Exercise walk-through

Artificial Intelligence (AI) ▪ 18.1

eXercise

You must be able to

- explain how **graphs** help AI and how an **artificial neural network** works
- compare **supervised**, **unsupervised** and **reinforcement** learning
- explain **deep learning** and **backpropagation**

Method — AI, ML and graphs

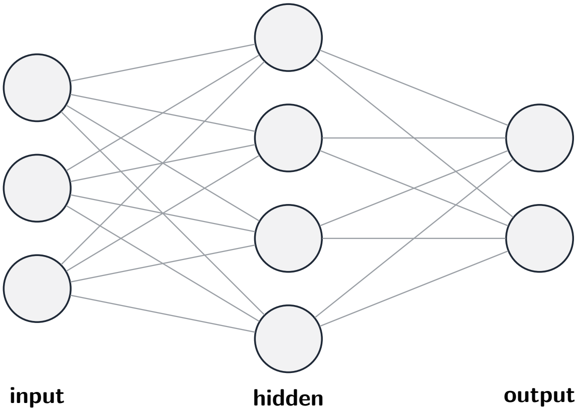
Machine learning (ML) learns patterns **from data** instead of fixed rules; **deep learning (DL)** is ML using deep neural networks. Many AI problems sit on a **graph** — **nodes** (states/places) joined by **edges** (moves, each with a cost). Two algorithms find the **shortest path**:

- **Dijkstra's algorithm** explores outward from the start, always extending the lowest-cost node, until it reaches the goal.
- **A*** does the same but adds a **heuristic** estimate of the distance **still to go**, which steers the search **toward the goal** — so it usually checks **fewer nodes** and is faster.

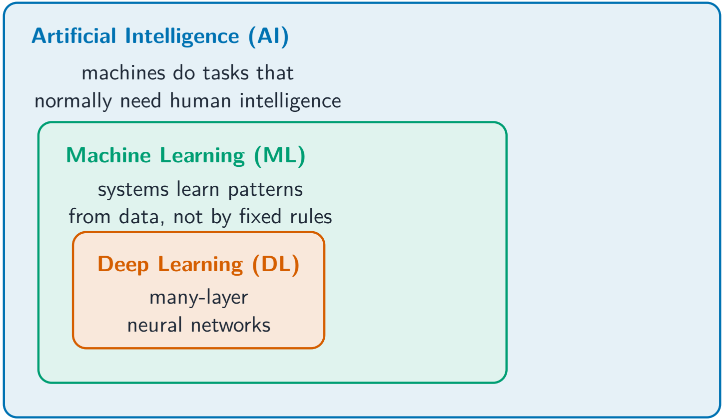
Method — Neural networks

A network has an **input** layer, one or more **hidden** layers, and an **output** layer. Each connection has a **weight**; a neuron sums its weighted inputs (plus a bias), applies an **activation function**, and passes the result on:

Method — Neural networks



Method — Neural networks



How AI, machine learning and deep learning relate

Method — Types of machine learning

- **supervised** — data has **labels**; learns input → label (classification, regression).
- **unsupervised** — **no labels**; finds structure (e.g. **clusters**).
- **reinforcement** — an **agent** acts, gets a **reward**, and learns a **policy** that maximises total reward by trial and error.

Training adjusts the weights by **backpropagation**: a forward pass computes the output, a **loss function** measures the error, and the error is propagated **backwards** to update each weight (gradient descent). Deep learning needs **lots of data and compute**.

Practice 2.1 ▪ 1 mark

State what **machine learning** is.

Solution 2.1

Worked steps

Algorithms that **learn patterns from data** (and improve with experience) instead of being programmed with explicit rules **B1**.

Practice 2.2 ▪ 2 marks

Name the **three** kinds of layer in a neural network.

Solution 2.2

Method: Neural networks

Worked steps

Input layer, **hidden** layer(s), **output** layer **B1** **B1** *for two, for all three.*

Practice 2.3 ▪ 1 mark

In a graph used for **pathfinding**, state what an **edge** represents.

Solution 2.3

Worked steps

A **move/connection** between states (e.g. a road between two places), often with a **cost/weight** B1.

Practice 2.4 ▪ 1 mark

State what a **weight** is in an artificial neuron.

Solution 2.4

Method: Neural networks

Worked steps

A value each input is **multiplied by**, setting how strongly that input affects the neuron's output B1.

Exam-style 3.1 ▪ 2 marks

Describe the difference between **supervised** and **unsupervised** learning.

Solution 3.1

Method: Types of machine learning

Worked steps

Supervised learning trains on **labelled** data (each input has a known correct output) **B1**; **unsupervised** learning uses **unlabelled** data and finds structure/clusters itself **B1**.

Exam-style 3.2 ▪ 3 marks

Explain how **reinforcement learning** works, using the terms **agent**, **reward** and **policy**.

Solution 3.2

Method: Types of machine learning

Worked steps

An **agent** takes an **action** in an environment, which changes the state and returns a **reward** B1; over many trials it learns a **policy** (strategy) B1 that **maximises total reward** by trial and error B1.

Exam-style 3.3 ▪ 2 marks

Explain why **deep learning** needs a large amount of **data** and **computing power**.

Solution 3.3

Method: Types of machine learning

Worked steps

Deep networks have **very many weights** to learn B1, so they need large datasets to learn good patterns and heavy compute (GPUs) to train in reasonable time B1.

Exam-style 3.4 ▪ 2 marks

State the role of **backpropagation** in training a neural network.

Solution 3.4

Worked steps

It propagates the output **error backwards** through the layers B1 to **adjust each weight** so the network's output gets closer to the target (gradient descent) B1.

Exam-style 3.5 ▪ 3 marks

State the purpose of **Dijkstra's algorithm** and **A***, and outline **one** difference between them.

Solution 3.5

Method: AI, ML and graphs

Worked steps

Both find the **shortest path** between two nodes in a weighted graph B1. Difference: **A*** uses a **heuristic** to estimate the remaining distance and head toward the goal, so it usually explores **fewer nodes / is faster** B1; **Dijkstra's** has no heuristic and explores outward uniformly B1.

Exam-style 3.6 ▪ 3 marks

A delivery company is building software to plan its routes and to read handwritten address labels.

(a) **Identify** one Artificial Intelligence algorithm suitable for finding the shortest distance between two depots on a road network, and **state** what the nodes and the edge weights represent.

(b) **Describe** how that algorithm proceeds from the starting node.

(c) The company also considers the **A*** algorithm. **State** the one thing A* uses that the algorithm in (a) does not, and **explain** the benefit.

(d) **Describe** what is meant by **Deep Learning**, and **state** one example of where it is used.

(e) **Explain** why Deep Learning suits the handwriting-reading task better than a set of hand-written rules.

Solution 3.6

Method: AI, ML and graphs

Worked steps

(a) **Dijkstra's algorithm** B1. The **nodes** are the depots and road junctions B1; the **edge weights** are the distances, or the travel times, along each road between them B1.

(b) It gives the start node a distance of zero and every other node infinity B1. It then repeatedly takes the **unvisited node with the smallest known distance**, and for each of its neighbours checks whether going via this node is shorter than the distance recorded so far, updating it if so B1. The node is then marked visited and never reconsidered; the process ends when the destination is visited B1.

Solution 3.6

(c) A* also uses a **heuristic** — an estimate of the remaining distance from each node to the goal, such as the straight-line distance B1. It therefore explores nodes that head **towards** the destination first B1, so it usually examines far fewer nodes than Dijkstra while still finding the shortest route, provided the heuristic never overestimates B1.

(d) Deep Learning uses an **artificial neural network with many hidden layers** B1, trained on large amounts of data, where each layer learns features built from the layer before it B1. Examples: speech recognition, image and face recognition, machine translation, self-driving vehicles B1.

(e) Handwriting varies enormously between people, so the rules that distinguish a 7 from a 1 cannot be written out completely B1. A network **learns** the distinguishing features from thousands of labelled examples instead of being told them B1, and it generalises to handwriting it has never seen, which a fixed rule set cannot do B1.