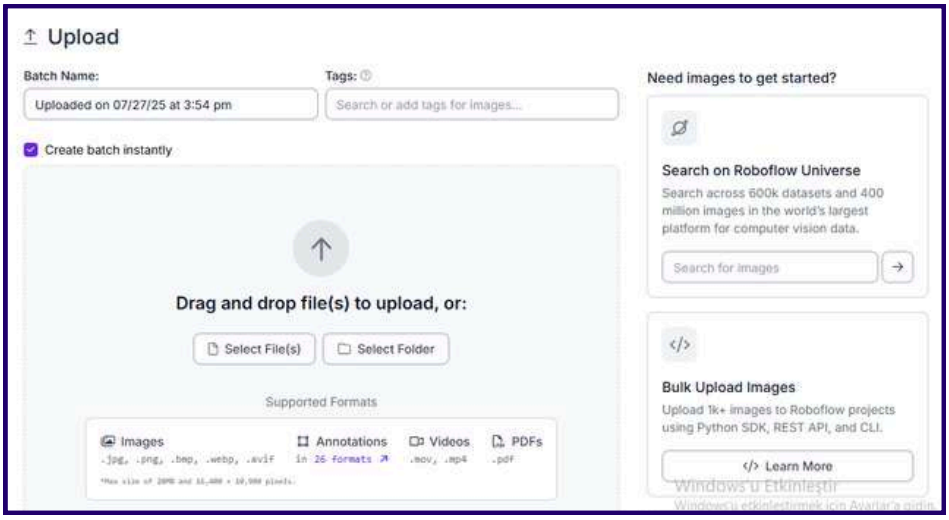



Figura 11 – Roboflow (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

Neste sistema, iremos continuar as etapas do processo utilizando um conjunto de dados pré-existente. Assim, utilizaremos o conjunto de dados disponível em: https://universe.roboflow.com/roboflow-58fyf/rock-paper-scissors-sxsw através da plataforma Roboflow.

Após aceder a este endereço, clique no botão para descarregar o conjunto de dados. No entanto, antes de efetuar o download, é necessário decidir qual o modelo de IA a utilizar. Nesta aplicação, iremos utilizar o modelo YOLOv11, que apresenta maior eficiência na deteção de objetos. Se visitar o perfil GitHub dos algoritmos relevantes da Ultralytics (https://github.com/ultralytics/ultralytics), encontrará diferentes pesos pré-treinados e informações detalhadas. (Teste online do YOLOv3: https://v-iashin.github.io/detector/)

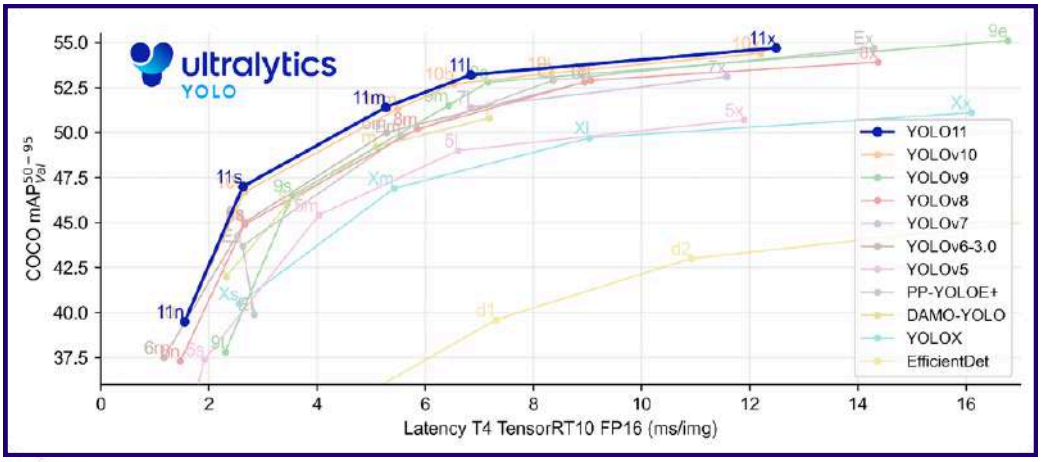



Figura 12 – Dataset (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

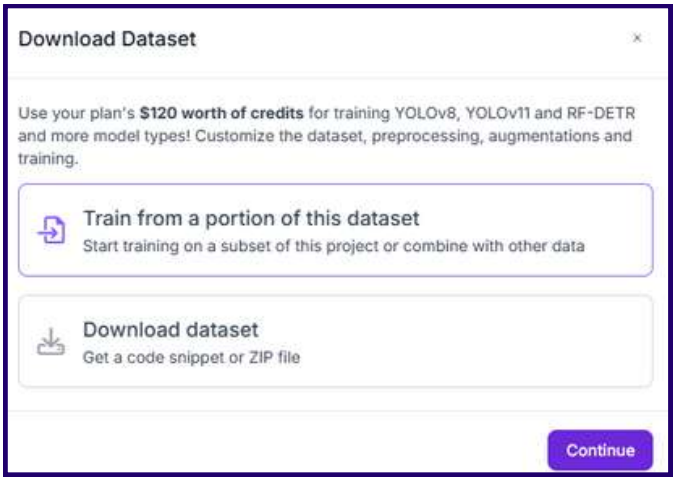



Figura 13 – Dataset (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

Após rotular e verificar os dados no conjunto de dados, clique no botão Download Dataset, selecione o modelo e o método online e obtenha o código Python disponibilizado pelo site.

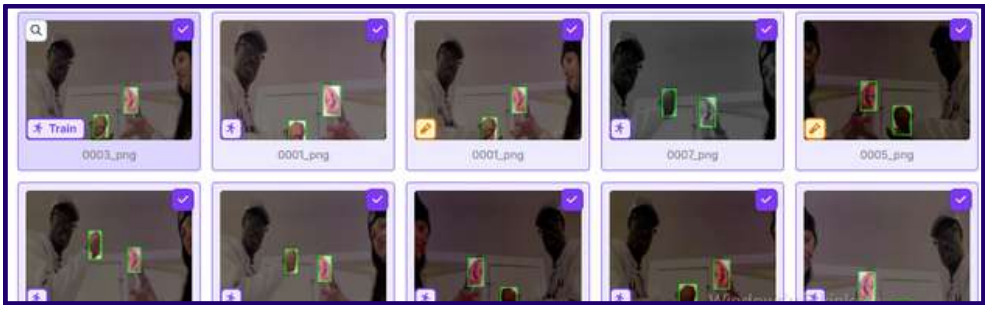



Figura 14 – Dataset (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

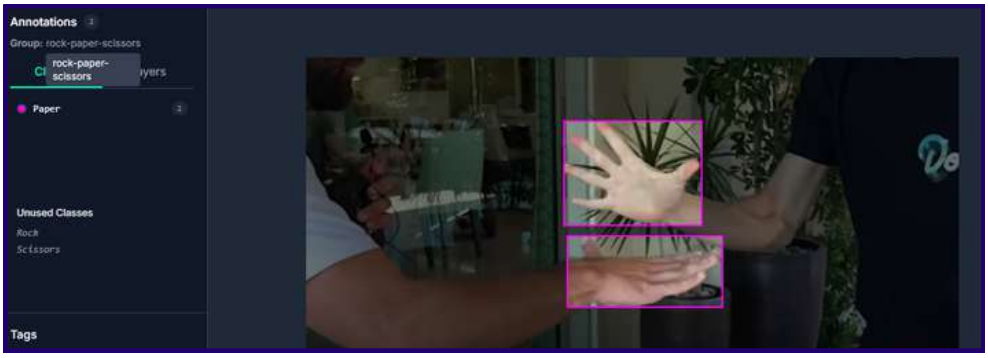



Figura 15 – Cut-off (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

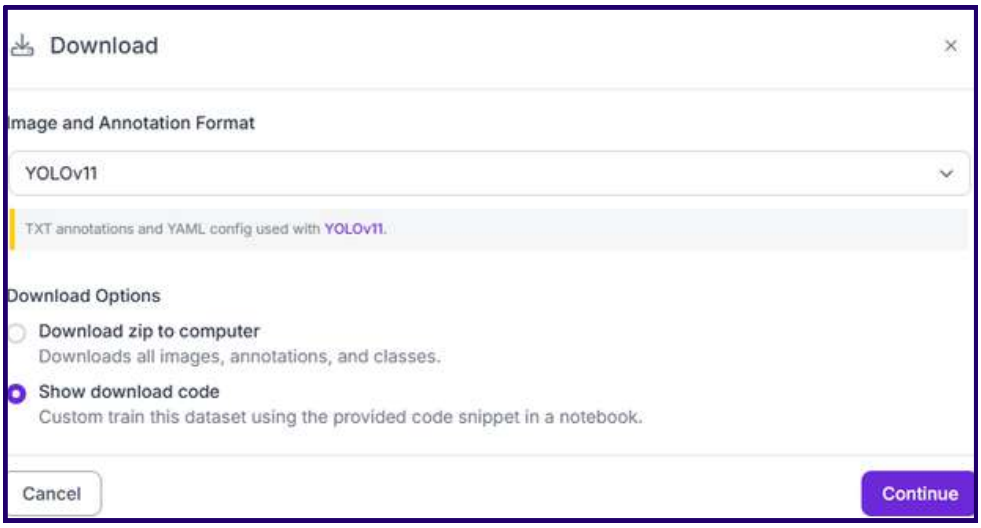



Figura 16 – YOLOv11 (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

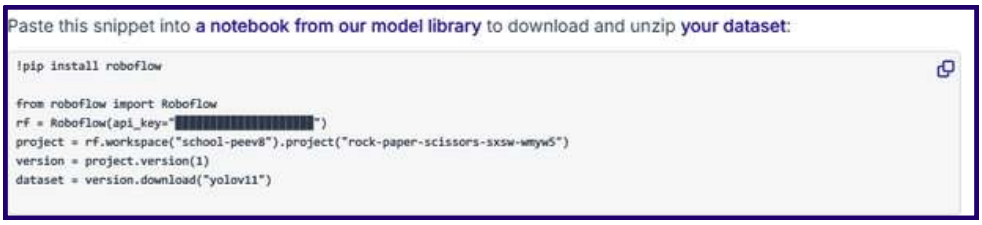



Figura 17 – Snippet (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

No passo seguinte, iremos abrir o código do algoritmo YOLOv11, que é executado no Google Colab e disponibilizado pela Roboflow: https://blog.roboflow.com/yolov11-how-to-train-custom-data/.

Aceda a: https://colab.research.google.com/github/roboflow-ai/notebooks/blob/main/notebooks/train-yolo11-object-detection-on-custom-dataset.ipynb?ref=blog.roboflow e siga as instruções apresentadas.

Outra possibilidade é criar e treinar o modelo diretamente na plataforma Roboflow. Esta opção pode ser mais rápida e simples, mas o YOLOv11 oferece, em geral, melhores e mais precisos resultados. Nos últimos anos, duas arquiteturas de deteção de objetos marcaram profundamente a área da visão computacional: as Redes Neuronais Convolucionais (CNNs) e o You Only Look Once (YOLO). Tanto as CNNs como o YOLO contribuíram significativamente para o desenvolvimento da visão computacional moderna. Mais recentemente, os Transformers têm vindo a assumir um papel cada vez mais relevante na visão computacional e na deteção de objetos.

As CNNs (Redes Neuronais Convolucionais) utilizam convoluções — uma técnica de processamento de imagem — para aprender características (features) de uma imagem. Este processo envolve a aplicação de uma "janela deslizante" sobre os píxeis da imagem, permitindo extrair padrões como contornos, texturas e formas. A informação obtida através das convoluções é posteriormente processada por uma rede neuronal. Existem várias implementações de CNN, incluindo R-CNN, Fast R-CNN e Mask R-CNN.

A família de modelos YOLO (You Only Look Once) também teve um papel extremamente relevante na evolução da visão computacional. Introduzido por Joseph Redmon em 2014, o YOLO tornou-se uma área ativa de investigação e desenvolvimento, sendo amplamente adotado pela comunidade científica e por programadores em todo o mundo. As versões YOLOv5 e YOLOv8, desenvolvidas e aperfeiçoadas pela equipa Ultralytics, são atualmente utilizadas em modelos de deteção de objetos em produção a nível global.

Mais informações sobre o YOLO podem ser consultadas em:
https://blog.roboflow.com/guide-to-yolo-models

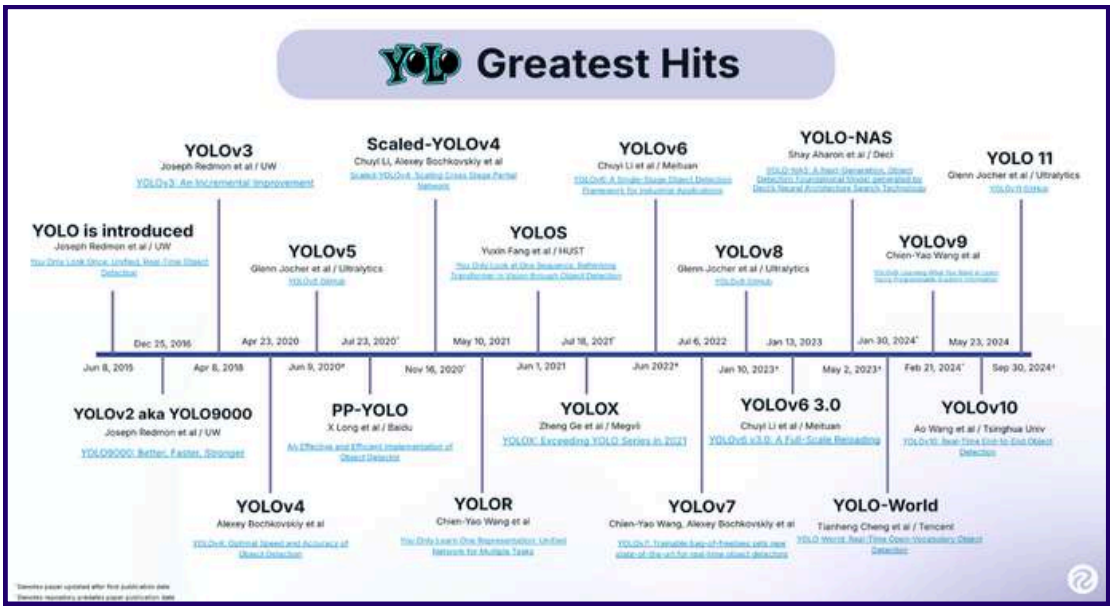



Figura 18 – História do YOLO (2025) disponível em www.roboflow.com, data de acesso: 09.08.2025.

Para analisar a classificação detalhada dos algoritmos atualmente utilizados na área da visão computacional a nível mundial, pode consultar o seguinte endereço:
https://leaderboard.roboflow.com/?ref=blog.roboflow.com

# REFERÊNCIAS

## Livros e Documentação


- Banerjee, Chayan & Fookes, Clinton & Karniadakis, George (2023). Physics-Informed Computer Vision: A Review and Perspectives. 10.48550/arXiv.2305.18035.

- LeCun, Y. (2025). MNIST: A Remote View of the Dataset. Disponível em: http://yann.lecun.com/exdb/mnist/ Data de acesso: 09.08.2025.

- Mohite, Amruta; Kulkarni, Atharva; Chitnis, Rutwik; Mane, Swapnil; Asabe, Shubham (2021). AI Inspection: Computer Vision for Visual Inspection. International Journal of Advance Research in Computer Science and Management, 7, 29.

- Object Recognition (2025). Disponível em: https://viso.ai/product/computer-vision-parking-lot-occupancy-tutorial/ Data de acesso: 10.08.2025.

- Using Neural Networks for Video Surveillance (2025). Disponível em: https://www.videonet9.com/using-neural-networks-for-video-surveillance.html Data de acesso: 09.08.2025.


## LIGAÇÕES PARA PLATAFORMAS DE APRENDIZAGEM

Para analisar a classificação detalhada dos algoritmos atualmente utilizados na área da visão computacional a nível mundial, pode consultar:
https://leaderboard.roboflow.com/?ref=blog.roboflow.com

O Roboflow é uma plataforma abrangente de inteligência artificial concebida para o desenvolvimento de projetos de visão computacional: https://roboflow.com/

## EXERCÍCIO PRÁTICO

### CAÇADORES DE OBJETOS – GUIA DO PROFESSOR

**Objetivo da Atividade**

- Compreender a lógica da visão computacional.
- Criar um modelo de reconhecimento de objetos utilizando o Roboflow.
- Vivenciar as etapas de recolha de dados, rotulagem, treino e teste do modelo.

**Lista de Preparação**

- Computadores ou tablets com ligação à internet.
- Grupos de 4–5 alunos.
- Smartphones (para tirar fotografias).
- Conta gratuita no Roboflow.
- Objetos simples (livro, lápis, garrafa de água, etc.).

**Duração e Cronograma**

Parte 1: Introdução à Visão Computacional – 15 minutos
Parte 2: Recolha de Fotografias – 20 minutos
Parte 3: Rotulagem – 25 minutos
Parte 4: Treino do Modelo – 20 minutos
Parte 5: Teste e Competição – 20 minutos

**Etapas de Implementação**

1. Apresentar aos alunos o conceito de visão computacional através de uma breve explicação.
2. Os grupos escolhem dois objetos diferentes e tiram 15–20 fotografias de diferentes ângulos.
3. Iniciar sessão no Roboflow e criar um novo projeto
4. Carregar as fotografias e rotular os objetos desenhando caixas delimitadoras.
5. Treinar e testar o modelo.
6. Os grupos testam os modelos uns dos outros e são atribuídas pontuações.

## CAÇADORES DE OBJETOS – GUIA DO PROFESSOR

**Ideias de Gamificação**
Corrida contra o Tempo: Quem consegue recolher e rotular os dados mais rapidamente?
Teste Surpresa: O modelo é testado com objetos que nunca viu antes.
Caça ao Erro: Identificar situações em que o modelo falha e discutir as causas.

## CAÇADORES DE OBJETOS – FICHA DE ATIVIDADE DO ALUNO

**O que é Visão Computacional?**
A visão computacional é uma área da inteligência artificial que permite aos computadores detetar e compreender objetos através de câmaras ou imagens.

**As Tuas Tarefas**

1. Em equipa, escolham dois objetos.
2. Tirem 15–20 fotografias de cada objeto, a partir de diferentes ângulos.
3. Criem um projeto no Roboflow e carreguem as fotografias.
4. Rotulem os objetos.
5. Treinem o vosso modelo.
6. Tentem reconhecer os objetos das outras equipas.

**Dicas para Fotografia**

- Fotografem a partir de diferentes ângulos.
- Experimentem diferentes condições de iluminação.
- Tirem fotografias tanto de perto como de longe.

**Guia de Rotulagem**
Desenhe uma caixa à volta do objeto na fotografia e escreva o seu nome.
Exemplo: Lápis, Borracha.

**As Minhas Observações**

- Situações em que o meu modelo reconhece melhor.
- Situações em que o meu modelo apresenta mais dificuldades.

## PERGUNTAS DE REFLEXÃO

Reserve algum tempo para refletir profundamente sobre o que aprendeu neste módulo.
Responda às seguintes questões de forma fundamentada:

- **Dos Píxeis à Perceção:** Neste módulo, aprendeu que os computadores percecionam imagens como grelhas de números e código binário, e não como formas e cores. De que forma esta visão "numérica" do mundo alterou a sua compreensão sobre como a IA distingue dois objetos diferentes, como um lápis e uma borracha?

- **A Importância da Diversidade de Dados:** Na atividade "Caçadores de Objetos", foi destacada a importância de tirar fotografias a partir de diferentes ângulos e em diferentes condições de iluminação. Porque é que esta diversidade no conjunto de dados é essencial para que a IA reconheça corretamente um objeto em situações reais?

- **Analisar a Precisão e os "Erros":** Durante a fase de teste, foi incentivado a identificar situações em que o modelo falhou — a chamada "Caça ao Erro". Com base nas suas observações, que fatores ambientais específicos (como sombras, distância ou fundo) dificultaram o desempenho correto do modelo?

- **Resolução de Problemas no Mundo Real:** A visão computacional já é utilizada em áreas como a saúde, os veículos autónomos e a agricultura. Após treinar o seu próprio modelo, identifique um problema específico na sua escola ou comunidade que poderia ser resolvido utilizando esta tecnologia.


## QUESTIONÁRIO DE ESCOLHA MÚLTIPLA (uma resposta correta por pergunta)

**1. Qual é o principal objetivo da visão computacional?**
a) Permitir que os computadores pensem como o cérebro humano.
b) Permitir que as máquinas "vejam" e interpretem o mundo como o olho humano.
c) Armazenar grandes conjuntos de dados em bases de dados.
d) Analisar apenas dados em formato de texto.

**2. Qual das seguintes opções NÃO é uma aplicação típica da visão computacional?**
a) Reconhecer sinais de trânsito em veículos autónomos.
b) Sistemas de reconhecimento facial.
c) Analisar preferências dos utilizadores em sistemas de recomendação de produtos.
d) Detetar tumores em imagens médicas.

**3. Qual é a tarefa de visão computacional que determina a localização e o tipo de objetos específicos numa imagem?**
a) Classificação de Imagem
b) Deteção de Objetos
c) Segmentação de Imagem
d) Extração de Características

**4. Qual das seguintes opções representa um dos desafios que os sistemas de visão computacional enfrentam no mundo real?**
a) Escassez de dados e diversidade limitada desses dados.
b) Baixa dimensionalidade e estrutura simples dos dados visuais.
c) Consistência na iluminação, posição e escala.
d) Modelos que alcançam sempre resultados perfeitamente precisos.

**5. Qual das seguintes afirmações indica que um modelo está em overfitting (sobreajuste)?**
a) Baixa precisão tanto nos dados de treino como nos dados de teste.
b) Elevada precisão nos dados de treino e baixa precisão nos dados de teste.
c) Elevada precisão tanto nos dados de treino como nos dados de teste.
d) O modelo atribui sempre o mesmo rótulo a todas as imagens.

# Resumo e Próximos Passos

Com base nos conhecimentos fundamentais adquiridos no primeiro curso, estes materiais de aprendizagem avançada foram concebidos para envolver alunos do ensino secundário na aplicação prática da inteligência artificial. O conteúdo estabelece a ligação entre teoria e prática, orientando os formandos na criação de modelos simples de IA utilizando Python e Scratch. Desde a exploração da lógica matemática por trás da tomada de decisões e da "Sala de Escape com IA", até ao treino de modelos de aprendizagem automática para personagens de jogos e ao desenvolvimento de aplicações de visão computacional com o Roboflow, os módulos proporcionam uma experiência abrangente e prática. A abordagem narrativa, com personagens como Aylin e Alara, contribui para tornar conceitos técnicos complexos — como redes neuronais, agrupamento K-means e reconhecimento de objetos — mais acessíveis e motivadores. Ao concluir este curso, os alunos passam de consumidores passivos de tecnologia a criadores ativos. Desenvolvem competências técnicas para programar sistemas inteligentes, capacidade crítica para avaliar a precisão dos modelos e consciência ética para aplicar a IA de forma responsável. Esta introdução aprofundada à programação e aos primeiros passos na ciência de dados reforça as competências STEM e prepara os estudantes para futuros percursos académicos e profissionais num mundo cada vez mais orientado pela tecnologia.

## O Que Se Segue?

Para apoiar a integração sustentável destes temas avançados na sala de aula, o projeto irá apresentar um E-Toolkit para Educadores do Ensino Secundário. Este recurso disponibilizará planos de aula detalhados e orientações metodológicas para a implementação eficaz da formação em IA. Além disso, os alunos serão incentivados a demonstrar as competências adquiridas através da participação nos futuros concursos "AI Future Explorers", onde poderão aplicar os seus conhecimentos na resolução de desafios reais e estabelecer ligação com uma comunidade alargada de jovens inovadores.

# Parceiros do Projeto

FUTURE-STEM-HUB

UNIVERSITÄT DUISBURG ESSEN
Open-Minded

KÜTAHYA
İL MİLLÎ EĞİTİM MÜDÜRLÜĞÜ

tetra solutions

M&M profuture training

ETAP
ESCOLA PROFISSIONAL

**Coordenador do Projeto:**
**Universidade de Duisburg-Essen, Alemanha**

**Endereço de Email:**
**mustafa.bilgin@uni-due.de**

**Website**
**www.future-stem-hub.eu**

Co-funded by the European Union