

Lab 01 Map AI Systems

Gavin · AI-assisted course design

Contents

1 Purpose	3
2 The questions to keep separate	3
3 Two worked examples	4
3.1 Example 1 A polished help window	4
3.2 Example 2 One text model call	4
4 Your worksheet	4
5 Six system cards	5
5.1 A Greenhouse alarm	5
5.2 B Route desk	5
5.3 C Sorting gate	5
5.4 D Surface inspector	5
5.5 E Illustrated request	5
5.6 F Warehouse rover	6
6 Predict three interventions	6
7 Your artifact and review	6
8 Self-check answer key	6
8.1 Card classifications and layers	6
8.2 Intervention answers	7
8.3 Check the quality of your reasoning	7

1 Purpose

A system can be neural, generative, multimodal, and part of an agent at the same time. Those labels answer different questions. This Lab turns the map from Lesson 1.1, The AI Zoo into a practical method: identify the mechanism, locate the behavior, and say what the evidence supports.

You will produce six short system analyses and three intervention predictions. The systems are fictional specifications, not descriptions of commercial products. There is no test of whether something deserves the word “AI.” The test is whether your explanation follows from its stated construction.

Time: 45–60 minutes: orientation and examples, 10; cards, 20–25; interventions, 5–10; self-check and revision, 10–15.

Requirements: paper or any text editor. No coding, installation, account, paid service, or internet connection is required once you have this Lab. Basic arithmetic is sufficient. Reading Lesson 1.1 first is helpful; the vocabulary below makes the activity standalone.

2 The questions to keep separate

Use several axes rather than selecting one winning category.

- **Mechanism:** explicit rules; search over candidate solutions; classical, non-neural machine learning; neural computation. A system may combine them.
- **Learning:** were parameters fitted from examples or experience? Supervised learning uses supplied targets; self-supervised learning derives targets from the data itself; reinforcement learning uses rewards associated with actions and their consequences. A reward-trained system need not be neural.
- **Task:** classification assigns a category; prediction estimates something; search selects a solution; generation constructs content. Producing a sentence from a fixed template does not establish a learned generative model.
- **Modality:** what kinds of information enter and leave? Text, images, and audio are examples. Identify the boundary: an entire product can connect modalities even when one component handles only text.
- **Action loop:** does the system choose actions, receive results, and choose again toward a goal? Call this an agent loop here. A fixed sequence of model calls is orchestration, but is not sufficient evidence of this loop. Other definitions of “agent” exist; state yours if it changes a label.

Also distinguish the six layers from the lesson:

1. **Algorithm:** a procedure, such as searching paths or updating parameters.
2. **Architecture:** the structural design of a model, such as a convolutional network or Transformer. Ordinary software also has architecture; specify which meaning you intend.
3. **Trained parameters:** fitted values, including neural weights, regression coefficients, or a learned table. Architecture alone does not specify these values.
4. **Inference runtime:** software that loads and executes a fitted model. Its implementation and numerical settings can matter even when parameters stay fixed. A card may leave these details unspecified.
5. **Orchestration:** software that passes inputs, calls components, applies controls, or repeats a loop.
6. **Product:** the complete user-facing application, including its interface.

Not every system has all six. Prompts, input data, and decoding settings are additional possible locations of behavior; do not force them into “weights.”

3 Two worked examples

Read each specification and make a quick classification before reading its explanation.

3.1 Example 1 A polished help window

A window matches a user’s typed keyword against a handwritten list, then displays a stored answer. Unmatched inputs display “Please contact support.” Nothing is fitted from data.

Predict: Is the displayed prose evidence of a language model? Where would you change the fallback?

Explanation: This is rule-based lookup with text input and output. Its algorithm matches keywords; the list and answers are manually authored data. There are no trained weights or neural architecture in this specification. The window is the product interface. Changing the fallback changes software or stored content, not model training. Fluent output alone would not reveal this implementation.

3.2 Example 2 One text model call

A command-line application loads a Transformer checkpoint trained on raw text to predict each next piece from preceding text in the same document. It receives a prompt, generates one completion, and exits. There are no tools or subsequent calls.

Predict: Which labels overlap, and does the application have an agent loop?

Explanation: The model is neural, generative, and text-only; its described training is self-supervised. Transformer names the architecture; the checkpoint supplies trained weights; token selection during generation is an algorithmic step. The application supplies the prompt and invokes generation. The specified one-call workflow lacks our action-result-action loop. Changing the prompt can change the completion without changing any weight.

4 Your worksheet

Make six copies of this compact record, one for each card. Short phrases are fine, but every important label needs evidence.

```
Card ID and system boundary:
Initial prediction:
  Mechanism(s):
  Learning method, or no learning:
  Task; input and output modalities:
  Agent loop: yes / no / not established, because...
Evidence: one or two exact details from the card:
Layers: algorithm; architecture; trained parameters; inference runtime;
        orchestration; product (use absent/unspecified when justified).
Uncertainty: one unsupported claim and evidence that would settle it:
After self-check: keep/change which claim, and why?
```

Complete all initial predictions before opening the answer key. “Not established” means evidence is missing; “no” needs evidence against the claim. Treat explicit “only” and “no” statements as constraints. Do not invent hidden models. Equally, do not fill unspecified training details with a familiar implementation.

If working with an LLM, make your first attempt independently. Then ask it to challenge the evidence for one label without replacing your entire worksheet. Record any assistance in your final artifact.

5 Six system cards

5.1 A Greenhouse alarm

A dashboard receives temperature and humidity numbers. Its entire decision procedure is: if temperature is above 28°C and humidity is below 40%, display “Water check needed”; otherwise display “No alert.” Staff wrote the thresholds and messages. There is no training, search, or learned component. The dashboard does not operate watering equipment. Its next update simply evaluates the rule on the next pair of readings.

5.2 B Route desk

A route-planning application receives a start and destination in a manually entered directed graph. It exhaustively enumerates the valid routes, sums their edge costs, and displays a least-cost route. There are only two routes from S to T: S–A–T costs 3 + 4; S–B–T costs 2 + 7. Costs are fixed numbers, and no parameters are learned. The application plans a route once; it neither drives a vehicle nor observes a journey.

5.3 C Sorting gate

A recycling dashboard uses logistic regression fitted to examples of two numeric sensor readings paired with human labels, “metal” or “other.” Its learned coefficients and intercept produce a metal probability. Software displays “metal” when that probability is at least 0.60; otherwise it displays “other.” For item X the probability is 0.72. Deployment loads fixed parameters, with no updates, neural components, or search. The dashboard cannot move the physical sorting gate.

5.4 D Surface inspector

An inspection application resizes a photograph to a fixed size, passes it through a convolutional neural network, and displays a crack probability and “crack” or “clear.” The network has four convolutional layers followed by a classification layer. Its weights were fitted to photographs with human-assigned labels. There are no other models. During use, the weights remain fixed. The application receives images only; it does not accept questions, generate new photographs, or control equipment.

5.5 E Illustrated request

A drawing application receives a text request. A Transformer language model generates a structured text instruction, which software passes to a separate neural image generator. The image generator was trained using paired text and images; its exact training objective is unspecified. Software then adds a fixed watermark and shows the image. These three stages always occur once in that order, with no evaluation or retry loop. The language model never receives the generated image. Its own training history is unspecified.

5.6 F Warehouse rover

A simulated rover observes its grid cell and battery level, chooses a movement, receives the resulting state, and repeats until it reaches its assigned destination. During training, a tabular reinforcement-learning algorithm updated state-action values using rewards for arrival and penalties for wasted moves. Deployment freezes that learned table. The rover chooses the highest-valued allowed action, using a fixed tie-break. A handwritten safety rule excludes moves into reserved cells. There is no neural network, language model, or media generator.

6 Predict three interventions

Before self-checking, answer each with: **changed component** → **expected effect** → **what remains unchanged** → **evidence or uncertainty**. These are deductions from specifications, not measurements from running software.

1. **A and B:** For A, readings are 29°C and 35%. Raise only the temperature threshold from 28°C to 30°C. For B, remove edge A–T, retaining the other route. Predict each new output. Did either change require training?
2. **C and D:** Raise C’s display threshold from 0.60 to 0.80 for item X. Separately, change only D’s resizing method while keeping its output dimensions fixed. Which outcome can you determine exactly? Can either change affect displayed predictions without changing weights? Explain why D’s accuracy direction is unknown.
3. **F:** Change the reward for wasted moves in the training configuration, but do not retrain or change anything deployed. Hold observations, allowed actions, and tie-breaking fixed. Will the deployed next action change? How would actually resuming learning alter your conclusion?

7 Your artifact and review

Save one document named lab-01-map-ai-systems-response in any format you can reopen, or keep numbered paper pages. Include the six records, three intervention answers, and a final 100–150-word explanation of why “the product changed” does not necessarily mean “the model was retrained.” Preserve original predictions and append revisions instead of overwriting them.

Record the Lab title and date, any assistance, and any assumptions beyond the cards. This makes another reader able to reconstruct your reasoning. You may share the artifact with a teacher, study partner, or chosen LLM for critique. No submission to the course website is needed; sharing and publication are optional.

8 Self-check answer key

Stop here until your predictions are recorded. These explanations are reference reasoning, not invented learner submissions. Compare evidence and boundaries, not exact wording.

8.1 Card classifications and layers

A: Rule-based decision logic; no learning; numeric inputs and fixed text output; no goal-directed action loop. The threshold comparison is the algorithm, the dashboard is the product, and display/update code is orchestration. There is no neural architecture or trained parameter set. Sensor accuracy is unknown; calibration evidence would be needed. Manual threshold authorship is established.

B: Search and optimization without learning. Enumeration and minimum-cost selection are algorithms; graph edges are problem data, not learned weights. The app supplies inputs and displays a route. A planner is a component that could be placed inside an agent; this one-shot product has no execution-feedback loop. “Symbolic search” is also a defensible mechanism label.

C: Classical supervised learning for classification. Logistic regression specifies the model form; coefficients and intercept are trained parameters, even though they are not neural weights. Probability computation and the threshold rule are distinct steps. The latter lives in software. Numeric inputs and a displayed label do not imply a language model. No active gate control or agent loop is specified.

D: Neural supervised classification, with image input and probability/label output. Layer organization is architecture; fitted tensors are weights; resizing and display are surrounding software. Returning a textual label does not make the model a language generator. Its exact training optimizer is unknown. Code or training records would establish it; the visible prediction would not.

E: A neural generative pipeline and a product spanning text and images. The Transformer architecture and two trained model parameter sets are distinct from the fixed orchestration and watermark code. “Multimodal system” is justified; “the language model can see” is not. A text-conditioned image model crosses modalities, but its exact architecture and training objective remain unknown. Paired data alone does not settle the objective. There is no agent loop under this Lab’s convention.

F: Non-neural reinforcement learning combined with explicit safety rules and an agent loop. The table contains learned values; action selection and training updates are algorithms. The safety filter and repeated observation-action cycle belong to orchestration. Calling it a reactive agent is reasonable; claiming that it plans ahead is unsupported. A broader use of “weights” for learned parameters is acceptable if you explicitly distinguish the table from neural weights.

The cards do not identify particular inference runtimes or numerical settings. For C, D, and E, model execution is present but its runtime details are unspecified. F executes a learned table lookup rather than a neural inference engine; A and B have no fitted model to run. Do not infer a library or hardware platform from these descriptions.

8.2 Intervention answers

1. **A:** “No alert,” because 29 is not above 30. **B:** S–B–T, cost 9. Rules or problem data changed; neither system learned.
2. **C:** Item X switches to “other”; its probability remains 0.72. **D:** Changed preprocessing may change network inputs and predictions with identical weights. The effect may also be zero for some images. Accuracy could improve, worsen, or stay the same; labeled evaluation examples and both resizing procedures are needed to measure it.
3. **F:** The next deployed action stays the same. An unused training configuration cannot change a frozen deployed table. Resuming learning permits table changes, but neither an immediate action change nor improved performance is guaranteed.

8.3 Check the quality of your reasoning

Your artifact is ready when you can:

- support classifications with construction details rather than names or fluent output;

- identify overlapping axes and distinguish component from product boundaries;
- locate a behavior in the appropriate layer;
- distinguish absent components from missing evidence;
- predict a controlled change without claiming an unmeasured result;
- explain at least one correction, or defend an unchanged prediction with evidence.

If one criterion fails, revise that part and name the missing evidence. A well-supported “unknown” is a successful answer.