

How does modern generative AI actually work?

15-113 – Monday 4/20

Final week!

<https://bit.ly/41FYOXZ>



Reminders!

Today is the last day to do your capstone checkin (and the only remaining time to do those will be in my office hours, from 2pm-3:30pm)

The project itself is due Thursday at 8pm. Don't wait to the last minute!

- Significant late deductions will apply after 8pm Thursday and on Friday
- **No submissions will be accepted after Friday** (we have to start grading them right away)

Remember to **watch Ed and sign up for your final presentation promptly.** This counts as one of only two grades in the quiz/exam category (along with the earlier Project 2 presentation).

What's up today?

- A quick look at the [RPG we started making last week](#)
- An overview for how modern generative AI and LLMs work
- Working time / time for a small number of checkins

First, a follow-up on agent teams

- Recall that last week we tried to make a role-playing game with an **Claude agent team** consisting of:

Agent	Primary focus
ux-designer	Screens, menus, input flow, visual layout
story-writer	Dialogue, world lore, quest design
qa-tester	Bug hunting, edge cases, test suites
player-sim	Simulates player archetypes, usability feedback

- The first iteration only finished at the end of class, **drained Claude Pro in two prompts** (30% making the team, 70% making an initial attempt to create the game)

The first attempt was shiny but not good

- It managed to make a decent title screen
- Plenty of story exposition (though it lifted the character names *directly* from Phantasy Star, an old Sega Genesis game)
- No free-roaming; just one instance of combat that can always be won by mashing 'Enter'
- Generated a seemingly large amount of unused code
- I suspect asking it to create a good demo with no well-defined spec and no well-defined end point is partly to blame

What I wanted, what we got:

- Ideally looking for a game [somewhat like this](#)
 - Just a demo though; these games are huge
 - Also we shouldn't expect sprites/art. Generating images is still expensive and AI usually uses basic shapes unless explicitly asked to use images.
- I made about three more naive “fix bugs and keep going” prompts to finish the demo. Each prompt immediately drained my Claude limits for 5 hours.
- The result is... technically a game, and impressive from an “AI can create basic games now” perspective... but [I wouldn't describe it as fun, clever, or well-designed.](#)

A few screenshots for posterity

STARFALL CHRONICLES

A Phantasy Star-Inspired Adventure

DEMO v0.3

Crash Site

A starship from another age... and a destiny.

Crestfall

Crash Site

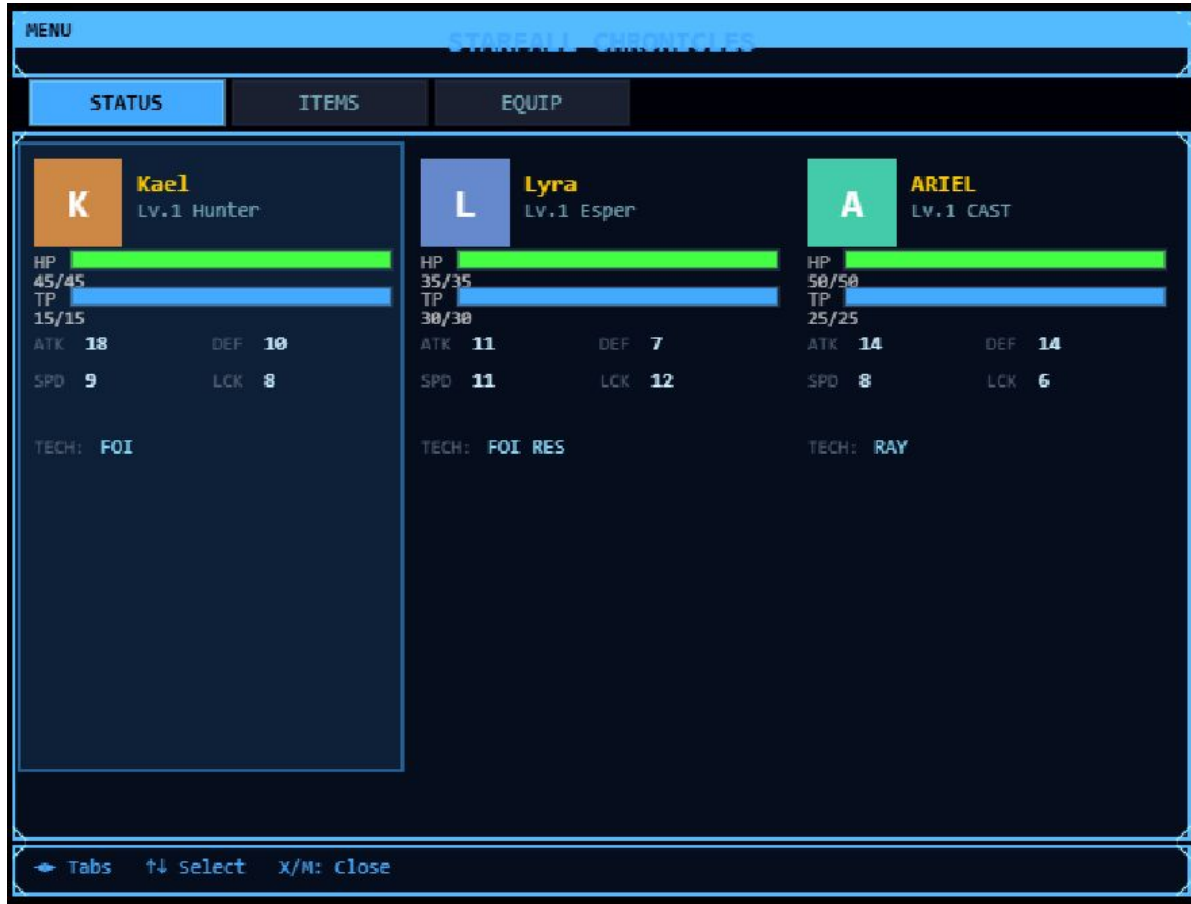
Deionis Woods [X]

Spaceport [X]

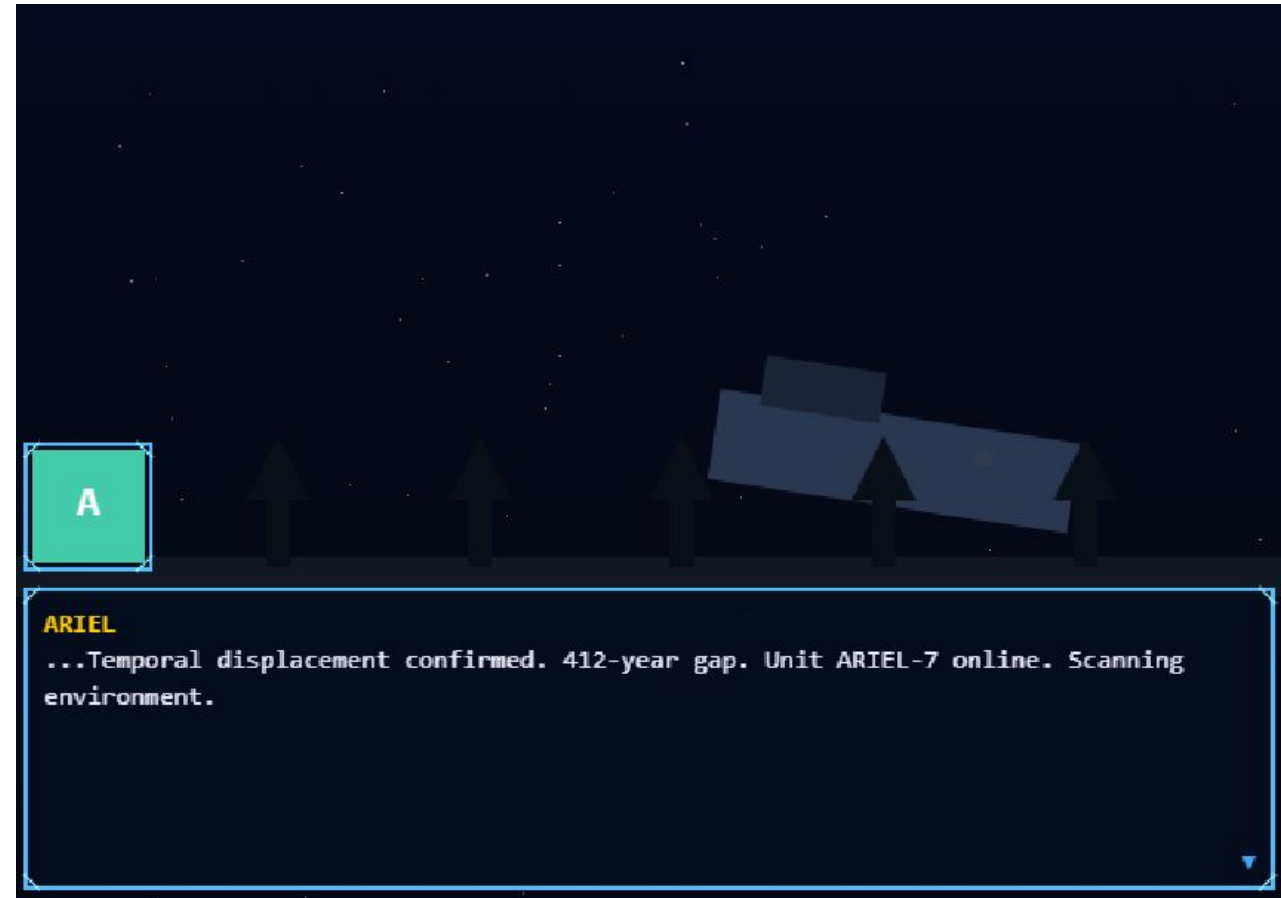
← Move ENTER: Interact M: Menu

Kael Lyra ARTE

A few screenshots for posterity



A few screenshots for posterity



So what do we think?

- I'd call this 'game-shaped' at best. Poor mimicry of true craft.
- And maybe there's some comfort in knowing that at this level of complexity, even these powerful tools will struggle to make something *good* unless we bring our skills in planning, thoughtfulness, and creativity

New Topic: How does modern GenAI work?

- This is going to be a *very* high-level overview. For a deeper look, you'll want Prof. Zico Kolter's 10-202 Introduction to Modern AI:
<https://modernaicourse.org/>
- And there are many other courses that dive into specific aspects like natural language processing and neural nets:
<https://ai.cmu.edu/curriculum>

Start with autocomplete

When I say each sentence, fill in the empty words:

- Tepper School of _____
- College of Fine _____
- Don't forget to tie your _____
- Put the fries in the _____
- Have a good _____

Start with autocomplete

If you answered 'bag' confidently, it's probably because you've seen a specific meme with this phrase, but not everyone has.

- Put the fries in the (bag)

This one's less definitive because there are many answers that are both plausible and common.

- Have a good (night/evening/day/flight/trip...)

In any case, you probably **answered with the most *probable* word based on your experience.** Your phone's autocomplete does the same thing.

Simplistic model: What word most commonly follows the previous word?

Read a bunch of text, count how many times word A follows word B, store those counts in something like a nested dictionary.

```
counts = {"fuzzy": {"bunny": 1402,  
                  "sock": 349,  
                  "bread": 78,  
                  "phone": 1, ...},  
         "potato": {"chip": 1809,  
                  "bread": 802,  
                  "bug": 283,  
                  "steven": 23, ...},  
         ...}
```

After reading, turn this into probabilities (i.e. probability of “steven” given “potato” would be very low, but “chip” is high)

More example text means better predictions, but they won't ever be great

Embeddings: Mathematically represent word *meaning and context* as a vector

Each component contains information

```
embeddings = {  
    "cat":    [0.9, 0.1, 0.8, 0.0, ...],  
    "dog":    [0.9, 0.1, 0.7, 0.0, ...],  
    "apple":  [0.1, 0.0, 0.1, 0.9, ...],  
    "student": [0.2, 0.9, 0.3, 0.1, ...],  
}
```

Perhaps $L[0]$ is how ‘animal’ something is, and $L[1]$ is how knowledgeable something is

In an N-dimensional vector space, similar words are close together, even if not truly synonyms (like ‘dog’ and ‘cat’)

Embeddings

Using these vector embeddings means our predictions can be based on observed patterns for similar words, like “overcast” and “cloudy”

A cool thing: Adding and subtracting vectors can make intuitive sense with a good embedding.

king - man + woman $\approx$ queen

paris - france + italy $\approx$ rome

Embeddings are trained on massive amounts of data!

Attention: Look at all words, not just the last

First, rather than predicting the next word based on just the last word, look at the last N words instead

- Embeddings make this more feasible since we won't have to find an *exact* word match in training

Attention determines which words are *most important* to each other word in the sentence.

If I mention a **king sized mattress**, the presence of **sized** and **mattress** tells me that I'm probably not talking about royalty. I can use that information to update the vector for **king** in this sentence. (Simultaneously, note that '**a**' contributes almost nothing of meaning here.)

Now we have an **LLM: Large Language Model**

Goal: Predict the next piece of text given prior text

- Step 1: Turn the text into **embeddings** (vectors)
- Step 2: Use a **transformer** to update the embeddings based on **attention** and context
 - Repeat this a bunch of times to continue enriching the embeddings
- Step 3: **Predict** the next text given the fully updated embeddings

Many of these steps use neural networks to learn complex patterns in training data

An autocomplete is not a dialogue, though

To turn this into a chatbot, you need to impose additional structure and fine-tuning.

This video I shared on Ed explains how this process (ideally) worked for GPT-2:

https://www.youtube.com/watch?v=qV_rOlHjvvs

Key idea:

Reinforcement Learning from Human Feedback (RLHF)

All of this is a simplified explanation

Simplifications have been made!

- LLMs deal in **tokens**, which are not necessarily equivalent to single words
- Guardrails add additional operational complexity
- Agents use **tools** to provide additional context and to act on other systems
- Newer models have additional steps in training and operation

...But hopefully this explanation makes it all feel a little less magic

...Explainability

Even though these systems aren't magic, their behavior is not strictly explainable

Embeddings are learned, not designed. Vector components do not necessarily map to something easily describable.

Transformer-enriched embeddings are even less meaningful to us as humans

Neural nets are composed of countless weights that get automatically adjusted to patterns we may not notice

The problem:

If an AI gives us a response that makes sense to us, we can believe that it found patterns that indicated a sensible response, but we can't peer into the LLM and discover what those patterns are.

If a response *doesn't* make sense, this is a hallucination.

...and there doesn't seem to be an easy way to have an AI explain itself, either. At best, it gives the illusion of a plausible explanation.

We'll leave it at that for now!

- Disclaimer: These explanations are based on my current understanding, and my experience with traditional machine learning. If you want the fully accurate and most up-to-date info, there are many experts at CMU who research these systems daily!
- Hopefully I didn't ramble too long; let's get to work on those capstones!