Visual Reasoning & Wranglin' GPTs

Welcome to Your First Dive into AI!

Feeling a little nervous? Don't worry—no coding experience necessary, no PhD required, and absolutely no AI expertise expected. You're in the right place, and we'll guide you every step of the way. In fact, this assignment is built to help you discover how powerful human reasoning really is. And that's why you'll do amazing—you are a human, after all!

Along the way, you'll learn to approach problems systematically, identify patterns, and communicate your ideas effectively. You'll also uncover the growing importance of Large Language Models (LLMs), like ChatGPT, and gain hands-on experience in putting them to work. With LLMs as your tool, you'll tackle challenges you might've thought out of reach, transforming your insights into real solutions. Together, we'll explore, learn, and problem-solve in ways that are both surprising and inspiring!

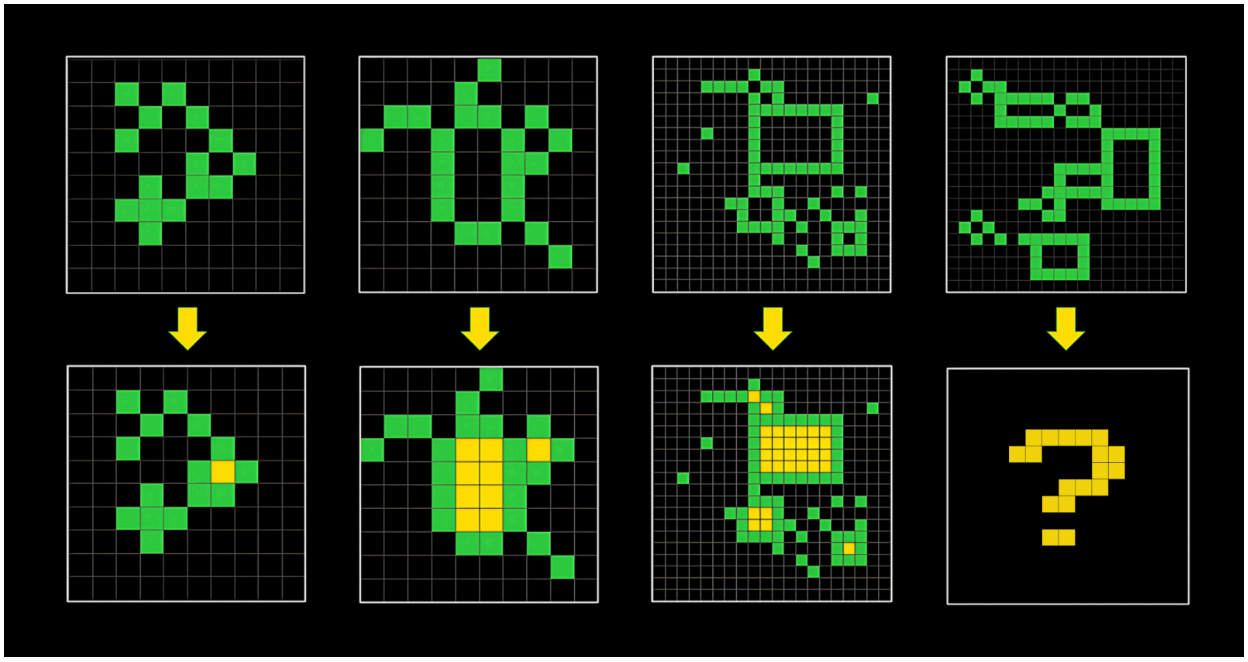

Figure 1: ARC-AGI Input, Output Example and Hidden Evaluation Test (Puzzle ID: 0962bcdd) Do you know what goes in grid 4?

For this assignment, you'll take on a task from the ARC (Abstraction and Reasoning Corpus) AGI Challenge. Each task involves transforming an input (the "before") into a specified output (the "after"). Don't stress—many of these tasks are simple enough for 5-year-olds! Yet, modern AI systems still struggle with them. For example, ChatGPT scores only 9% on the challenge. That's exactly why we're exploring these tasks: they reveal the limitations of AI and the unique strengths of human reasoning.

Ready? Let’s dive in!

Learning Objectives:

Systems Thinking: Develop a systems-thinking approach to identify visual reasoning patterns, break down challenges into manageable components, and solve problems.

Mental Modeling: Reflect on how you approach problems, identify the thought processes or mental models involved, and use your insights to develop more effective problem-solving strategies.

Communication: Master the ability to clearly explain the core elements of a problem and the steps needed to solve it by translating your ideas into step-by-step, “algorithms,” written in clear English.

Big Model Cowboycraft: Use popular chatbots like ChatGPT or Claude as tools to convert your English instructions into working Python code, and become familiar with their strengths and shortcomings.

Scalable Thinking: Explore the difference between generalized and narrow solutions. Learn to design robust, adaptable solutions that can handle a variety of problems, avoiding reliance on hard-coded values.

Assignment Background:

What is AGI?

Artificial General Intelligence (AGI) represents a system capable of acquiring new skills in unfamiliar environments without prior training. True intelligence isn’t about mastering specific tasks—it’s about generalizing efficiently and adapting to entirely novel challenges.

Skill alone isn’t intelligence. Skills can often be “bought” through extensive training data or prior knowledge, masking a system’s real ability to generalize. Intelligence, as François Chollet defines it, is “skill-acquisition efficiency across a scope of tasks, relative to priors, experience, and task difficulty.”

In simple terms, AGI means creating systems that learn to solve problems their developers didn’t anticipate. Such systems could adapt to new environments, think flexibly, and invent solutions to novel challenges.

What is ARC?

The Abstraction and Reasoning Corpus (ARC) is a benchmark specifically designed to measure progress toward AGI. Unlike traditional AI tests, which focus on task-specific skills, ARC-AGI evaluates generalization—how well a system can infer new rules and solve problems with minimal examples.

ARC tasks mimic human reasoning and emphasize skill acquisition and abstraction over brute-force learning. This makes ARC-AGI a unique benchmark for guiding the development of truly intelligent systems.

ARC-AGI Design

ARC-AGI tasks consist of grids made of cells, where each cell is one of ten possible colors. Tasks require transforming an input grid into an output grid based on a hidden rule inferred from training examples.

Each task includes:

1. Training pairs: Examples showing input-output transformations.
2. Test pairs: Inputs where the correct output must be predicted.

Successful solutions must be pixel-perfect, including grid dimensions and precise color placements. Grids can range from 1x1 to 30x30 in size.

The Data Format

Tasks are stored in JSON format with two main sections:

1. train: A list of input-output pairs (usually 2-10). These provide examples to infer the rule.
2. test: A list of inputs for which the output must be predicted.

Example:

```
{
  "train": [
    {"input": [[1, 0], [0, 0]], "output": [[1, 1], [1, 1]]},
    {"input": [[0, 0], [4, 0]], "output": [[4, 4], [4, 4]]}
  ],
  "test": [
    {"input": [[0, 0], [0, 8]], "output": [[8, 8], [8, 8]]}
  ]
}
```

In this example, the rule involves duplicating the non-zero color across the entire grid.

Impact of Solving ARC-AGI

Cracking ARC-AGI would be a groundbreaking step toward AGI. A successful solution would:

1. Create systems capable of adapting to entirely new tasks without retraining.
2. Enable anyone, regardless of technical expertise, to program systems by providing simple input-output examples.
3. Revolutionize software and automation, making programming as accessible as teaching by example.

This would result in a new programming paradigm, transforming how we interact with technology. Solving ARC-AGI would be as impactful—if not more so—than the invention of the Transformer, opening up entirely new fields of research and application.

ARC-AGI represents more than a benchmark—it's a path to redefining intelligence and building systems that think like us. Are you ready to take the first step?

Let's Go.

Step 1: Get Your Hands Dirty

Instruction:
Visit https://arcprize.org/play and try out the daily puzzle. Don't worry about being perfect—just explore and see what patterns you can find.

Ask yourself:

- What's happening in the puzzle?
- Can you spot any patterns, like shapes or colors that repeat?
- What changes from the input to the output?

Explore more by clicking the dropdown in the top-right corner and selecting "Public Training Set (Easy)" or "Public Evaluation Set (Hard)." How many can you solve?

If you beat ChatGPT's 9%, congratulations—you've outsmarted a multi-billion-dollar AI model. Go ahead, brag a little.

Step 2: Reflect on Your Thinking

Take a moment to think about how you naturally notice patterns. For example:

- Do you see objects in the grid? (Shapes or groups that feel connected.)
- Are you focusing on how colors or lines stand out?
- Are there clear relationships, like objects being next to or stacked on each other?

Ask yourself:

- What helps me spot patterns quickly?
- How do I decide what's important to notice?

These are skills you use without even thinking about them, but they're powerful tools! Try explaining your thought process to someone else. Try to describe what the core skills are, and how they relate.

Imagine that you're teaching a blind alien who doesn't know anything about objects, colors, or patterns. Can you explain what an "object" is? Or why one square feels more important than another?

Think of these basic ideas—shapes, colors, groups, positions—as your human "toolkit" for solving problems.

Step 3: Break It Down

Instruction:
Pick a simple puzzle and study the "before" and "after" grids.

Ask yourself:

- What’s in the grid? Are there shapes, colors, lines, or empty spaces?
- What changes from the input to the output? Do shapes move? Do colors change?
- Can you group parts of the grid? Are there clusters that seem connected?
- How are the pieces related? Do they align, mirror, or repeat?

Now, try writing the rules step by step. For example:

1. Find all red squares.
2. Move each red square up by one row.
3. Add a green square to the right of each red square.

Ask yourself:

- Which adverbs and adjectives do you use to describe objects and change?
- Can you think of a way a computer could “sense” or “detect” these same things?
- Could you describe the rules in numbers or positions, like rows and columns?

If your explanation includes “and then magic happens,” keep working.

Step 4: Identify and Write Big Steps

Instruction:
Look at your solution and break it into big, overarching steps. Imagine you’re explaining what to do, one major task at a time, to someone who has no context. Write these steps in clear, plain English.

For example:

1. Make a new grid the same size as the input.
2. Identify all yellow triangles.
3. Rotate each yellow triangle 90 degrees clockwise.
4. Place a blue square in the position where the yellow triangle started.

Ask yourself:

- What are the main actions that need to happen?
- Does each step describe something meaningful and clear?

- If you read these big steps to a partner, would they understand your solution without explanation?

Once the big steps are written, stop and read them back. Does the sequence make sense as a whole? If not, adjust before diving deeper.

If one of your big steps is “Make it work somehow,” you’re going to confuse your partner and probably a few aliens.

Step 5: Break Big Steps into Substeps

Instruction:
Now take each of your big steps and break it into smaller, detailed substeps. This is where you explain exactly how to do each part. Write these substeps as if teaching someone who doesn’t know what a square or triangle is.

For example:
Big Step: Identify all yellow triangles.
Substeps:

1. Look at each square in the grid, one by one.
2. If a square is yellow mark it.
3. For all yellow squares, check surrounding squares for yellow squares.
4. For all groups of yellow squares
5. Use its position (row and column) to remember where it is.

Big Step: Rotate each yellow triangle 90 degrees clockwise.
Substeps:

1. Find the position of the triangle you marked earlier.
2. Change the triangle so its point moves to the right.
3. Keep its yellow color the same.

Now, test your substeps:

1. Read them to a partner.
2. Hand them a screenshot of a grid and have them follow your steps exactly—no guessing or adjustments.

Ask yourself:

- Did the instructions work perfectly?

- Were there any moments where your partner got confused or had to guess what you meant?
- Was it too specific (only working for one example) or too vague (missing key details)?
- How can you refine the steps to avoid confusion?
- How could you adjust the rules to handle every case?

These updated steps emphasize breaking solutions into manageable chunks and refining the details through feedback.

If your partner spends five minutes asking, “What’s a row?” you may need to explain some basics.

Step 6: Put GPT to Work

Instruction:
Take your step-by-step instructions and ask ChatGPT (or another AI) to write Python code for them. Test the code with your puzzle examples.

Ask yourself:

- Did the code match my instructions?
- If it didn’t work, were my steps unclear, or did the AI misunderstand?

If the code doesn’t work, tweak your instructions or add more detail. Machines aren’t mind readers—they only follow exactly what you tell them.

If GPT writes you a poem about grids instead of code, it’s time to rephrase your steps.

Step 7: Reflect on What You’ve Learned

Instruction:
Look back at the puzzle and your process. Think about what worked, what didn’t, and how you solved the problem.

Ask yourself:

- How did I figure out what mattered in the grid?
- How did I use relationships like position, color, or symmetry to explain the solution?
- Could I improve how I teach the AI to solve similar puzzles?

If you find yourself explaining grid symmetry to your pet, you’re officially in deep.

Setup Requirements: Step-by-Step Guide for Intro to AGI

To get started, we need to set up your computer to work with the repository for this assignment. Follow the Step-by-Step Guide and when your setup is complete, we'll dive into solving tasks and learning how to think like an AGI researcher!

Jump to Mac Installation
Jump to Windows Installation

For Mac:

---

1. Open the Terminal

The Terminal lets you type commands to control your computer:

1. Press **Spotlight Search** (magnifying glass in the top-right) and type **"Terminal"**.
2. Press **Enter** to open it.

---

2. Install Homebrew

Homebrew makes it easy to install software.

1. Check if it's already installed by copying and pasting this into your terminal and pressing enter:

```
brew --version
```

2. If you see a version number, skip to Step 3. If not, install Homebrew:

```
/bin/bash -c "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/HEAD/install.s
h)"
```

Important: Only run commands from trusted sources, as harmful ones can damage your computer.

---

3. Install Git

Git is a tool for downloading and managing files.

1. Install Git:

```
brew install git
```

2. Verify installation:

```
git --version
```

---

4. Download the Course Files

1. Now, download the repository with the course materials. Run this command:

```
git clone https://github.com/commotum/intro2agi.git
```

This creates a folder called `intro2agi` with all the course files.

---

5. Confirm Setup

1. Navigate into the folder:

```
cd intro2agi
```

2. List the files:

```
ls
```

If you see course files, you're ready to start!

For Windows:

---

1. Install Git for Windows

Git is a tool for downloading and managing files. Here's how to install it on Windows:

1. Go to https://gitforwindows.org/ and download the installer.
2. Run the installer. During the installation:
   a. Keep the default settings unless you know what you're doing.
   b. Ensure the "Git Bash Here" option is checked so you can use a terminal-like tool.
   c. Finish the installation and open "Git Bash" (your terminal for Git).

---

2. Verify Git Installation

1. Once Git Bash is open, check that Git is installed by typing:

```
git --version
```

If you see a version number, Git is installed correctly. If not, go back to Step 1.

---

3. Download the Course Files

Now, download the repository with the course materials.

1. Run this command in Git Bash:

```
git clone https://github.com/commotum/intro2agi.git
```

This creates a folder called `intro2agi` in your current location, containing all the course files.

4. Confirm Setup

Once the repository is cloned:

1. Navigate into the folder:

   ```
   cd intro2agi
   ```

2. List the files:

   ```
   ls
   ```

   If you see course files, you’re ready to start!