LLM Ecosystems:
From Foundations to Frontiers

ACSPRI Summer Program 2026 Dr. Maria Prokofieva

Maria Aise

Why Do LLMs Matter in Social Science?

LLMs scale human-level language understanding

They can analyze millions of words — reports, interviews, articles — with nuanced pattern recognition that mimics expert judgment at scale.

They reduce cost + time for text-heavy workflows

Manual coding, thematic tagging, or policy trend analysis that once took weeks can now be done in hours using reproducible, scriptable methods.

Applications include policy, legal analysis, qualitative coding

LLMs are already used to monitor citizen sentiment, identify emerging legal themes, cluster qualitative survey data, and flag ethical risks in reports.

Academic access is now feasible — no longer just for Big Tech

Open-weight models and APIs (like Hugging Face, OpenAI, Gemini) now allow researchers to apply cutting-edge models with minimal infrastructure.

What is an LLM?

Machine learning models trained on billions of words

Understand and generate human-like text

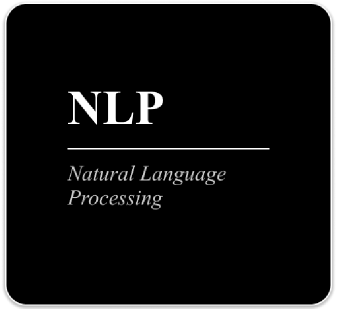
Can summarize, classify, extract, and generate

Work via token prediction, not memorization

Used via APIs or code

Not perfect, but powerful and improving

LLMs Power Text Analysis

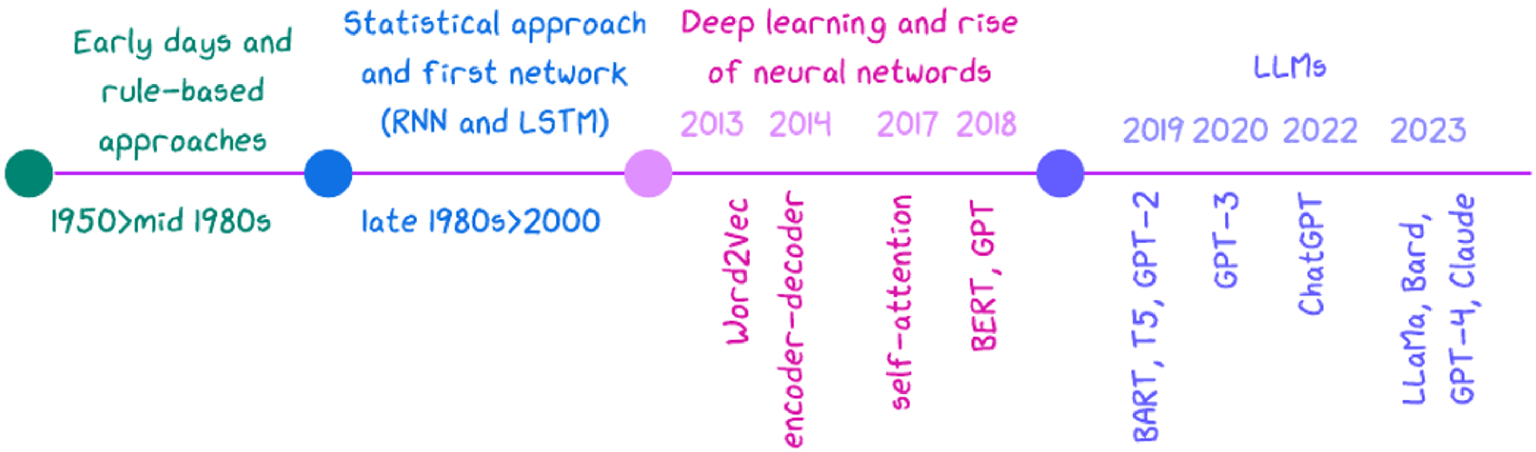


Auto-code interviews or survey responses

Summarize long documents for policymaking

Detect sentiment shifts across large datasets

Classify thousands of news articles in minutes



The LLM Timeline

How Do LLMs Actually Work?

📚 Trained on billions of words

Ingests books, articles, websites — huge corpus

📚 Predicts tokens, not retrieves facts

Generates next word based on probability

🧠 Uses transformer architecture

“Attention” lets model focus on relevant input

🧾 Understands context in chunks (context window)

But not global memory — limited to recent input

Principles

Self-Attention

Key idea: understand word relevance across a sequence

Positional Encoding

Inject structure into unordered tokens

Transformer Block

Composed of dense layers + normalization + residual paths, and stacked deep

Major Types of LLM Models

Encoder-only

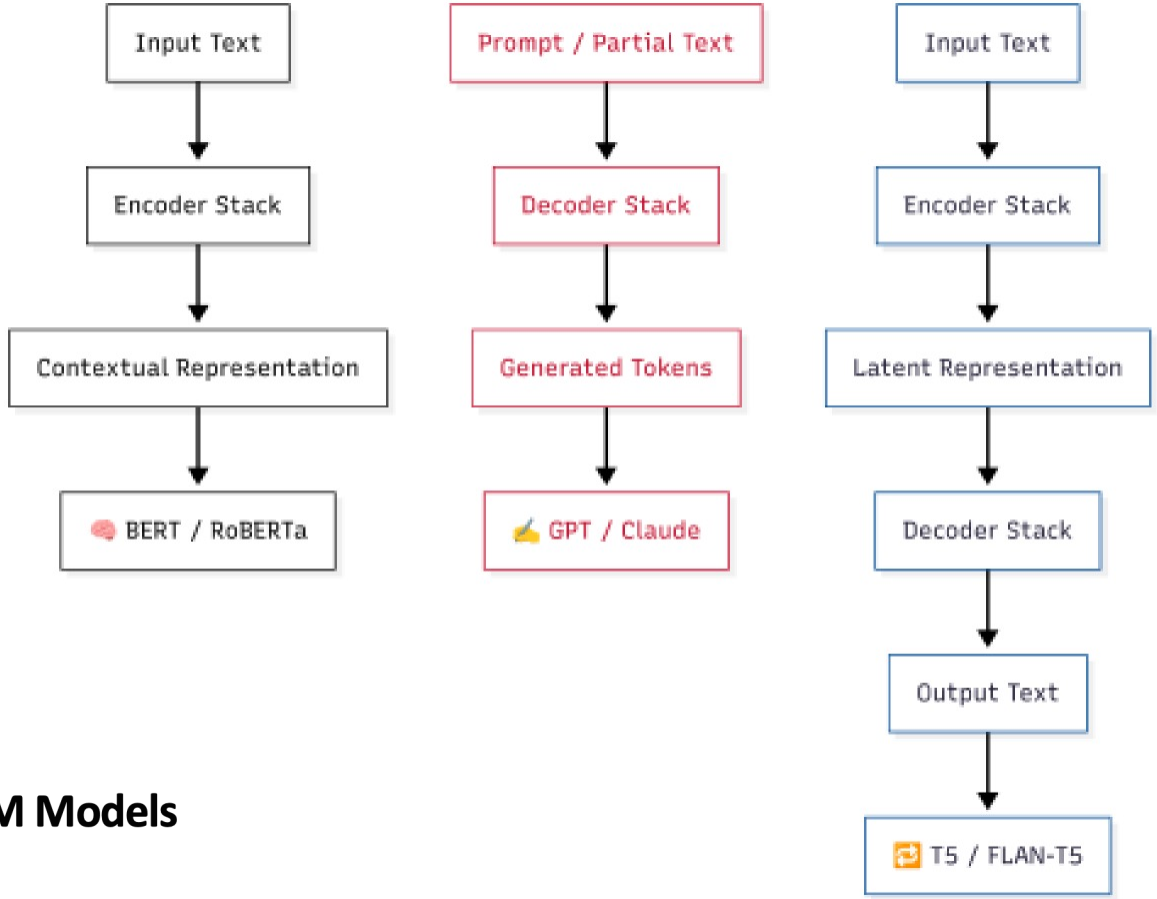
Goal: Understand input text deeply
What it does: Reads the full input and builds a contextual representation
Popular examples: BERT, RoBERTa, DistilBERT
Used for:
Text classification
Named Entity Recognition
Sentence similarity *No text generation*

Decoder-only

Goal: Generate fluent text from left to right
What it does: Predicts one token at a time based on previous tokens
Popular examples: GPT-2, GPT-3, GPT-4, Claude, Gemini
Used for:
Chatbots
Text generation
Summarization *These models don't "see" the entire input at once — they build it as they go.*

Encoder–Decoder (Seq2Seq)

Goal: Transform one sequence into another
What it does: Encoder reads the full input
Decoder generates output
Popular examples: T5, FLAN-T5, BART, mT5
Used for:
Translation
Text summarization
Question answering *They're the most flexible, but more complex.*



Major Types of LLM Models

Modalities in LLMs

Modality

What kinds of input can LLMs understand?

Text
Natural Language input/output

Image
Visual input: images/photos

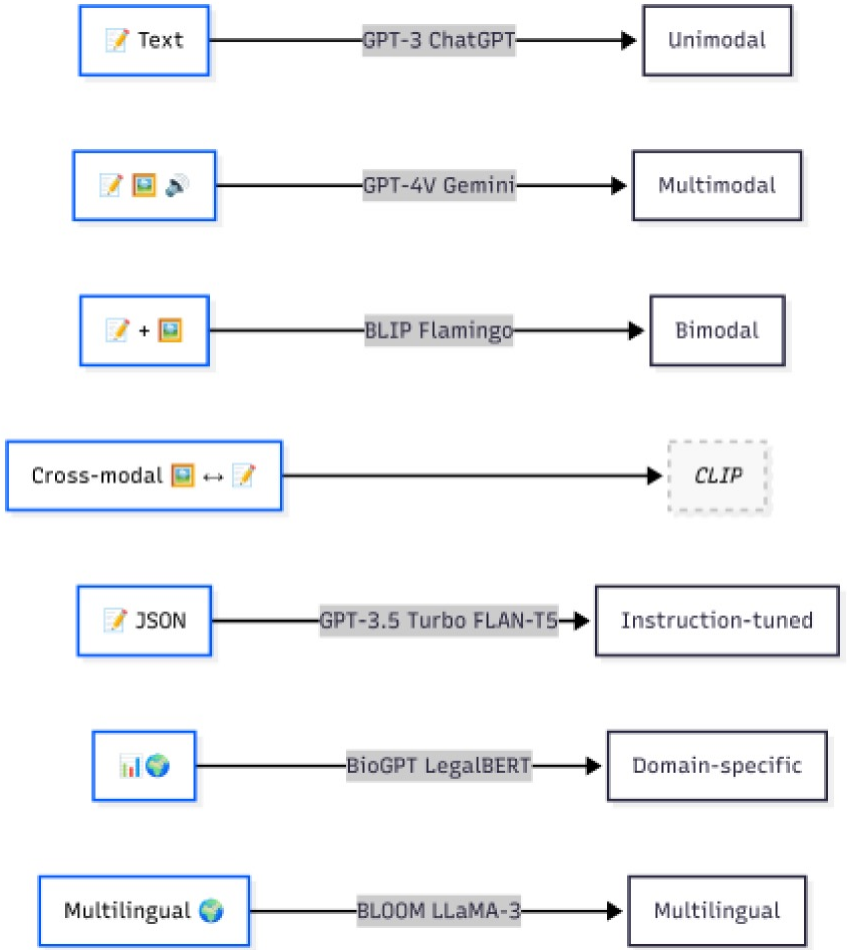
Audio
Voice/sound input/output

Video
Moving sequences

Multilingual
Cross-lingual understanding

Structured Data
Tables, JSON, code

How Modalities Map to Model Types



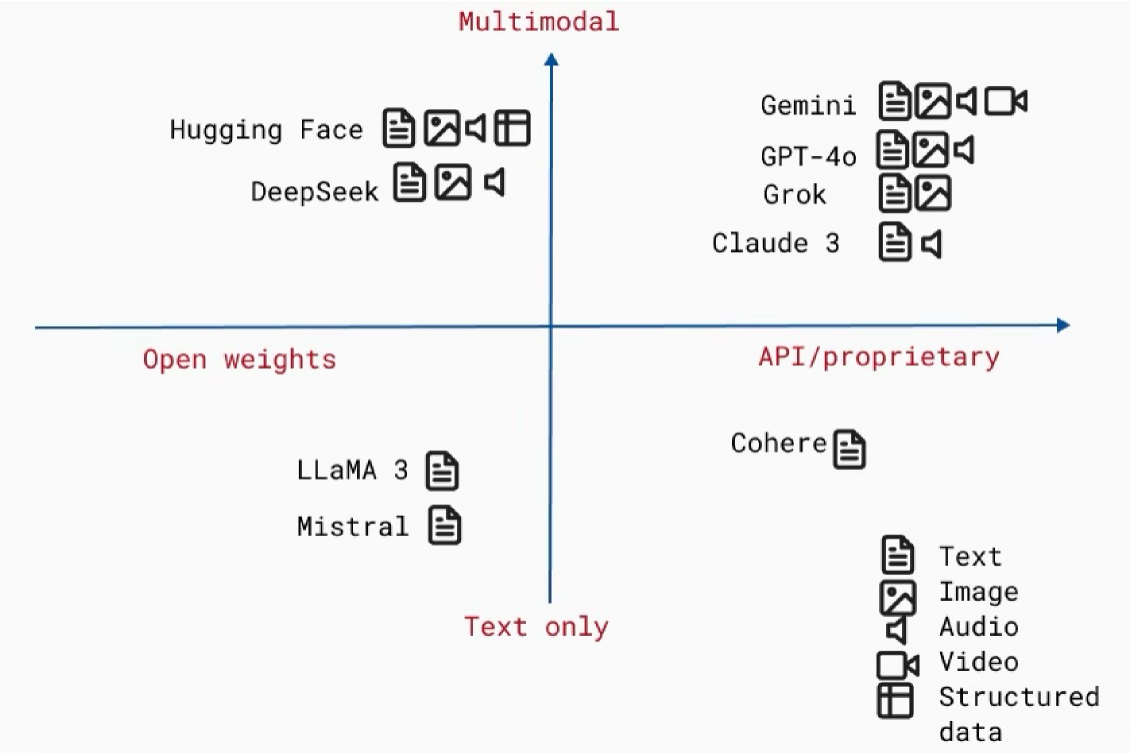
Foundational vs Proprietary Models

Foundational Models

- Trained on broad, general-purpose datasets (e.g. Common Crawl, Wikipedia, books)
- Released as base models.
- **Examples**: LLaMA 3, Mistral, Falcon.

Proprietary Models

- Developed by private companies
- High performance, but API-only access
- Customization often limited.
- **Examples**: GPT-4, Gemini, Claude.



Model Families Landscape

Model= Architecture + Weights

What Makes a Model Work?
*Architecture gives **structure**.*
*Weights give **skill**.*

Architecture

Blueprint: How many layers, what type, how they connect
Defines **how**:

- data flows through the model
- data patterns can be learned

Architecture choice determines if a model can handle images (CNNs) or text (transformers).

Weights

Learned values: What the model knows after training
Numerical values (e.g., 0.347, -1.892) that scale input data.
Weights determine if a model actually works well for your specific task.

Proprietary Models: What You Don't Get to See

Understanding the hidden layers behind closed-source LLMs

- **Weights are not released** — only accessible via API.
- **Training data** is undisclosed .
- **Model architecture** may not disclosed.
- **Fine-tuning access** is limited or expensive.
- **Usage tracking + safety layers** often run in the background.

Parameters

What does a trillion-parameter model actually mean?

Numerical values

Parameters are the that define how a **neural network processes** input data to produce output.

Not "knowledge"

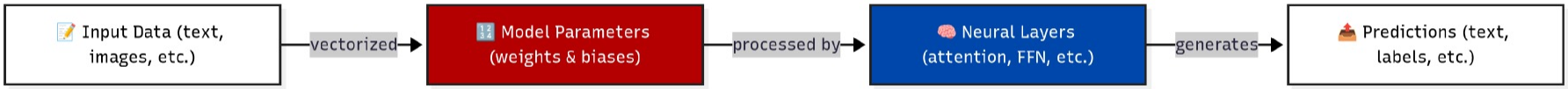
That's the weights and patterns formed by parameters during training.

Not "rules"

The model has no explicit logic—just math.

Not fixed

They change during training via optimization.



What Parameters Actually Do

Transform Input Data

Every input (text, image, etc.) is converted to numbers (vectors).
Parameters are the coefficients in the functions that:
Scale (**weight**),
Shift (**bias**), or
Gate (**attention**) these numbers.

Enable Learning

During training, parameters are adjusted via:
Loss calculation: How wrong the model's output is.
Gradient descent: Tweaking parameters to reduce loss.

Dictate Model Behavior

The arrangement and values of parameters determine:
Capacity: Can it memorize complex patterns?
Generalization: Can it apply patterns to new data?

📦 *Parameters = the knobs, levers, and memory of a model. They shape how input becomes output — and how models evolve.*

What Does “Open Weights” Actually Mean?

Open Weights ≠ Full Openness

Downloadable model weights

Training data (usually unreleased or partially described)

Ability to run/fine-tune locally

Full license to deploy commercially (varies by model)

Architecture details (often published)

Tokenizers, eval metrics, or guardrails often separate

Does Bigger = Better?

More parameters ≠ better results — unless trained well and aligned to the right task.

Memorization vs Generalization

🧠 **Bigger ≠ Smarter**

Large models can **memorize training data** if not trained properly — making them brittle or biased.

***Example**: A poorly trained 70B model might repeat toxic forum posts instead of understanding nuance in a new context.*

Finetuned > Gigantic

🎯 **Smaller + Targeted = Better**

A smaller model, **finetuned on clinical trials**, can outperform GPT-4 on biomedical summarization.

Example: BioGPT or a custom Gemini finetune often beats generic LLMs in specialized settings.

Cost Reality

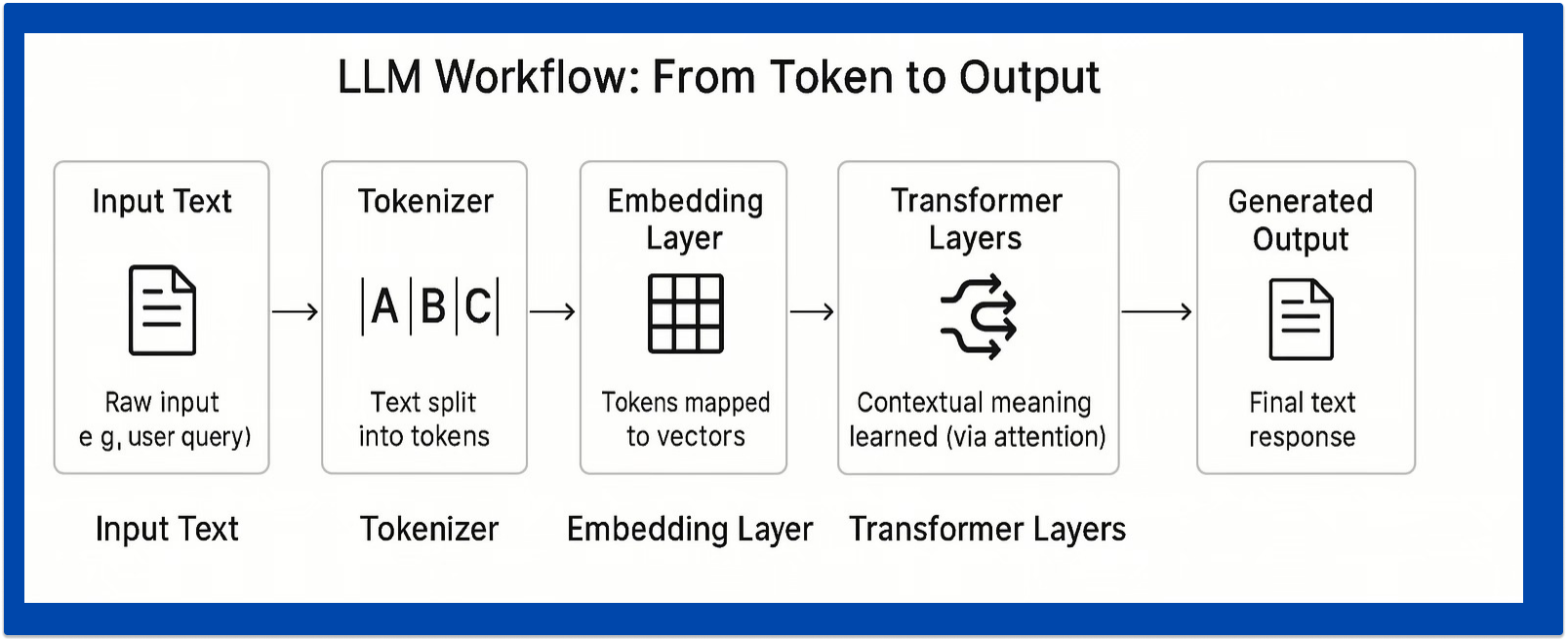
💰 **More Params = More Compute = More $$$**

Every billion parameters adds cost — for both training and inference.

Example: GPT-4-turbo is cheaper than GPT-4 because it optimizes how those parameters are used — not because it's smaller.

Maria Arise

Preview — From Parameters to Meaning (Day 2)



"The model doesn't just generate text. It learns to position language in space."

Maria Aise