

Using Large Language Models in Data Engineering Tasks

Sean Jensen-Grey
Vince Gonzalez



- Some Terminology
- Some Context, Grounding
 - Re-evaluate your evaluation
- Prompt Engineering
- Use Cases
- Risks
- Outro



Goal



- Spark your curiosity
- Encourage new workflows
- Operate at higher semantic level
- Temper the hype
 - But also the detractors



Terminology



- NLP - Natural Language Processing
- LLM - Large Language Model
- Transformer
 - "Attention Is All You Need" by Vaswani et al., published in 2017. NLP LLM architecture based on Self-Attention
- GPT - Generative Pre-trained Transformer
- RLHF - Reinforcement Learning
 - from Human Feedback
- Context Window - working memory in tokens
- Token - Roughly a word, part of a word
- Not an LLM introduction
 - Stanford Webinar - GPT-3 & Beyond by Christopher Potts
 - <https://www.youtube.com/watch?v=-lnHHWRCDGk>





Terminology (2)



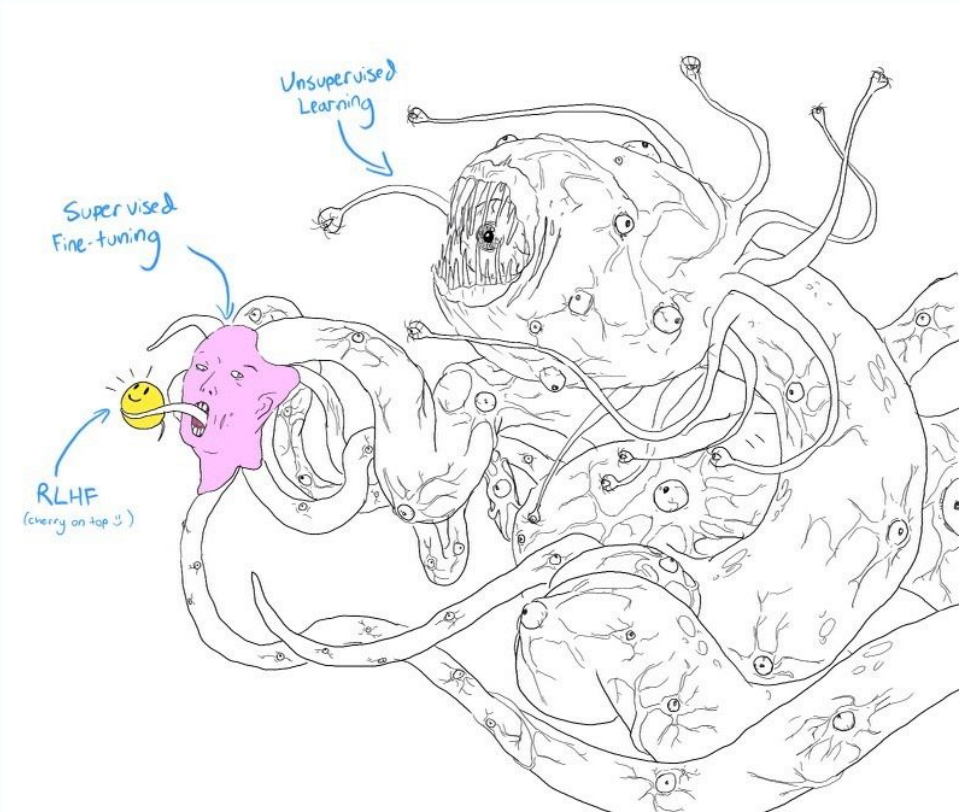
- ChatLLM (catch all for Bard, ChatGPT-4, Claude)
- Not ChatLLM specific
 - Everything is changing so quickly
 - Talk is about techniques and approaches
- Chat Model : LLM + RLHF
 - saving the funny RLHF Meme for a later slide



Understanding LLMs + RLHF



- LLM is the raw model
- RLHF
 - Reinforcement
 - Learning (from)
 - Human
 - Feedback
- RLHF turns the LLM
 - into the Chat Bot
- Everything
 - is still under there
- Image Credit
 - @anthrupad
 - "Shoggoth with Smiley Face"





ChatLLM (Re)Evaluation



- Let me bring you back
- I roughly see three patterns of evaluation
 - Pick it up, give it some "tests" and then discount it entirely
 - "Just a glorified Markov Generator"
 - "It has no ___"
 - Consciously or unconsciously, these folks seek to discount it
 - Give it some simple prompts, gets a mediocre answer
 - "Meh, kinda a toy"
 - "Good for Jokes, but what else?"
 - Probes it like a curious artist-scientist
 - Explores its limits
 - Finds where it is applicable
 - and where it is not



ChatLLM Evaluation (2)



- Pick it up, write a lousy prompt in an area they are an expert in and declare
 - "This thing is trash! It can't even!"
- What are you evaluating?
 - Ego protection?
 - How to justify avoiding it?
- It is an amazing new tool
 - The nature of technology work just had a
 - before and after event
 - step function
- It has breadth as well as depth.
- You aren't great at everything either
- Figure out how to use it



The top ChatLLM right now, ChatGPT4



	GPT-4 Evaluated few-shot	GPT-3.5 Evaluated few-shot	LM SOTA Best external LM evaluated few-shot	SOTA Best external model (incl. benchmark-specific tuning)
MMLU [43] Multiple-choice questions in 57 subjects (professional & academic)	86.4% 5-shot	70.0% 5-shot	70.7% 5-shot U-PaLM [44]	75.2% 5-shot Flan-PaLM [45]
HellaSwag [46] Commonsense reasoning around everyday events	95.3% 10-shot	85.5% 10-shot	84.2% LLaMA (validation set) [28]	85.6 ALUM [47]
AI2 Reasoning Challenge (ARC) [48] Grade-school multiple choice science questions. Challenge-set.	96.3% 25-shot	85.2% 25-shot	85.2% 8-shot PaLM [49]	86.5% ST-MOE [18]
WinoGrande [50] Commonsense reasoning around pronoun resolution	87.5% 5-shot	81.6% 5-shot	85.1% 5-shot PaLM [3]	85.1% 5-shot PaLM [3]
HumanEval [37] Python coding tasks	67.0% 0-shot	48.1% 0-shot	26.2% 0-shot PaLM [3]	65.8% CodeT + GPT-3.5 [51]
DROP [52] (F1 score) Reading comprehension & arithmetic.	80.9 3-shot	64.1 3-shot	70.8 1-shot PaLM [3]	88.4 QDGAT [53]
GSM-8K [54] Grade-school mathematics questions	92.0%* 5-shot chain-of-thought	57.1% 5-shot	58.8% 8-shot Minerva [55]	87.3% Chinchilla + SFT+ORM-RL, ORM reranking [56]

Table 2. Performance of GPT-4 on academic benchmarks. We compare GPT-4 alongside the best SOTA (with benchmark-specific training) and the best SOTA for an LM evaluated few-shot. GPT-4 outperforms existing LMs on all benchmarks, and beats SOTA with benchmark-specific training on all

From GPT4 paper



ChatLLM Evaluation (3)



When you skip the tutorial:





But it isn't great at my area of expertise!



- It is ok/good at an enormous breadth of things
- You are the expert
- Have it help you where you aren't great
 - Testing
 - Documentation
 - Code
 - Boilerplate/Skeletonizing Projects
 - Custom project templates
 - Translating idioms
 - Assist on your 2nd or 3rd order languages
 - JS/HTML/CSS, Bash, Java, JQ, etc.
 - Translating Jargon
 - Getting up to speed on a new domain
 - Test Data Extraction, Data Generation



Prompt Engineering



- Crafting and optimizing input queries to achieve the desired output from a ChatLLM
- Boils down to
 - How to Ask a Question
- See ESR's, "How To Ask Questions The Smart Way"
 - <https://wiki.c2.com/?HowToAskQuestionsTheSmartWay>
- Bloom's Taxonomy
 - Remember, Understand, Apply, Analyze, Evaluate, and Create



- Source
 - Sets the context for who is answering the question
 - Project the LLM into the correct space
 - Embody the knowledge of a PhD level ____ skilled in the use of ____
- The Question
- Destination
 - Who is receiving the question and what the expectations are
 - ELI5
 - Using analogy and metaphor explain it to a five year old
 - ELI20
 - An advanced undergraduate skilled in discrete math and physics
 - PhD
 - A PhD with an advanced degree in the hard sciences but in a different domain from this problem.



Prompt Engineering



- Clarity and Precision
- Context and Background Information
 - Personas
 - Create "The Expert", project the LLM into a space where it becomes the kind of person you would ask the question in the real world
 - Target the receiver, is that person an expert or novice? What knowledge to they have and what kind of response do they expect
 - Base Data, Context Setting, Jargon and Terminology
- Brevity
 - Don't overload the model, don't exhaust the Context Window



- Externalize the Reasoning
 - Step By Step
 - Show Your Work
 - Chain of Thought
- Shots (Examples)
 - Zero Shot
 - Please Translate "Hello! How are you into French"
 - 1, Few, Many Shot
 - 1: Here is a picture of cat, find me more cats
 - Few: Here are some positive, neutral and negative comments, please classify
 - Many: Here are thousands of pictures of digits, please label

Prompt Engineering - Step by Step



"Can you explain to me the step-by-step process to bake a vanilla cake?"

1. Gather your ingredients: The ingredients you'll need include 2 cups of sugar, 1 cup of butter, 4 eggs, 4 teaspoons of vanilla extract, 3 cups of all-purpose flour, 3.5 teaspoons of baking powder, and 1 cup of milk.
2. Preheat your oven: Preheat your oven to 350 degrees F (175 degrees C). Grease and flour a 9x9 inch pan.
3. Mix sugar and butter: In a medium bowl, cream together the sugar and butter.

And so forth ...

Prompt Engineering - Show Your Work



"Solve the quadratic equation $3x^2 + 4x - 4 = 0$ and show your work."

Let's plug these values into the quadratic formula:

$$\begin{aligned}x &= \frac{-(-4) \pm \sqrt{(-4)^2 - 4(3)(-4)}}{2(3)} \\&= \frac{-4 \pm \sqrt{16+48}}{6} \\&= \frac{-4 \pm \sqrt{64}}{6} \\&= \frac{-4 \pm 8}{6}\end{aligned}$$

Now, we'll solve for the two solutions separately:

For the positive square root:

$$\begin{aligned}x &= \frac{-4 + 8}{6} \\&= \frac{4}{6} \\&= \frac{2}{3}\end{aligned}$$

For the negative square root:

$$\begin{aligned}x &= \frac{-4 - 8}{6} \\&= \frac{-12}{6} \\&= -2\end{aligned}$$

Therefore, the solutions to the quadratic equation $3x^2 + 4x - 4 = 0$ are $x = \frac{2}{3}$ and $x = -2$.



Prompt Engineering - Chain of Thought



"Chain of thought" prompting refers to guiding the model through a series of connected prompts that build on each other.

What are some general factors that can contribute to the fall of a civilization?

...

Thank you. Now, regarding the Roman Empire, what were some of the specific factors that led to its fall in 476 AD?

...

Interesting. Could you elaborate on how the division of the Roman Empire made the Western Roman Empire more vulnerable?



Prompt Engineering - Final



- Single best resource
 - <https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/>
- <https://www.promptingguide.ai/>
- <https://platform.openai.com/docs/guides/gpt-best-practices>
- Prompt Testing
 - <https://github.com/typpo/promptfoo>





ChatLLM - All Purpose Tool with no Analog



- Language Translation
- Sentiment Analysis
- Text Summarization
- Data Generation
- Data Extraction, Filtering, Recommendation
- Data Translation
- All of those things at once
 - Programmable with human language



Use Cases



- Extracting data from text
- Plotting data
- Generating fake data
- Code Understanding
- Anything you can make work



Data Extraction



Total number of papers for "Machine Learning" posted to arxiv.org in that year.

<https://arxiv.org/list/cs.LG/22> 28644

<https://arxiv.org/list/cs.LG/21> 26513

<https://arxiv.org/list/cs.LG/20> 25887

<https://arxiv.org/list/cs.LG/19> 19253

<https://arxiv.org/list/cs.LG/18> 10491

<https://arxiv.org/list/cs.LG/17> 5232

<https://gist.github.com/seanjensengrey/92a6c19520f36798aeab0c3b0911f9a2>



Data Extraction



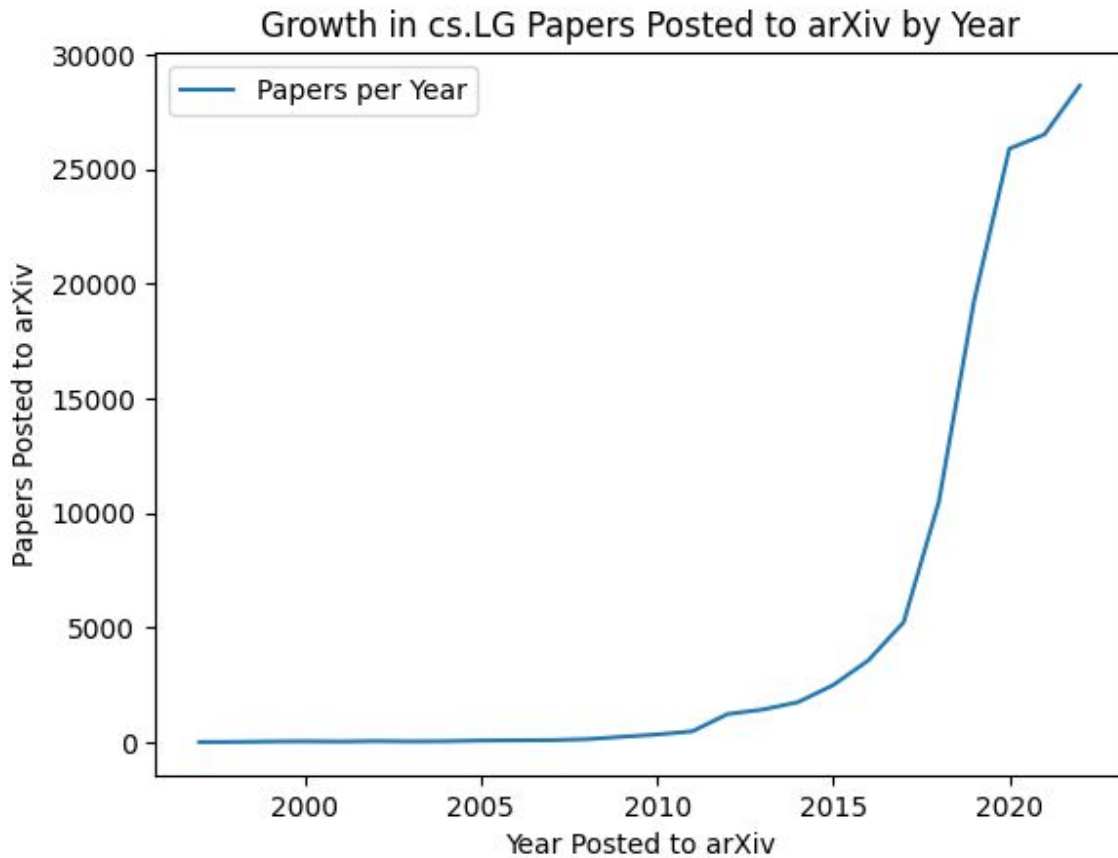
Prompt

I am going to paste a block of URLs followed by a space and a number. The last part of the URL is the year and then space followed by the number is the number of papers in that year. Please extract the data and then order by year, soonest to latest.

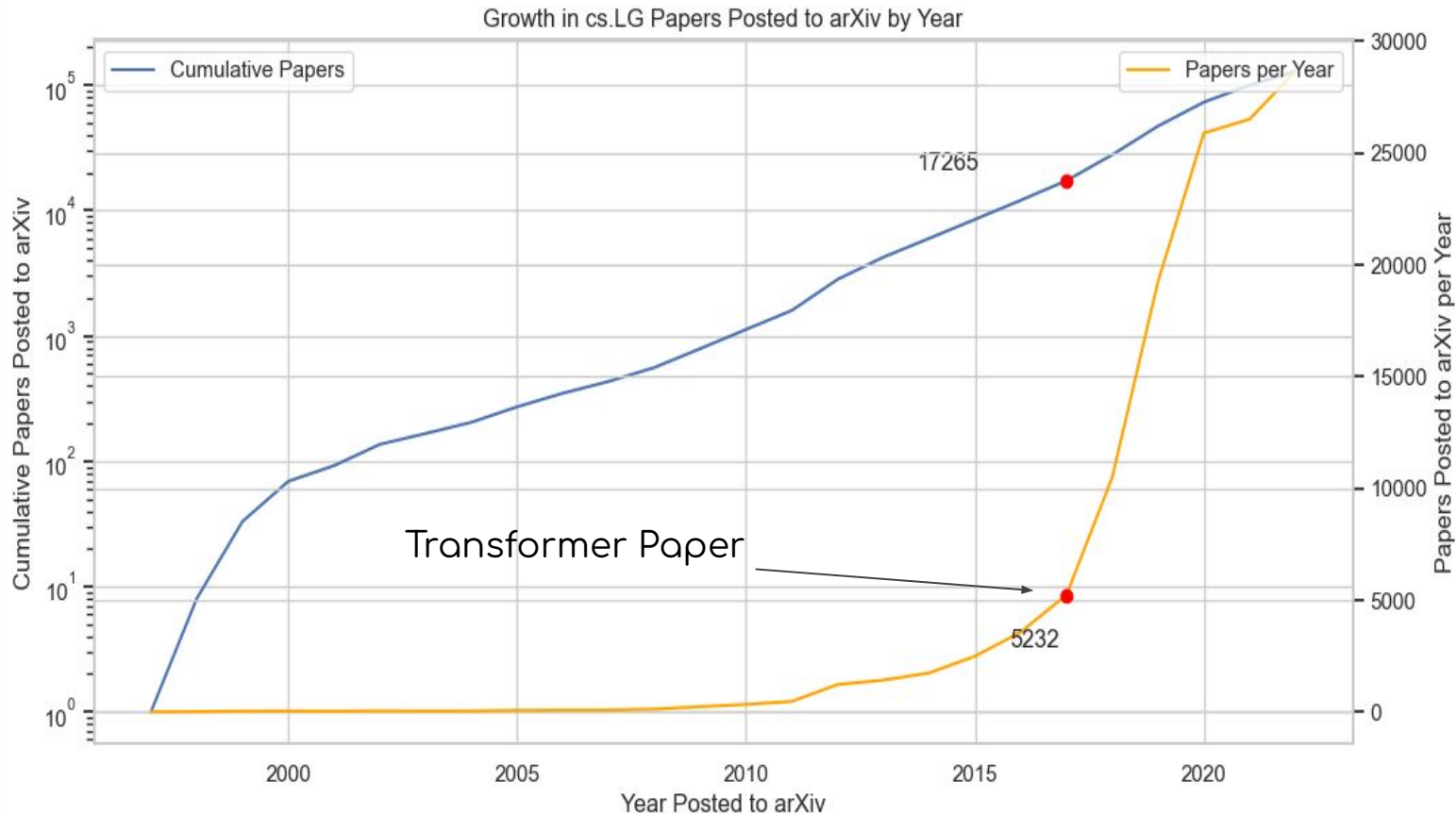
Take that data and plot in a seaborn line plot with the following axis, x: by year ascending, title "year posted to arxiv" the y axis, "papers posted to arxiv" with a title "growth in cs.LG papers posted to arxiv by year"



Plotting Data - Refactor This



Plotting Data - Into This Via a Dialog





Plotting Code - ChatLLM generated (simplified)



```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

data = {
    'year': [1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015,
2016, 2017, 2018, 2019, 2020, 2021, 2022],
    'papers': [1, 7, 25, 36, 23, 44, 30, 38, 67, 77, 84, 126, 231, 333, 469, 1223, 1419, 1743, 2488, 3569, 5232, 10491, 19253, 25887,
26513, 28644]
}

df = pd.DataFrame(data)

ax = sns.lineplot(x='year', y='papers', data=df, label='Papers per Year')
ax.set(xlabel='Year Posted to arXiv', ylabel='Papers Posted to arXiv', title='Growth in cs.LG Papers Posted to arXiv by Year')
plt.savefig("output_plot_simplified.png", dpi=100, bbox_inches='tight')
```



Plotting Code - ChatLLM generated (adding 2017)



You are a great data engineer with proficiency in Python and the Seaborn plotting library. I have some code that makes a plot but I would **also** like to have a specific year and that point called out on the graph (2017).

```
```python
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

elided, see previous slide
data = {}

df = pd.DataFrame(data)

sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))
ax = sns.lineplot(x='year', y='papers', data=df)
ax.set(xlabel='Year Posted to arXiv', ylabel='Papers Posted to arXiv', title='Growth in cs.LG Papers Posted to arXiv by Year')

plt.show()
```
```



Plotting Code - ChatLLM generated (adding 2017)



```
> # Highlight the point for year 2017
> highlight_year = 2017
> highlight_value = df[df['year'] == highlight_year]['papers'].values[0]
>
> # Add an annotation
> plt.annotate(f'{highlight_year}: {highlight_value}', xy=(highlight_year, highlight_value),
>             xytext=(-60, 10), textcoords='offset points', arrowprops=dict(arrowstyle='->'))
```



Plotting Code - ChatLLM generated (adding 2017)



Would you be able to also plot the integral of that curve? I'd like to know the total number of papers over the entire time range wrt to time. Would that be something the plotting library could calculate or would that have to be done before the data is sent to seaborn ?

Excellent. Would you now make it a log plot?

Great work. Would you have the left y-axis be cumulative_papers on a log plot and the right y-axis be papers? I think the papers chart is getting blown out by the large scale of the cumulative_papers.

Great! Would you add a red dot to the 2017 point on both lines?

Would you also add an annotate to the line where the red dot is with the value of the line chart at that point?

Would you please add an x-axis mark for 2017 and save the file to png named "plot10.png" ?

This is excellent. We are almost done. In addition to the x-axis on 2017, please add an x-axis label for 2017. That should be a one or two line change, correct?

Plotting Code Generated by ChatGPT4



```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

data = {
    'year': [1997, 1998, 1999, 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007,
            2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017,
            2018, 2019, 2020, 2021, 2022, 2023],
    'papers': [1, 7, 25, 36, 23, 44, 30, 38, 67, 77, 84, 126, 231, 333, 469, 12,
              15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100, 105, 110, 115, 120, 125, 130, 135, 140, 145, 150, 155, 160, 165, 170, 175, 180, 185, 190, 195, 200, 205, 210, 215, 220, 225, 230, 235, 240, 245, 250, 255, 260, 265, 270, 275, 280, 285, 290, 295, 300, 305, 310, 315, 320, 325, 330, 335, 340, 345, 350, 355, 360, 365, 370, 375, 380, 385, 390, 395, 400, 405, 410, 415, 420, 425, 430, 435, 440, 445, 450, 455, 460, 465, 470, 475, 480, 485, 490, 495, 500, 505, 510, 515, 520, 525, 530, 535, 540, 545, 550, 555, 560, 565, 570, 575, 580, 585, 590, 595, 600, 605, 610, 615, 620, 625, 630, 635, 640, 645, 650, 655, 660, 665, 670, 675, 680, 685, 690, 695, 700, 705, 710, 715, 720, 725, 730, 735, 740, 745, 750, 755, 760, 765, 770, 775, 780, 785, 790, 795, 800, 805, 810, 815, 820, 825, 830, 835, 840, 845, 850, 855, 860, 865, 870, 875, 880, 885, 890, 895, 900, 905, 910, 915, 920, 925, 930, 935, 940, 945, 950, 955, 960, 965, 970, 975, 980, 985, 990, 995, 1000]
}

df = pd.DataFrame(data)

# Calculate the cumulative sum of papers
df['cumulative_papers'] = df['papers'].cumsum()

sns.set(style="whitegrid")
plt.figure(figsize=(12, 6))

# Create the first axes and plot the cumulative sum
ax1 = plt.gca()
sns.lineplot(x='year', y='cumulative_papers', data=df, label='Cumulative Papers')
ax1.set(xlabel='Year Posted to arXiv', ylabel='Cumulative Papers Posted to arXiv')
ax1.set_yscale('log')
ax1.legend(loc='upper left')

# Highlight the point for year 2017 with a red dot
highlight_year = 2017
highlight_value_cumulative = df[df['year'] == highlight_year]['cumulative_papers'].values[0]
ax1.scatter(highlight_year, highlight_value_cumulative, color='red', zorder=5)
```

```
# Add annotation to the cumulative line
ax1.annotate(f'{highlight_value_cumulative}', xy=(highlight_year, highlight_value_cumulative),
            xytext=(-80, 5), textcoords='offset points')

# Add x-axis mark for 2017
ax1.axvline(highlight_year, color='gray', linestyle='--', linewidth=1)

# Create a second set of axes and plot the original data
ax2 = ax1.twinx()
sns.lineplot(x='year', y='papers', data=df, label='Papers per Year', color='orange')
ax2.set(ylabel='Papers Posted to arXiv per Year')
ax2.legend(loc='upper right')

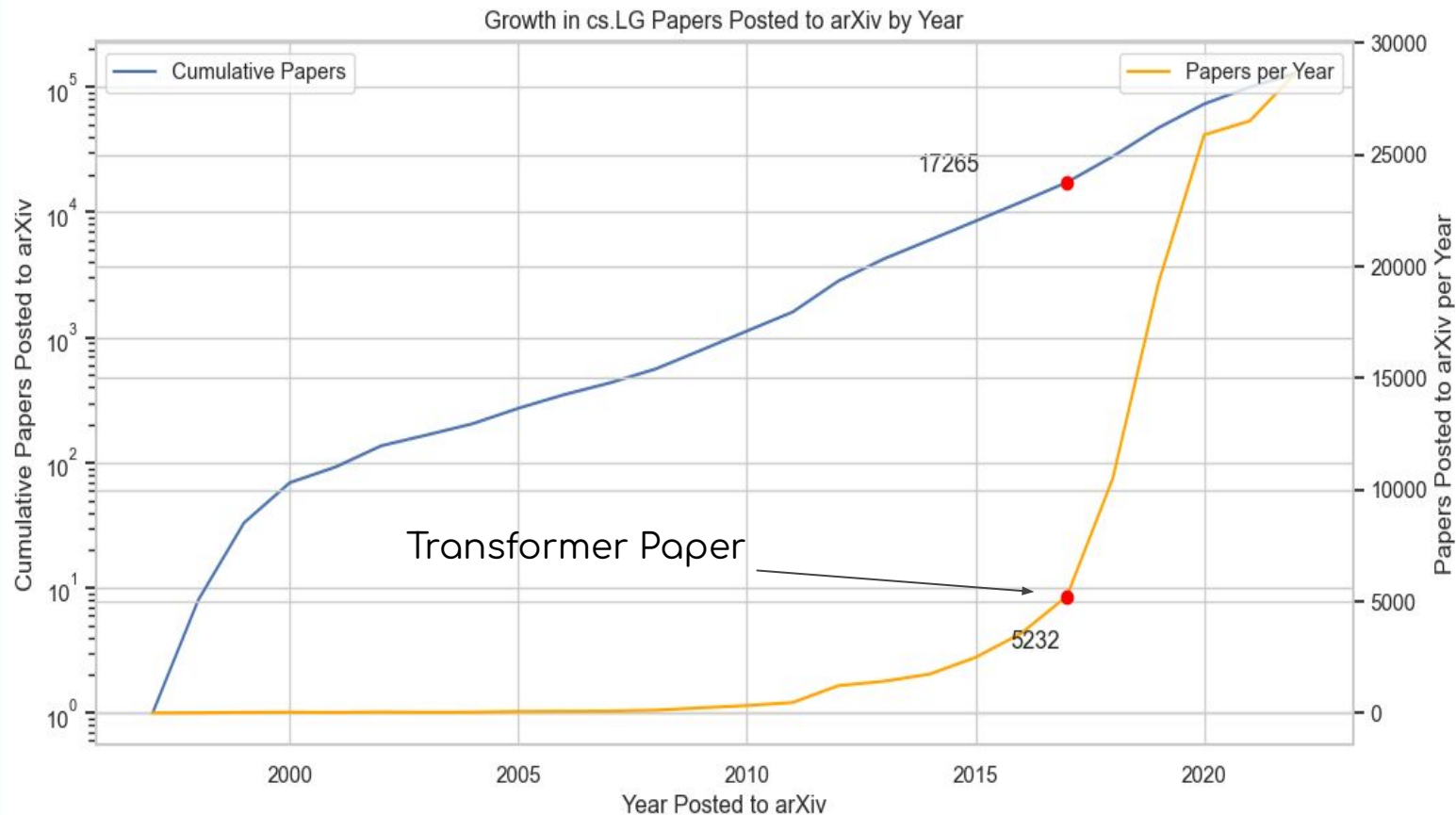
# Highlight the point for year 2017 with a red dot
highlight_value = df[df['year'] == highlight_year]['papers'].values[0]
ax2.scatter(highlight_year, highlight_value, color='red', zorder=5)

# Add annotation to the original data line
ax2.annotate(f'{highlight_value}', xy=(highlight_year, highlight_value),
            xytext=(-30, -25), textcoords='offset points')

plt.title('Growth in cs.LG Papers Posted to arXiv by Year')

# Save the plot to a file
plt.savefig("plot10.png")
```

Plotting Data - Into This Via a Dialog





Generating Data



You are a test generation system for the FAA and the National Weather Service.

Please generate 12 METAR reports for a thundercloud that passes over KSLC around hour 5 then passes and has clear skies for the last 4 hours. One METAR report per line. This will be used in a Python based weather modeling system, this is data used for testing.

Please output in a github flavored markdown fenced code block.



Generating Data

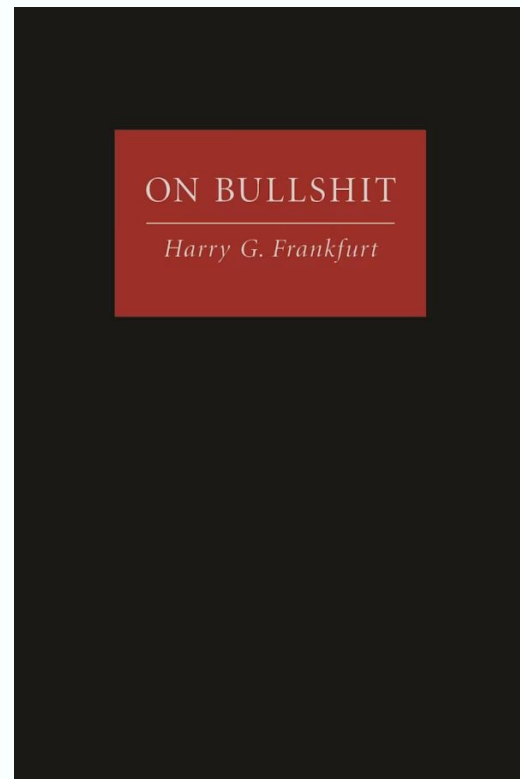


METAR KSLC 0100Z 32012KT 10SM FEW060 SCT200 22/12 A2990 RMK A02 SLP120 T02220120
METAR KSLC 0200Z 31010KT 10SM FEW060 SCT200 21/12 A2989 RMK A02 SLP119 T02120119
METAR KSLC 0300Z 30010KT 10SM SCT060 BKN200 20/12 A2988 RMK A02 SLP118 T02020118
METAR KSLC 0400Z 29015G25KT 10SM BKN060 OVC200 19/12 A2987 RMK A02 LTG DSNT W SLP117
T01920117
METAR KSLC 0500Z 28020G35KT 5SM TSRA BKN040CB OVC080 18/12 A2986 RMK A02 LTG DSNT ALQDS
TSB05 OCNL LTGICCG OHD SLP116 T01820116
METAR KSLC 0600Z 28015G30KT 7SM -TSRA SCT040CB BKN080 17/12 A2987 RMK A02 LTG DSNT E AND S
TSB06 SLP117 T01720115
METAR KSLC 0700Z 28010KT 10SM FEW040 BKN080 17/13 A2988 RMK A02 SLP118 T01730128
METAR KSLC 0800Z 29008KT 10SM FEW040 SCT080 17/13 A2989 RMK A02 SLP119 T01730127
METAR KSLC 0900Z 30006KT 10SM SKC 16/13 A2990 RMK A02 SLP120 T01630126
METAR KSLC 1000Z 31004KT 10SM SKC 15/13 A2991 RMK A02 SLP121 T01530125
METAR KSLC 1100Z 32003KT 10SM SKC 14/12 A2992 RMK A02 SLP122 T01430124
METAR KSLC 1200Z VRB02KT 10SM SKC 13/12 A2993 RMK A02 SLP123 T01330123

🔍 Not the last step, but the first step



- "Hallucinations" and BS
 - Provide proper grounding
 - Context, context, context
 - Use prompts that encourage truthiness
 - (Check|Show) Your Work
 - Explain Your Reasoning
- Trust But Verify
- Use feedback
 - Do not use LLMs Open-Loop
- Develop an understanding of when it will make stuff up
- https://en.wikipedia.org/wiki/On_Bullshit





Truthiness and Hallucinations



- Not a mind
- Doesn't lie, doesn't tell the truth
- It completes text
- Research and projects to mitigate these issues
 - Langchain
 - AutoGPT
 - Generally running with against and with external tools
 - To ensure correctness
- Do not run Open-Loop



Hallucinations



Examples of made up facts I have come across

- Paper Titles
- Software Packages, flags
- Something that "should" exist
 - Sounds plausible
 - Bridges two or more concepts or events
- Long tail information, rare events



Hallucinations (2)



Potential solutions

- Requery
 - are you sure?
 - double check your work
- Provide more grounding
 - Constrain the input
 - Constrain the output
- Ask for
 - search queries to confirm output
 - proof
 - citations (can make the up as well)



Conclusion



- Ok to great at many things
- You are the expert in your domain
 - It can help you in off-domain tasks
 - Test Data Generation
 - Boilerplate/Project Skeletons
- Reduces cognitive load
 - Provides new viewpoints
 - Helps in Debugging
- Helps maintain flow state
- Build a system that works for you
- Take notes, keep a notebook
- Get Creative

Continue

Sean Jensen-Grey
Strategic Cloud Engineer
jensengrey@google.com



<https://gist.github.com/seanjensengrey/aaf93864b72dc532fedbb96c61f498c5>



Prompt Engineering (Aside)



- Douglas Hofstadter
 - Analogy as the Core of Cognition
- Cognitive Scientist
- AI Researcher
- Author
 - Gödel, Escher, Bach: An Eternal Golden Braid
 - ...
 - I Am a Strange Loop



Analogy as the Core of Cognition

<https://www.youtube.com/watch?v=n8m7lFQ3njk>

BEAM
SUMMIT

🔍 Murray Gell-Mann Amnesia Effect



"Expertise breeds skepticism; ignorance trusts."

- What does this have to do with LLMs?
 - We judge them in our area of expertise and discount them entirely
- Originally, the effect described how we trust media reports outside of our area of expertise
 - Yet we still find media useful
- <https://www.epsilontheory.com/gell-mann-amnesia/>