

# ECS759P Artificial Intelligence

## Coursework 1

Student Name: Dariga Kassenova

Student Number: 250477881

### 1. Four Categories of AI

#### 1.1 Category: Acting Humanly

<b>Prompt</b>	I'm currently in another country, in England. After arriving here, I started to feel homesick and realized I'm completely alone in a foreign place. I need some help, can you comfort me? Try to answer briefly.
<b>AI Agent+Version</b>	ChatGPT Plus (GPT-5)
<b>Response</b>	Of course Dariga  It's completely okay to feel this way — being far from home can feel heavy at first. Remember, it's not loneliness, it's just your heart adjusting to a new place. Take a deep breath, maybe call someone you love, or go for a small walk — you're not truly alone, you're just beginning a new chapter, and it will get easier soon.
<b>Explanation</b>	This prompt shows an emotionally sensitive and empathetic reaction rather than logical reasoning. The response shows warmth and gentle advice, reflecting human-like behaviour and empathy.

#### 1.2 Category: Thinking Humanly

<b>Prompt</b>	You're facing a choice, and every option feels both right and wrong. Describe what's happening in your mind at that moment, without rushing to a conclusion. Try to answer briefly.
<b>AI Agent+Version</b>	ChatGPT Plus (GPT-5)
<b>Response</b>	It feels like waves colliding — one thought insists "this is right," another whispers "but what if...?" Each possibility blooms, fades, returns, until everything quiets into still uncertainty.
<b>Explanation</b>	This prompt explores inner thought processes. The response shows emotional conflict, using metaphors like "waves" to imitate human consciousness and mental uncertainty.

#### 1.3 Category: Thinking Rationally

<b>Prompt</b>	There are three jars: one with rice, one with buckwheat, and one with a mixture of both. The labels are all wrong. How can you determine which jar contains only rice?
<b>AI Agent+Version</b>	ChatGPT Plus (GPT-5)
<b>Response</b>	<p>Pick a single grain from the jar labeled "Mixture."</p> <ul style="list-style-type: none"> <li>- Because all labels are wrong, that jar is <b>not</b> the mixture—it's either all rice or all buckwheat.</li> <li>- If the grain you pull is <b>rice</b>, then that jar is the <b>rice</b> jar.</li> <li>- If the grain you pull is <b>buckwheat</b>, then that jar is the <b>buckwheat</b> jar, and the <b>rice</b> jar must be the one labeled "Buckwheat" (since the one labeled "Rice" can't be rice and therefore is the mixture).</li> </ul> <p>So one grain from the "Mixture" label tells you exactly which jar is rice.</p>
<b>Explanation</b>	Solution comes from logical reasoning, just clear thinking to figure out which jar is which. The step where one grain from the "mixture" reveals all labels shows rational, step-by-step reasoning.

#### **1.4 Category: Acting Rationally**

<b>Prompt</b>	I have 50 pounds left for a week. I need to buy groceries, transport, and purchase a QMML membership for 5.99 pounds. How should I spend them wisely? Try to answer briefly.
<b>AI Agent+Version</b>	ChatGPT Plus (GPT-5)
<b>Response</b>	Set aside £6 for the QMML membership, leaving £44. Budget £25 for groceries (buy basics—rice, pasta, eggs, frozen veggies), £10 for transport (use student discounts or walk when possible), and keep £9 as an emergency buffer or small treat.
<b>Explanation</b>	This prompt requires practical decision-making under limited resources. The response shows rational planning, prioritization, and cost management to achieve the best outcome within constraints.

## **2. Research Paper Analysis**

- a) "Pain" is an internal signal that allows AI to "feel" emotional intensity, advancing affective empathy beyond just cognitive understanding.
- b) The model fits the Thinking Humanly category. The paper focuses on internal processes inspired by brain structures (ACC) that encode emotional pain and empathy, model imitates human neural mechanisms rather than just acting human-like.
- c) Integrating multimodal data, such as voice or facial expressions could greatly enhance the model's affective empathy and depth.

## **3. Jay McClelland & AI History**

### ***Part a) Historical Timeline***

1. **1950** - McClelland (born 1948) was later inspired by Turing's ideas on machine intelligence.
2. **1957** - As a child, he would later draw on Rosenblatt's perceptron ideas for his PDP models [2].
3. **1969** - While studying psychology at Columbia, he later revived neural research after Minsky and Papert's critique through the PDP approach [2].
4. **1980** - McClelland served on the faculty of the UC San Diego. His neural work contrasted with Searle's symbolic critique in the Chinese Room argument.
5. **1986** - At Carnegie Mellon, McClelland co-founded the PDP group with Rumelhart, integrating backpropagation into connectionism. [1].
6. **1997** - At Carnegie Mellon, he explored learning models opposing Deep Blue's rule-based logic.
7. **2006** - McClelland joined Stanford's Psychology Department, his cognitive modeling influenced Hinton's renewed deep learning research [4].
8. **2012** - Hinton, McClelland's collaborator, applied PDP principles in AlexNet, realizing connectionist ideas
9. **2017** - At Stanford, McClelland advanced connectionist theories that conceptually support transformer models [3].
10. **2024** - At Stanford, his research on learning and cognition continued, as modern AI embodied large-scale PDP principles. [3].

*(Word count excluding dates and citations: 147 words)*

### ***Part b) Foundations of AI***

McClelland's work connects to *Neuroscience*, as his PDP models simulate brain-like neural processing, and to *Psychology*, as they explain learning and memory through computational models that reflect human cognition.

### ***Part c) Network Positioning***

McClelland would be on the connectionist side, as his models focus on neural learning, not symbolic rules [2].

### ***Part d) Bibliography and Citations***

- [1] Rumelhart, D.E., 1986. Parallel distributed processing: Explorations in the microstructure of cognition. *The MIT Press google schola*, 2, pp.359-366.

- [2] Medler, D.A., 1998. A brief history of connectionism. *Neural computing surveys*, 1, pp.18-72.
- [3] Thomas, M.S. and McClelland, J.L., 2008. Connectionist models of cognition. *The Cambridge handbook of computational psychology*, pp.23-58.
- [4] Hinton, G.E. and Salakhutdinov, R.R., 2006. Reducing the dimensionality of data with neural networks. *science*, 313(5786), pp.504-507.

#### **4. Agent Description** (*Prepared with assistance from ChatGPT (GPT-5, October 2025)*)

**Agent:** Music Understanding Evaluation Agent

**Performance:** Accuracy, Sensitivity to musical structure, Human alignment

**Environment:** Audio stimuli, Audio LLMs, Human participants

**Actuators:** Benchmark scoring, Result visualization, Automated data logging

**Sensors:** Audio waveforms, Spectrograms, Tempo, Tonal features

#### **5. Path Finding Warmup**

**Part a) Breadth-First Search:** A, B, C, D, H, E, F, G

**Part b) Depth-First Search:** A, B, D, C, H, E, F, G

**Part c) Least expensive routes and their total costs:**

City	Least expensive route	Total cost
B	H -> C -> D -> B	90
D	H -> C -> D	60
F	H -> C -> D -> E -> G -> F	200

**Part d) There is two possible UCS routes from A to G:**

1. A -> B -> D -> E -> G = 40 + 30 + 60 + 70 = 200
2. A -> C -> D -> E -> G = 50 + 20 + 60 + 70 = 200

**Part e)**

**Optimal path: A->C->D->E->G** (Figure 5.1)

**Total cost: 200**

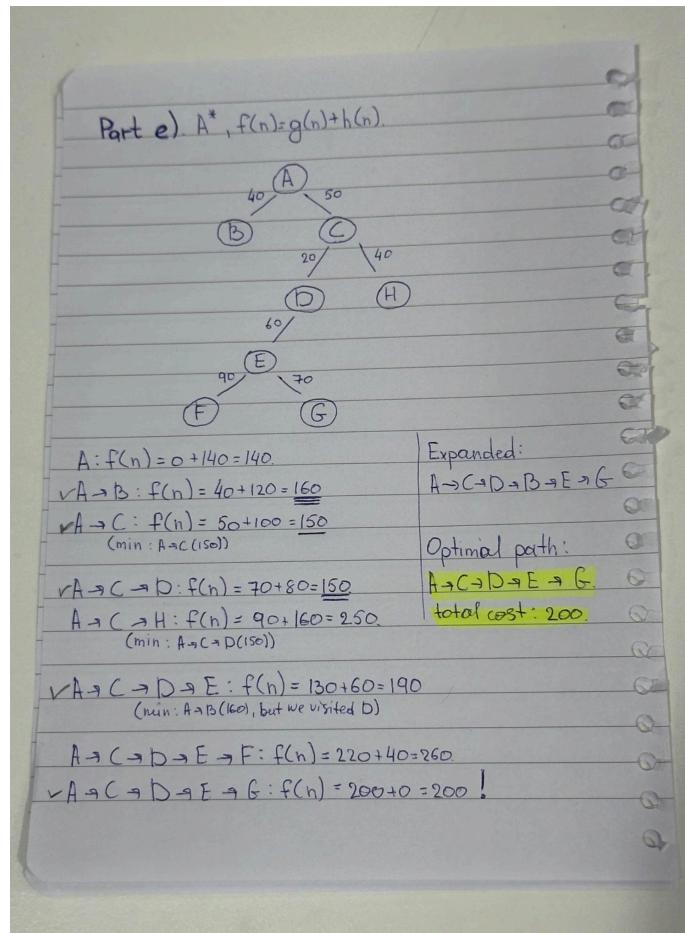


Figure 5.1 – A\* search

## 6. Networks and Pathfinding

### 6.1 Task 1: Longest Path

To identify the words for the `print_long_path` function, I used a custom heuristic algorithm. The algorithm began its search from nodes with the lowest degree, since such nodes are located on the outer edges of the graph and have fewer connections to other words. This increases the likelihood of constructing a longer path without cycles, effectively traversing the network from edge to edge. The path was then greedily expanded by selecting unvisited neighbors with similarly low degrees, avoiding loops. After several iterations, the route with the greatest number of unique nodes was selected, and its first and last words were fixed as `start_word` and `end_word` in the `print_long_path` function.

### 6.2 Task 2: Longest Quote (explanation was written with the help of ChatGPT (GPT-5, November 2025) to improve clarity and wording)

The helper function performs a linear traversal through the `original_tokens` of the text, the algorithm sequentially checks each word to find the longest continuous chain of words that are connected in the graph and not included in `rare_tokens`. When a connection break or a rare word is encountered, the current chain is reset. Finally, the

longest sequence found is selected, and its first and last words are fixed as the `start_word` and `end_word` in the `print_long_quote` function.

### 6.3 Task 3: Most Expensive Path (explanation was written with the help of ChatGPT (GPT-5, November 2025) to improve clarity and wording)

To select the words for `print_expensive_path`, I used a greedy search approach. I formed a set of starting nodes from those with the lowest and highest degrees, covering both the edges and the center of the network. For each start node, the algorithm built a path without cycles: at each step, it chose an unvisited neighbor with the maximum edge weight, where the weight was taken from the `distance_matrix`. The search was limited by `max_depth` to prevent infinite growth and revisits. For each path, the total edge cost was accumulated, and the pair with the highest `total_cost` was saved. The final expensive path defined `start_word = path[0]` and `end_word = path[-1]`.

### 6.4 Task 4: Most Expensive Quote

The algorithm uses similar logic to task Longest Quote (6.2). For each continuous sequence, the total edge cost is accumulated from the `distance_matrix`, and the one with the highest `max_cost` is retained.

### 6.5 Task 5: Heuristic Search (Part c. Heuristic Description) (explanation was written with the help of ChatGPT (GPT-5, November 2025) to improve clarity and wording)

Both functions `complete_sentence` and `start_sentence` were implemented based on a modified **A\*** search algorithm.

In `complete_sentence`, the algorithm starts from the last word of the left context (`start`) and searches for a connection to the first word of the right context (`end`), or to a period if the right side is missing. I used the adjacency data and token frequency counts (`token_counts`) to build an outgoing neighbor map `out_n`, showing which words most commonly follow each other in the text. I then combined several heuristic factors to guide the search. The total cost  $f$  consists of three components:

1. The **path cost**  $g$ , which adds a small base cost for each step plus an additional penalty for rare word transitions.
2. The **heuristic estimate**  $h$ , computed using a BFS distance from the current node to the target node, representing how close a word is to the goal.
3. A **length penalty** that prevents the algorithm from producing sentences that are too short.

The rarity penalty is calculated as

$$-\log((\text{adj}[(u,v)] + 1) / (\text{max\_count} + 1)) - 0.5 \cdot \log(\text{freq}(v) + 1)$$

This formula penalizes uncommon transitions and infrequent words. The first part,  $-\log((\text{adj}[(u,v)] + 1) / (\text{max\_count} + 1))$ , decreases the score for bigrams that rarely occur in the text, while the second term,  $-0.5 \cdot \log(\text{freq}(v) + 1)$ , slightly penalizes globally rare words. Together, these components push the search toward statistically probable and natural word sequences. The algorithm skips `<RARE>` tokens ensuring grammatical

and coherent completions. After the search, I merge the left context, the generated segment, and the right context, removing any duplicated boundary words.

For *start\_sentence*, I used the same principle but in reverse. I built an inverse neighbor map *in\_n* to identify which words typically come before others in the text. The algorithm begins from the first word of the right-hand context and moves backward through the graph, using the same cost function and heuristics as in *complete\_sentence*. Once a valid sequence is found, I reverse it and join it with the right context to form a coherent full sentence.

Both heuristics balance statistical plausibility and structural coherence: the rarity penalty ensures semantic naturalness, while the distance heuristic and length control maintain syntactic flow and sentence balance.

## USE OF AI AND LLMs

I used the LLM (GPT-5) primarily to express my ideas clearly and accurately in English. I wrote explanations and technical descriptions in Russian and then asked the model to translate them carefully into English while preserving the meaning, logic, and academic tone. This helped me avoid formulation errors and improve the clarity of my report. (e.g Figure 2)

In addition, I used the model while writing and improving my code. I asked it to check the correctness of my functions and assist with optimization. This was especially helpful in the final tasks 6.5 (a) and 6.5 (b) where I built upon ideas suggested by the model, adapted them to my own logic, and asked for feedback on their correctness. The model also helped me derive and refine the heuristic formula used in my A\* search algorithm.

Overall, the LLM served as a tool for verification, inspiration, and language support.

The screenshot shows a ChatGPT interface. At the top, there is a navigation bar with a back arrow, the text 'ChatGPT 5', and a share icon. Below the navigation bar is a large input field containing Russian text. To the right of the input field is a large gray box containing the English translation of the Russian text. At the bottom of the screen, there is a footer with a message about potential errors and a link to cookie settings.

Чтобы определить начальное и конечное слова для функции Longest Path, я использовала собственный эвристический алгоритм `greedy_longest_path()`. Алгоритм начал поиск с узлов, имеющих наименьшую степень связей, поскольку такие узлы располагаются на периферии графа и имеют меньше переходов к другим словам. Это повышает вероятность построить более длинный путь без циклов, проходящий «из края в край» сети. Затем путь жадно расширялся за счёт выбора непосещённых соседей с низкой степенью, избегая повторов. После нескольких итераций выбирался маршрут с наибольшим количеством уникальных узлов, а его первое и последнее слова были зафиксированы как `start_word` и `end_word` в финальной реализации функции.

"translate this text into english accurately and clearly

To identify the start and end words for the *Longest Path* function, I used a custom heuristic algorithm called `greedy_longest_path()`. The algorithm began its search from nodes with the lowest degree, since such nodes lie on the periphery of the graph and have fewer connections to other words. This increases the likelihood of constructing a longer path without cycles, effectively traversing the network "from edge to edge." The path was then greedily expanded by selecting unvisited neighbors with similarly low degrees, avoiding repetitions. After several iterations, the route with the greatest number of unique nodes was selected, and its first and last words were fixed as `start_word` and `end_word` in the final implementation of the function.

+ Спросите что-нибудь...

ChatGPT может допускать ошибки. Проверьте важную информацию. См. [настройки cookie-файлов](#).

Figure 2 – English translation prompt