**Methodology – First Draft**

**Research Design and Framework**

Our first step during data collection was figuring out how to organize the types of questions we wanted to ask. At first, we thought about putting everything on a single spectrum from logical to creative. That seemed straightforward, but when we tried it in practice, we noticed that many questions didn’t fit neatly on that line or into the sections we initially tried to make. To create more structure, we designed a four-quadrant graph, using two axes:

* The x-axis ran from creative inference to logical strategies.
* The y-axis ran from specific wording to vague wording.

By crossing these two axes, we created four categories of questions:

* Factual Questions (logical + specific): questions with clear, fact-based answers.
* Logical Questions (logical + vague): reasoning-based questions that allow some flexibility.
* Subjective Questions (creative + specific): opinion-driven questions with precise framing.
* Creative Questions (creative + vague): open-ended, imaginative questions.

This system gave us a consistent way to sort and design our prompts. It also made later analysis easier, since we could directly compare how different models performed across categories and add more data within a specific axis if we noticed anything interesting or worth exploring further.

**Selection of Models**

When deciding which large language models (LLMs) to use, we wanted a mix that reflected both popularity and diversity. We specifically chose ChatGPT-5, DeepSeek-V3.2-Exp, Gemini 2.5 Flash, Llama 4 (Meta), and Grok 4.

* ChatGPT-5: widely used and familiar to many, making it a good baseline.
* DeepSeek-V3.2-Exp: experimental and less mainstream, included to provide contrast.
* Gemini 2.5 Flash: designed for speed and adaptability, which might affect its style of answering.
* Llama 4 (Meta): an open-source option from a major company, allowing us to see differences between closed and open approaches.
* Grok 4: relatively new but distinct, adding variety to the dataset.

By selecting across this range, we ensured that our results wouldn’t just reflect one type of model design or one company’s approach.

**Data Collection Procedure**

Because this was a two-person project, we wanted to make sure our process didn’t just split the workload but also generated richer data. To do this, we each submitted the same set of prompts to the models, but in two different ways:

* Hayden asked each question in a brand-new chat, so every response was isolated from the others. This helped us see how models performed without any prior context.
* Smera asked multiple questions (the same set of prompts) within the same chat before starting a new one, so the models could potentially build on earlier answers. This helped us see how continuity and memory influenced responses.

Each of us entered every prompt five times per model, which gave us repeated samples. Doing this reduced the chance that our results were skewed by randomness and allowed us to check for consistency in the answers.

**Controlling for Variables**

To make the study fair and replicable, we kept the settings consistent across all models. We used the default versions of each model without adjusting advanced options. In particular, we chose not to change the temperature setting, which controls how random or creative a model’s outputs can be. We also avoided using APIs, because we wanted our data to reflect the type of experience a regular user would have when using these tools. By not altering these variables, we hoped to minimize external factors that could influence the outputs.

**Development of Prompts**

The prompts themselves were developed collaboratively. We began by brainstorming a wide range of potential questions for each of the four categories. After generating a large number of questions, we went through rounds of discussion to refine the list, eliminate redundancy, and make sure the questions truly matched the categories we had designed.

We ended up with 36 final prompts, divided evenly across the categories (about 9–15 per type). This balance allowed us to test all four categories without overemphasizing one.

**Data Organization and Storage**

All responses were carefully stored and organized in an Excel spreadsheet to keep the dataset clean and structured. Each entry included the following information:

* Prompt (the question asked)
* Model (the LLM used)
* Prompt Type (factual, logical, subjective, or creative)
* Response Number (1–5 for repeated trials)
* Response Text (the model’s actual answer)
* Researcher Identifier (whether Hayden or Smera submitted it)
* Date (the day the entry was collected)

This setup allowed us to trace every response back to its context, making it easier to identify patterns and revisit the data when needed. Additionally, our data was stored in Excel in a format that made it ready for further processing and analysis.

Three additional columns were found from the data and stored in the spreadsheet:

* Response length in characters
* Response length in words
* The vector embedding of the response

These columns are stored for ease of use later, particularly in the case of the embedding, which would take a long time to find if we had to embed the response every time.

**Data Manipulation**

As is standard for analysis of text similarity in the age of NLP, we find similarity by finding the cosine similarity of text embedding vectors.

All embeddings are created through the same sentence transformer model, so that each resulting vector present in the dataset is of the same format. Such vectors are quite large and illegible to a human reader, but maintain the semantic meaning of their input. For example, the vectors for the sentences "I will read a book tonight." and "I read a book last week." are somewhat similar (our model gives a similarity of 0.68), but when comparing that first embedding with the embedding for the sentence "The sensor will read the temperature." the two have a very low similarity (0.068).

**Similarity Measure**

To find similarity between two vectors, we use the following formula:

This is the standard equation for finding the angle between vectors. The equation produces a value between -1 and 1, with negative values indicating opposite direction and positive values indicating high similarity, with low values (close to 0) indicating that the two vectors are unrelated.

For our purposes, this means that we are examining the similarity of vector directions, which has been examined to correlate with the semantic meaning of their respective strings. The magnitude of the vectors do not matter, only their direction. This means that the embedding for “Apple” has the same similarity to an unknown vector as the same vector rescaled by a constant factor.

In theory, this equation produces a normalized number between -1 and 1, but in practice we only obtain similarity scores between 0 and 1. For our purposes, a value of 0 indicates complete dissimilarity (impossible to achieve in practice), a value between 0 - 0.3 indicates dissimilarity, a value between 0.3 - 0.6 indicates slight similarity, a value between 0.6 - 1 indicates strong similarity, and a value of 1 indicates that the two vectors being compared are the same. We will see many ‘1’ values later when we plot out the similarity of all vectors in the dataset.