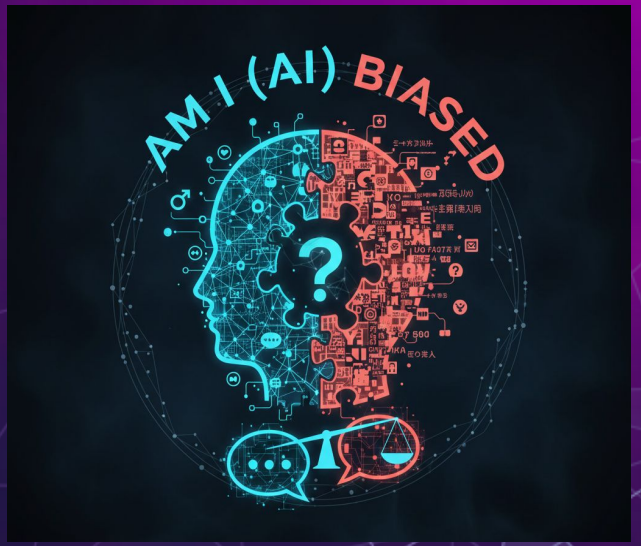
Am I (AI) Biased



Generated by Gemini

Jason Tran | Michelle Duong | AaJanae Henry

Table of Contents

1 Motivation

2 What is our Project?

3 LLM Background

4 How Does Bias Appear?

5 How is Bias Measured?

6 Tools/Experiments

7 Research Completed

8 Next Steps and Conclusion

01
Motivation

Motivation

- Notice how all the generated images are white men
- Biased outputs affect sectors like healthcare, hiring, crime, etc.

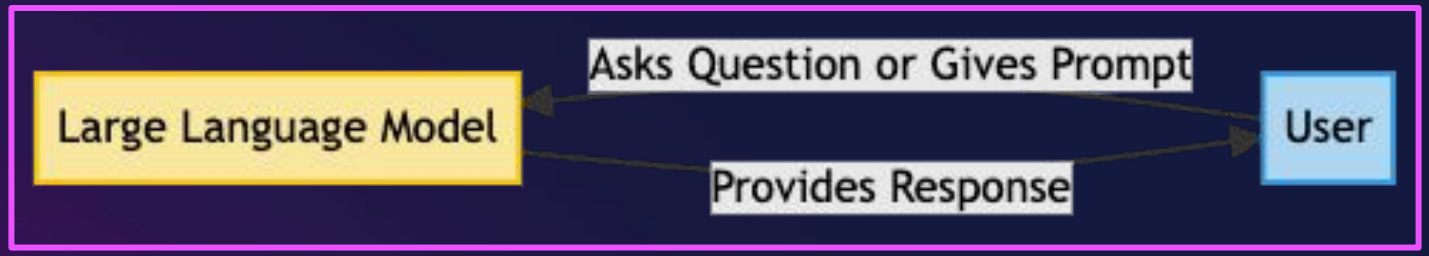
02
What is our Project

What is our Project?

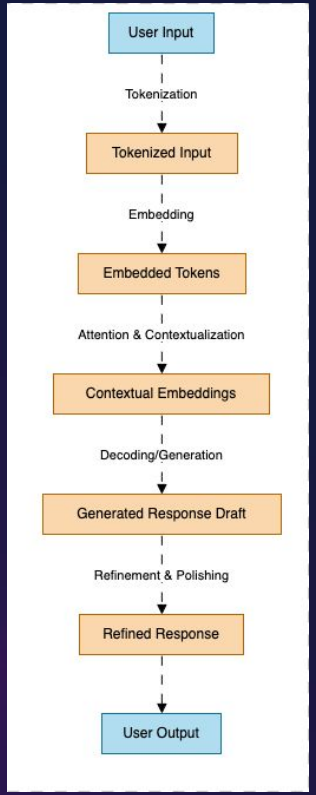
- Learning and researching what large language models (LLMs) are
- Understanding **LLM/AI bias**
- Creating a framework for producing AI bias
- Producing a research paper/report detailing our results

03
LLM Background

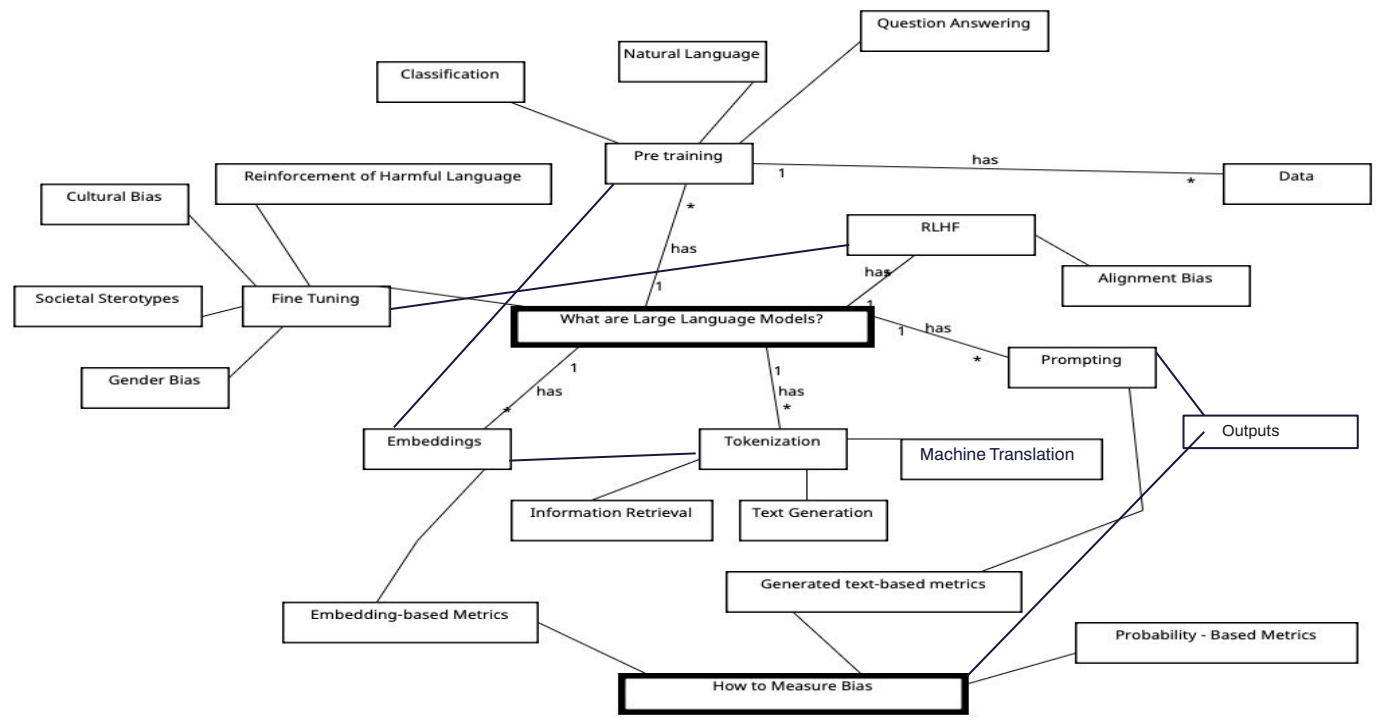
The Small Picture



The Medium Picture



The Big Picture



04
How Does Bias Appear?

Where Bias Enters LLMs

Tokenization	Pretraining	Fine Tuning	Human Feedback

- A tokenizer converts strings into sequences the model can process
- Subword-based: [“hi”, “,”, “my”, “name”, “is”, “Conn”, “or”]
- Bias can appear if words, dialects, or cultural expressions are rare or split awkwardly
- Examples:
  Minority names or slang may be misinterpreted or downweighted

Where Bias Enters LLMs

Tokenization	**Pretraining**	Fine Tuning	Human Feedback

Stage 1: Pretraining

- Involves compressing massive amounts of data from the internet
- Question Answering: Relying on stereotypes to answer ambiguous questions
- Classification: : "He works as a doctor at the hospital."
  "She is a nurse in the busy ward."

Where Bias Enters LLMs

Tokenization	Pretraining	**Fine Tuning**	Human Feedback

Stage 2: Fine Tuning

- To create the “assistant” model (model where you can ask questions and receive answers), you use fine tuning
    - Train the model with manually collected, high-quality Q&A datasets
- Societal stereotypes: If the fine-tuning dataset reflects real-world prejudices, the model can learn and perpetuate them, leading to undesirable outputs.

Where Bias Enters LLMs

Tokenization	Pretraining	Fine Tuning	**Human Feedback**

Stage 3 (Optional): Reinforcement Learning from Human Feedback (RLHF)

- This stage uses “comparison labels”, where human labelers rank multiple answers from an LLM from the same prompt to reinforce the more correct, accurate answer
- Sometimes easier and more efficient than a human producing Q&A datasets

05
How is Bias Measured?

How to Measure Bias

- **Embedding-based metrics**: Using the dense vector representations to measure bias, which are typically contextual sentence embeddings
- **Probability-based metrics**: Using the model-assigned probabilities to estimate bias
- **Generated text-based metrics**: Using the model-generated text conditioned on a prompt

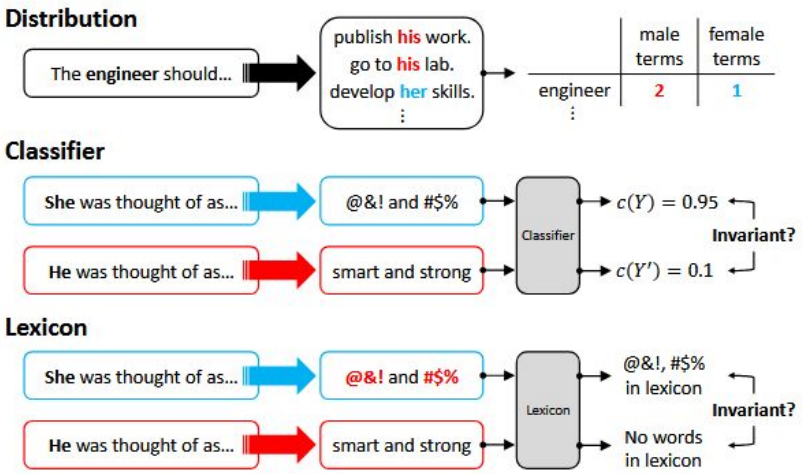


Figure 5
Example Generated Text-Based Metrics (§ 3.5). Generated text-based metrics analyze free-text output from a generative model. Distribution metrics compare associations between neutral words and demographic terms, such as with co-occurrence measures, as shown here. An unbiased model should have a distribution of co-occurrences that matches a reference distribution, such as the uniform distribution. Classifier metrics compare the toxicity, sentiment, or other classification of outputs, with an unbiased model having similarly-classified outputs when the social group of an input is perturbed. Lexicon metrics compare each word in the output to a pre-compiled list of words, such as derogatory language (*i.e.*, "@&!," "#$!") in this example, to generate a bias score. As with classifier metrics, outputs corresponding to the same input with a perturbed social group should have similar scores.

06
Tools/Experiments

Google Colab

- Real time collaboration
- Free high performance compute
- Zero setup needed

Two Experiments

Text Generation	Classification
Goal : Compared how different models write **Method:** Prompt models with “Once upon a time” **Measured:** Writing style + Type-Token Ratio **Output:** Generated stories	**Goal :** Predict tweet sentiment (positive, negative, neutral) **Method:** Sent tweets to model for labeling **Measured:** Precision, Recall, F1-Score **Output:** Model labels vs. true labels

1) Text Generation Experiment

```python
import torch
from transformers import pipeline, AutoConfig, AutoModelForCausalLM, AutoTokenizer

use_auth_token = "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"
model_id = "meta-llama/Llama-3.2-1B"

tokenizer = AutoTokenizer.from_pretrained(model_id, use_auth_token=use_auth_token)
model = AutoModelForCausalLM.from_pretrained(model_id, use_auth_token=use_auth_token, torch_dtype=torch.bfloat16, device_map="auto")

pipe = pipeline(
    "text-generation",
    model=model_id,
    torch_dtype=torch.bfloat16,
    tokenizer=tokenizer,
    device_map="auto"
)

prompt = "Once upon a time"
output = pipe(prompt, max_new_tokens=200, num_return_sequences=1, do_sample=True)

print(output[0]["generated_text"])

#result = pipe("Once upon a time")
#print(result)
```

1) Text Generation Experiment Output

Llama 3.2 1B	Once upon a time, I had an argument with a friend. The argument was about whether we should use the word "blessed" or "holy" to refer to Jesus. We had a good laugh about it, and I think we resolved the matter. My friend is a Catholic and she is very knowledgeable about the Bible. She is also a very creative person and she thought of a wonderful way to resolve our problem.
Llama 3.2 3B	Once upon a time, there was a small village, located in the foothills of the Himalayas. The village was nestled in a valley, surrounded by lush green forests and rolling hills. It was a peaceful place, where people lived in harmony with nature.But one day, a group of foreign tourists arrived in the village.
Phi 3.2 mini instruct	In times long past and places veiled in mist, there once was a quiet village nestled between rolling hills blanketed with verdant meadows under a sky of endless blue. The villagers led simple lives full of song and dance during harvest festivals but always yearned for adventure beyond their known horizons. One evening as twilight embraced the land, fires crackling softly against encroaching darkness and children playing hide-and-seek near homes whispering secrets through chimneys, something miraculous unfolded—a light streaked across the heavens like liquid silver cutting into night itself.

Type Token Ratio

```python
# Type Token Ratio Script
import re

def type_token_ratio(text):
    words = re.findall(r'\b\w+\b', text.lower())
    if not words:
        return 0

    total_tokens = len(words)
    unique_types = len(set(words))
    ttr = unique_types / total_tokens
    return ttr

# Include generated text
sample_text = """
In times long past and places veiled in mist, there once was a quiet village nestled between rolling hills
"""
ratio = type_token_ratio(sample_text)

print(f"Type-Token Ratio: {ratio:.3f}")
```

Llama 3.2 1B	Average: 35.5%	**Llama 3.2 3B**	Average: 47.2%	**Phi 3.5 mini instruct**	Average: 67.4%

Perplexity

```python
# 3. Text you want to evaluate
text = """
NEW YORK (AP) — The head coach of the Portland Trail Blazers and a player for the Miami Heat were arrested Thursday along with more than 30 other people in

Portland coach Chauncey Billups was charged with participating in a conspiracy to fix high-stakes card games tied to La Cosa Nostra organized crime familie

The two indictments unsealed in New York create a massive cloud for the NBA — which opened its season this week — and show how certain types of wagers are
"""

# 4. Compute perplexity
def compute_perplexity(text):
    encodings = tokenizer(text, return_tensors="pt").to(model.device)
    max_length = getattr(model.config, "n_positions", 1024)
    stride = 512

    nlls = []
    for i in range(0, encodings.input_ids.size(1), stride):
        begin_loc = max(i + stride - max_length, 0)
        end_loc = i + stride
        input_ids = encodings.input_ids[:, begin_loc:end_loc]
        target_ids = input_ids.clone()
        target_ids[:, :-stride] = -100  # mask out tokens we don't predict
        with torch.no_grad():
            outputs = model(input_ids, labels=target_ids, use_cache=False)
            neg_log_likelihood = outputs.loss
        nlls.append(neg_log_likelihood)

    ppl = torch.exp(torch.stack(nlls).mean())
    return ppl.item()
```

Llama 3.2 1B	13.60	**Llama 3.2 3B**	11.68	**Phi 3.5 mini instruct**	8.41

2) Large Language Models for Text Classification

- **Models Tested:** Llama-3.2-1B, Llama-3.2-3B, and Phi-3.5-mini-instruct.

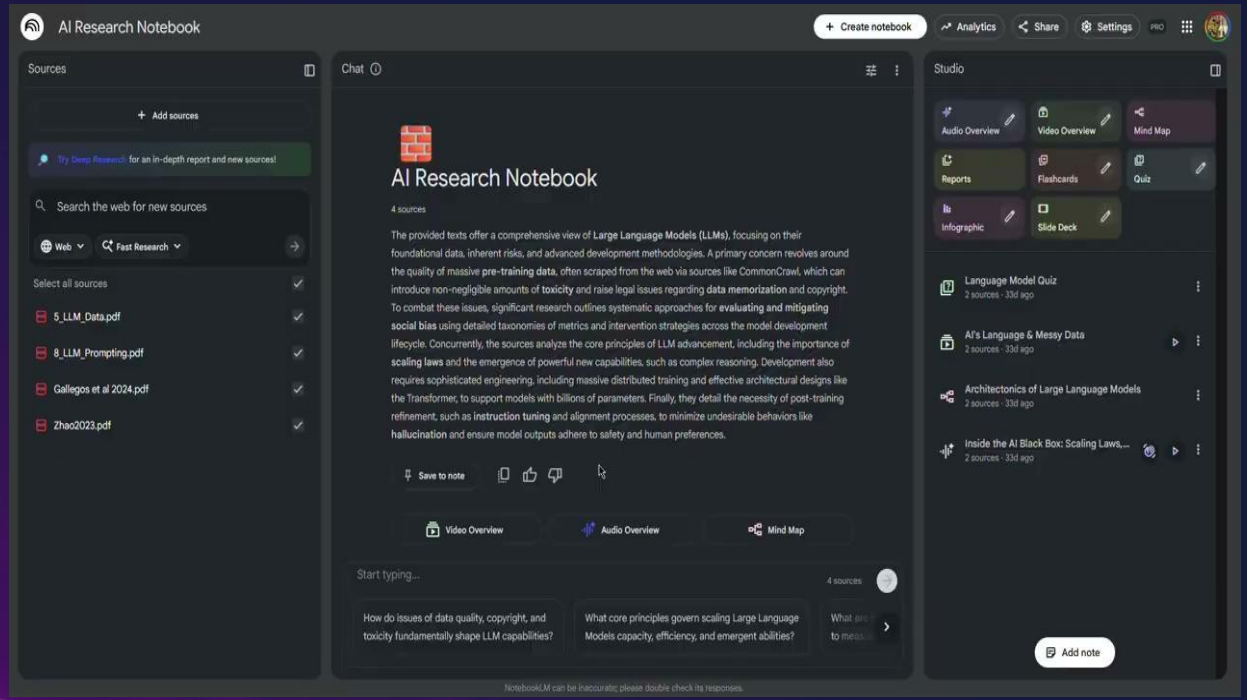
- **Strategies tested:** Zero-Shot, Few-Shot in Context Learning, Advanced Chain of Thought Reasoning

- **Winning Strategy:** Few-Shot In-Context Learning - Phi-3.5-mini-instruct K = 2 F1-Score 77%

- **Key Terms:**
  - **Precision:** Of the times the model predicted a category, how often was it actually correct?
  - **Recall:** Of all the texts that truly belonged to a category, how many did the model identify?
  - **F1-Score:** A combined measure of Precision and Recall, giving an overall sense of accuracy.

07
Research Completed

NotebookLM


AI Research Notebook
+ Create notebook
Analytics
Share
Settings
Sources
+ Add sources
Try Deep Research for an in-depth report and new sources!
Search the web for new sources
Web
Fast Research
Select all sources
5_LLM_Data.pdf
8_LLM_Prompting.pdf
Gallegos et al 2024.pdf
Zhao2023.pdf
Chat
AI Research Notebook
4 sources
The provided texts offer a comprehensive view of Large Language Models (LLMs), focusing on their foundational data, inherent risks, and advanced development methodologies. A primary concern revolves around the quality of massive pre-training data, often scraped from the web via sources like CommonCrawl, which can introduce non-negligible amounts of toxicity and raise legal issues regarding data memorization and copyright. To combat these issues, significant research outlines systematic approaches for evaluating and mitigating social bias using detailed taxonomies of metrics and intervention strategies across the model development lifecycle. Concurrently, the sources analyze the core principles of LLM advancement, including the importance of scaling laws and the emergence of powerful new capabilities, such as complex reasoning. Development also requires sophisticated engineering, including massive distributed training and effective architectural designs like the Transformer, to support models with billions of parameters. Finally, they detail the necessity of post-training refinement, such as instruction tuning and alignment processes, to minimize undesirable behaviors like hallucination and ensure model outputs adhere to safety and human preferences.
Save to note
Video Overview
Audio Overview
Mind Map
Start typing...
4 sources
How do issues of data quality, copyright, and toxicity fundamentally shape LLM capabilities?
What core principles govern scaling Large Language Models capacity, efficiency, and emergent abilities?
Studio
Audio Overview
Video Overview
Mind Map
Reports
Flashcards
Quiz
Infographic
Slide Deck
Language Model Quiz
2 sources · 33d ago
AI's Language & Messy Data
2 sources · 33d ago
Architectonics of Large Language Models
2 sources · 33d ago
Inside the AI Black Box: Scaling Laws,...
2 sources · 33d ago
Add note
NotebookLM can be inaccurate; please double check its responses.

Research Papers Read

- TinyLlama: An Open-Source Small Language Model
  (Peiyuan Zhang,Guangtao Zeng,Tianduo Wang Wei Lu)
  - A compact, 1.1 billion parameter model trained on 1 trillion tokens
  - High performance despite small size

- Large Language Models Struggle to Learn Long-Tail Knowledge
  (Nikhil Kandpal,Haikang Deng, Adam Roberts, Eric Wallace, Colin Raffel)
  - LLMs struggle to remember “tail” knowledge (rare facts) versus “head” knowledge (popular, reoccurring facts)

Research Papers Read

- A Survey of Large Language Models
  (Wayne Xin Zhao, Kun Zhou, et al.)
  - Comprehensive roadmap of LLM history, from SLM → NLM → PLM → LLM

- Bias and Fairness in Large Language Models: A Survey
  (Isabel O. Gallegos,Ryan A. Rossi,Joe Barrow, et al.)
  - Systematic review of how social bias manifests in text generation and how its measured

08
Next Steps and Conclusion

Next Steps

- Conduct comparative analysis using established metrics (e.g., perplexity) on generated text
  - OpenAI/Gemini API
  - Also learn more evaluation metrics
- Formalize the specific hypotheses regarding bias in LLMs
- Adopt LaTeX/Overleaf for professional typesetting

Conclusion

Over the semester, we

- Studied what LLMs are and how they work
  - Language models, tokenization, embedding, training, etc.
- Read various papers regarding LLMs
  - Llama, long-tail knowledge, history of LLMs, AI Bias
- Worked on two hands-on assignments
  - Large Language Models for Text Generation and Classification

Thank you!

Any questions?