Interpretability on How Large Language Models Process Information

From Tokens to Reasoning

Jasper Kyle Catapang, MA

Doctoral (PhD) Researcher, Tokyo University of Foreign Studies
Senior AI Engineer, Money Forward, Inc., Japan

DLSU CCS via Zoom

July 14, 2025

Overview

Level 1

General

“Before we explain *how* they work,
we must ask: *what* they are.”

Back to Index

What Is a Large Language Model?

- A model trained to predict the next word in a sequence.
- Learns from billions of words (web, books, conversations).
- Generates fluent language but doesn't "know" facts.
- Analogy: autocomplete on steroids.

`Input: The weather today is` ⇒ `sunny`

Where Are LLMs Used Today?

- Chatbots
- Email writing
- Code generation
- Tutoring
- Summarization
- Translation
- Poetry & fiction
- Research assistance
- Brainstorming
- Game narrative tools

LLMs aren't one tool — they're a toolbox.

ChatGPT: The Breakout Moment

- Released by OpenAI in late 2022, based on GPT-3.5.

ChatGPT: The Breakout Moment

- Released by OpenAI in late 2022, based on GPT-3.5.
- Reached 1 million users in 5 days — fastest-growing consumer app at the time.

ChatGPT: The Breakout Moment

- Released by OpenAI in late 2022, based on GPT-3.5.
- Reached 1 million users in 5 days — fastest-growing consumer app at the time.
- Used for:
  - Homework help, coding, resumes
  - Legal briefs, medical advice (sometimes incorrectly)
  - Conversation, companionship, therapy prompts

ChatGPT: The Breakout Moment

- Released by OpenAI in late 2022, based on GPT-3.5.
- Reached 1 million users in 5 days — fastest-growing consumer app at the time.
- Used for:
  - Homework help, coding, resumes
  - Legal briefs, medical advice (sometimes incorrectly)
  - Conversation, companionship, therapy prompts
- Sparked debates:
  - Is this cheating?
  - Should it be banned in classrooms?
  - Who's responsible for mistakes?

ChatGPT showed not just what LLMs can do — but how unprepared society was.

Prompt Design

- **Be clear and explicit.** Don't assume the model knows your intent.
- **Provide context.** Include relevant information (e.g., audience, tone).
- **Specify the output format.** List? Table? Email?
- **Use examples.** Show what you want with demonstrations.
- **Iterate.** Refine prompts to improve quality.

Less Effective Prompt:

- "Write me a thunderstorm poem."
- No format, no constraints, unclear tone.

More Effective Prompt:

- "Write a haiku (3 lines, 5-7-5) about a thunderstorm. Use vivid imagery."
- Clear task, form, and style guidance.

Better prompts → better results: LLMs follow instructions, not intentions.

What Is Hallucination?

- An LLM may generate confident but **false** statements.
- It's not lying — it just predicts what sounds plausible.
- The output isn't grounded in real knowledge.

```
"Hawaiian pizza was invented in Hawaii in 1962."
```

The correct origin of Hawaiian pizza is Ontario, Canada.
Fluency $\neq$ *truth.*

LLMs Are Not Search Engines

- Google **retrieves** — it looks up from a database.

LLMs Are Not Search Engines

- Google **retrieves** — it looks up from a database.
- LLMs **generate** — they synthesize based on probability.

LLMs Are Not Search Engines

- Google **retrieves** — it looks up from a database.
- LLMs **generate** — they synthesize based on probability.
- They don't verify facts or cross-check reality.

LLMs Are Not Search Engines

- Google **retrieves** — it looks up from a database.
- LLMs **generate** — they synthesize based on probability.
- They don't verify facts or cross-check reality.
- Tools like RAG *try* to fix this, but not always reliably.

When you ask an LLM a question, you're getting a best guess — not a lookup.

Why Level 1 Matters

- LLMs are powerful assistants — but also flawed.

Why Level 1 Matters

- LLMs are powerful assistants — but also flawed.
- Understanding what they *can't* do is key to using them well.

Why Level 1 Matters

- LLMs are powerful assistants — but also flawed.
- Understanding what they *can't* do is key to using them well.
- Critical thinking is your best defense.

Why Level 1 Matters

- LLMs are powerful assistants — but also flawed.
- Understanding what they *can't* do is key to using them well.
- Critical thinking is your best defense.

Quote: "Every tool is safe — if you know what it can't do."

Roadmap

- ~~Level 1: General~~
- **Level 2: Novice**
- Level 3: Intermediate
- Level 4: Advanced
- Level 5: Expert

Level 2

Novice

"Understanding why LLMs behave the way they do — and why they sometimes fail."

▸ Back to Index

How Language Models Evolved

Symbolic AI (pre-2010)

- Rule-based systems
- Logic + search trees
- Expert systems
- High transparency

Neural AI (2010–2018)

- Feedforward networks
- RNNs / LSTMs / GRUs
- Word2Vec, GloVe embeddings
- Sequence learning, but limited memory

LLMs (2018–Now)

- Transformers: attention-based
- Self-supervised pretraining
- Contextual embeddings
- Emergent generalization

From rules → patterns → prediction at scale.

What Are Tokens?

- LLMs don't read raw words or sentences — they use **tokens**.
- A token is a chunk of text: word, subword, or character.
- Depends on the tokenizer (e.g., Byte Pair Encoding, SentencePiece).

Input:

- `"Understanding transformers is difficult."`

Tokenized:

- `["Understand", "ing", "transform", "ers", "is", "difficult", "."]`

Tokens are the model's language building blocks — not letters nor words.

What Are Embeddings?

- Tokens are converted into high-dimensional vectors called **embeddings**.
- Each embedding captures syntactic and semantic info.
- These vectors are learned during training.

$$\text{Token: ``dog''} \rightarrow \vec{v}_{\text{dog}} \in \mathbb{R}^d$$

Example: $d = 768$ or 1024 for small models.

Embeddings let neural networks process language numerically.

What Is Attention?

- Attention lets the model **weigh the importance** of each token in context.
- Instead of reading token in sequence, it looks at everything — and decides what to focus on.
- Each token attends to other words before making a prediction.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$

Q = query, K = key, V = value — all vectors derived from input embeddings.

Attention is how LLMs "decide what matters" for every token.

What's Going On Inside an LLM?

- Input text → tokens → embeddings.

What's Going On Inside an LLM?

- Input text → tokens → embeddings.
- Embeddings pass through transformer layers (attention + MLP).

What's Going On Inside an LLM?

- Input text $\rightarrow$ tokens $\rightarrow$ embeddings.
- Embeddings pass through transformer layers (attention + MLP).
- Output is a probability distribution over next token.

What's Going On Inside an LLM?

- Input text → tokens → embeddings.
- Embeddings pass through transformer layers (attention + MLP).
- Output is a probability distribution over next token.

$$x_1, \ldots, x_{t-1} \to \text{Embedding} \to \text{Transformer} \to P(x_t \mid x_{<t})$$

No memory. No long-term knowledge. Just context.

Why Do Hallucinations Happen?

- LLMs optimize for likelihood, not truth.
- They output tokens that **sound** likely, not ones that are **verified**.
- No connection to ground truth or external facts.

$$\text{LLM objective:} \quad \max_{\theta} \sum_{t=1}^{T} \log P_{\theta}(x_t \mid x_{<t})$$

Fluency $\neq$ *factuality. Confidence* $\neq$ *correctness.*

LLMs Are Probabilistic

- For a given input, there is no single "correct" output.
- The model samples from a probability distribution:

$$P(x_t \mid x_{<t}) = \mathrm{softmax}(f_\theta(x_{<t}))$$

- Sampling parameters:
  - Temperature T: controls randomness.
  - Top-k: restricts to k highest-probability tokens.
  - Top-p: cumulative probability threshold.

Same prompt $\Rightarrow$ different completions.

Shallow Understanding, Deep Fluency

- LLMs often mimic correct reasoning without understanding it.

Shallow Understanding, Deep Fluency

- LLMs often mimic correct reasoning without understanding it.
- They rely on statistical regularities, not causal models.

Shallow Understanding, Deep Fluency

- LLMs often mimic correct reasoning without understanding it.
- They rely on statistical regularities, not causal models.
- Signs of deep understanding:
  - Generalization to novel contexts
  - Robustness to paraphrase
  - Consistency over long sequences

High perplexity $\Rightarrow$ *confusion. Low perplexity* $\not\Rightarrow$ *understanding.*

Roadmap

- ~~Level 1: General~~
- ~~Level 2: Novice~~
- **Level 3: Intermediate**
- Level 4: Advanced
- Level 5: Expert

Level 3

Intermediate

"How do we interpret model behavior — when we can only see what it says?"

Back to Index

RAG: Expanding Output Context

- RAG augments the LLM's prompt with retrieved documents from an external source.
- This aims to improve grounding and reduce hallucinations by conditioning generation on real-world evidence.
- It operates entirely at the input/output level — no model parameters are changed.
- Paired with source/reference attribution, this offers some semblance of interpretability.

Query $\rightarrow$ **Retriever** $\rightarrow$ Documents $\rightarrow$ **LLM** $\rightarrow$ Completion

RAG offers a partial correction: it reduces hallucination without solving interpretability.

Interpretability vs Explainability

- **Interpretability:** human-accessible understanding of behavior.
- **Explainability:** tools or proxies for providing such insight.
- **Output-based analysis:** observes *what* the model does, not *how*.

At this layer, the output is our microscope.

Output-Based Layer

- **Generated text**: the final tokens predicted.
- **Log probabilities**: likelihoods assigned at each step.
- **Sampling behavior**: controlled by temperature, top-k, top-p.
- **Response variability**: rerun sensitivity to prompt tweaks.

$$P(x_t \mid x_{<t}) = \mathrm{softmax}(f_\theta(x_{<t}))$$

Output-based methods analyze what the model chooses to say.

Generated Text

- The most direct artifact of model behavior.
- Useful for analyzing fluency, coherence, style, and factual consistency.
- Provides first-pass insight into reasoning patterns, errors, and implicit biases.

Example Output (Same Prompt, Two Models)

`Prompt: "What caused the EDSA People Power Revolution?"`

Model A: "The revolution was a response to electoral fraud and declining public trust in the Marcos Sr. administration."

Model B: "The EDSA Revolution was sparked by U.S. interference and elite interests aiming to replace Marcos Sr."

Interpretation: Differences in factual emphasis, framing, and specificity reflect training biases and generalization behavior.

Log Probabilities

- Every token is assigned a log-probability.
- Can reveal:
  - Which words are "expected" or "surprising."
  - Where uncertainty spikes — hinting at confusion or ambiguity.

Example:

```
Prompt: "What was the popular name for the three Filipino priests executed by a garrote in 1872?"
Candidates: GomBurZa (0.91), MaJoHa (0.04), TitoVicJoey (0.01)
```

Low entropy = confidence; high entropy = uncertainty. Useful for detecting hallucination zones.

Sampling Behavior

- LLMs don't output a single deterministic result.
- Sampling parameters:
  - Temperature (T): randomness.
  - Top-k: limit to k most likely tokens.
  - Top-p: nucleus sampling.
- These affect coherence vs creativity trade-off.

Example:

`Prompt: "Write an opening sentence for a sci-fi novel."`

Top-k $=$ *1:* "The ship arrived."

Top-k $=$ *50:* "Starlight spilled across the ruins of Mars as Captain Oloyemi stepped out of cryo."

Sampling analysis helps probe model fluency vs surprise.

Prompt Sensitivity

- Small prompt changes can cause major output shifts.
- Useful for probing robustness and adversarial sensitivity.
- Sensitivity indicates shallow or unstable generalization.
- Interestingly, making prompts more rewarding, more respectful, or more terrifying, forces LLMs to behave differently.

Example:

```
Prompt A: "Summarize the article."
Prompt B: "Please summarize the article."
Prompt C: "Summarize the article in three points."
```

Analysis: How consistent is output across near-equivalent prompts?

Few-Shot Prompting

- More than a usage trick — it acts as a **behavioral probe**.
- Adds example Q&A pairs to the prompt to guide model behavior.
- Tests the model's ability to:
  - Recognize patterns in-context
  - Generalize from limited data
  - Simulate learning without weight updates

Example Prompt

```
Q: Translate "book" to French.
A: livre
Q: Translate "cat" to French.
A:
```

This lets us observe what models infer, not just what they memorize.

Chain-of-Thought (CoT)

- CoT encourages step-by-step output.
- Useful for arithmetic, logic, and reasoning probes.
- Exposes LLMs' ability to follow (or fake) structure.

Example

```
Q: If Sam has 3 Sonny Angels and buys 2 more, how many
angels in total?
A: Sam has 3 Sonny Angels.  He buys 2.  3 + 2 = 5.
Answer:  5
```

Reasoning may be correct, mimicked, or brittle — CoT makes it visible.

Where Do We Go From Here?

Examining the output is only scratching the surface. We could go deeper. Each level opens more of the model — and more responsibility.

Roadmap

- ~~Level 1: General~~
- ~~Level 2: Novice~~
- ~~Level 3: Intermediate~~
- **Level 4: Advanced**
- Level 5: Expert

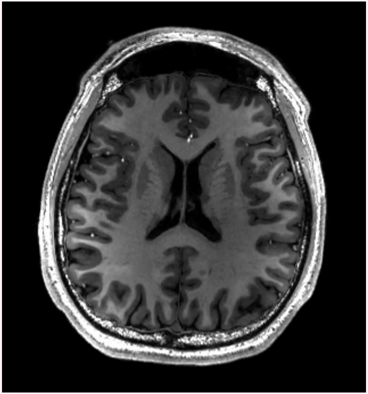
Level 4

Advanced

“Interpretability isn’t just about outputs — it’s about internal structure and function.”

Back to Index

Symptoms vs Causes



- Observing LLM outputs is like watching symptoms.
- You can spot patterns — but not mechanisms.
- To truly diagnose, we need an MRI: internal representations, layer activations, and causal circuits.

Surface clues help — but internal scans reveal the real picture.

Feature-Based Layer

- Goes beyond outputs to internal states — embeddings and activations.
- Methods:
  - Input attribution scores (e.g., Integrated Gradients)
  - Token-level importance (e.g., saliency maps)
  - Attention pattern heatmaps
  - Activation analysis: inspect hidden states layer by layer
- Enables hypothesis about what parts of the network respond to what inputs.

Feature-based interpretability opens the black box slightly

Attribution & Saliency

- **Input attribution scores**: quantify which input tokens influenced a decision.
- **Saliency maps**: gradient-based visualization of sensitivity per token.
- Typical techniques: Integrated Gradients, Gradient × Input, SmoothGrad.
- Useful for surfacing decision boundaries, detecting brittle reasoning.

Example (Sentiment Classification)

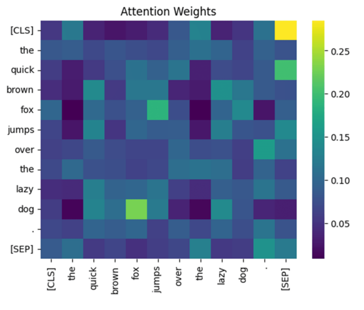
`Input: "I absolutely hated the movie."`

Salient Token: `"hated"` (high attribution score)

These methods expose which parts of the input drive model predictions — and whether they make sense.

Attention & Activations

- **Attention heatmaps**: visualize token-to-token attention weights.
  - Highlight long-range dependencies, alignment errors, or spurious focus.
  - Tools like **BertViz** support interactive exploration of attention patterns across heads and layers.
- **Activation analysis**: inspect internal states layer-by-layer.
  - Probe which neurons/layers fire for certain input types.
  - Can uncover specialization (e.g., syntax, entity tracking).



Example of attention heatmap

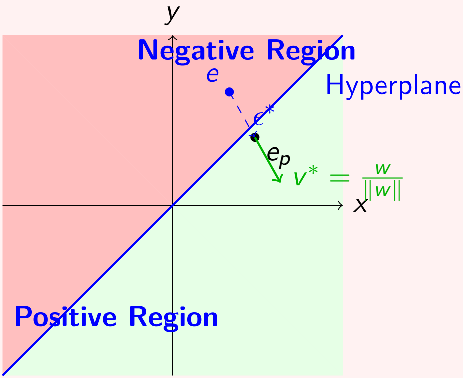
Concept Activation Vectors (CAVs)

- CAVs identify directions in activation space aligned with human-interpretable concepts.
- Originally from **TCAV** (Kim et al., 2018, vision); extended in NLP:
  - **SCAV** — toxicity axes (Xu et al., 2025)
  - **Bias-CAV** — identity framing, societal bias (Catapang, 2025)

CAVs cont'd.

Pipeline:

1. Collect labeled examples (e.g., biased vs. safe).
2. Train classifier on activations (usually linear).
3. Use the separating hyperplane as the concept direction v^*.



Mechanistic Layer

- Goes beyond analysis — aims for causal explanation of internal computation.
- Methods seek to reverse-engineer specific model behaviors:
  - Activation patching
  - Neuron/attention head tracing
  - Modular decomposition
- Goal: not just where or what — but **how** a model implements specific functions.

Mechanistic interpretability tries to open the engine — not just listening to how the engine hums.

Causal Tracing

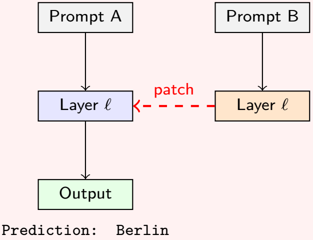
- Replace internal activations with those from other examples.
- If output changes $\Rightarrow$ the swapped layer mattered.
- **Technique:**
  1. Run two inputs (A and B)
  2. Patch activation from B into A at layer ℓ
  3. Observe effect on output
- Used to locate "where" specific behaviors are encoded.

Causal Tracing

Example Prompts:

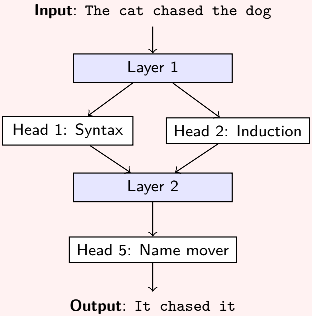
- A: “Paris is the capital of France.”
- B: “Berlin is the capital of Germany.”

→ *Patching B at layer ℓ changes output to “Berlin.”*



Modularity: Circuits, Heads, and Roles

- Transformers have specialized attention heads.
- **Induction head:** copies earlier token.
- **Name mover:** resolves coreference.
- **Syntax head:** tracks grammatical roles.
- Modular behaviors form circuits across layers.
- These circuits act like subroutines.



Goal

Map `input` → `head activity` → `output` to explain behavior causally.

Limitations & Open Challenges

- Scale: most circuit-level work is on small models (GPT-2, etc.)
- Polysemantic neurons: same unit fires for many unrelated concepts.
- Interdependence: behaviors often depend on distributed interactions.
- No guarantees: reverse-engineering is hard, brittle, and incomplete.

Mechanistic insight is promising — but remains aspirational at GPT-4 scale.

Roadmap

- ~~Level 1: General~~
- ~~Level 2: Novice~~
- ~~Level 3: Intermediate~~
- ~~Level 4: Advanced~~
- **Level 5: Expert**

Level 5

Expert

"Are we asking the right questions — or building deeper illusions?"

Back to Index

The Limits of Interpretability

- Not all model behavior is localizable or isolatable.
- Interpretability is often post hoc — narrative over mechanism.
- Most techniques are proxies: we don't know what we're missing.
- Can we ever "understand" a 175B parameter model?
- Do users want interpretability — or just reliability?

Interpretability gives the illusion of insight — but is that insight actionable?

Are LLMs the Right Architecture?

- Current LLMs are:
  - Autoregressive next-token predictors,
  - Largely inductive, not deductive,
  - Memoryless, goal-agnostic, and simulation-driven.
- Symbolic reasoning, long-term planning, abstraction are bolted on (e.g., RAG, agents, function calling).
- Core question: Can prediction truly yield reasoning?

We're training simulators — and asking them to become solvers.

The RAG Illusion

- RAG is often framed as a solution to hallucination — but it's a patch, not a cure.
- It still relies entirely on the LLM to interpret and generate from retrieved data.
- Retrieval does not constrain generation — it merely adds inputs.
- The core model remains:
  - Unverifiable,
  - Unfaithful to sources,
  - Unaware of retrieval provenance.
- The illusion: retrieval = truth $\Rightarrow$ generation = fact. **Not guaranteed.**

RAG is helpful — but it does not resolve the unreliability of the generator itself.

Skepticism: Are LLMs a Dead End?

- Critics argue LLMs:
  - Mimic reasoning, but don't possess it.
  - Encode bias and brittleness at scale.
  - Lack grounding, goals, or understanding.
- Alternative views:
  - Neuro-symbolic hybrids (LLMs + planning + logic).
  - Small, specialized models (MoE, retrieval agents).
  - Whole new paradigms (active inference, memory transformer, JEPA-like architectures (LeCun, 2022)).

LLMs are powerful — but perhaps wrong-shaped for general intelligence.

Philosophy: What Is Understanding?

- Is behavior sufficient for understanding?
- Can internal structure without intentionality be meaningful?
- How do we distinguish simulation vs comprehension?
- Is interpretability about *our* understanding — or the model's?

Quote: "If a model can explain itself — do we trust it more, or fear it more?"

Interpretability is not just a technical problem — it's a philosophical one.

The Future of LLM Research

- Toward models that:
  - Can reflect and reason over their own output,
  - Maintain consistent memory or self-state,
  - Integrate symbolic reasoning with deep priors.
- Interpretability will require:
  - Better tools (scalable probing, causal abstraction),
  - New norms (transparency standards, auditing),
  - A cultural shift from magic to mechanism.

Future models must not only generate — but justify.

Takeaways

- **Interpretability is layered.**
  - From outputs → features → mechanisms.
- **LLMs are powerful, but limited.**
  - Fluent, but not grounded.
  - Consistent, but not reflective.
- **Techniques exist — but insight is fragile.**
  - Attribution, patching, probing offer glimpses.
  - Causality remains elusive at scale.
- **The future may lie beyond LLMs.**
  - Toward modularity, memory, planning, and predictive representations.
  - Paradigms like JEPA reimagine what reasoning could mean.

Interpretability is not just a tool — it's a mirror. It reveals both the model and our assumptions about intelligence.

End

"By far, the greatest danger of artificial intelligence is that people conclude too early that they understand it."
— Eliezer Yudkowsky

Thank you for listening!

If you have questions, please email me at `jasperkylecatapang@gmail.com`.

Linkedin: jcatapang

Backup Q&A Index

- Opinion: Do LLMs Really Understand What They Say?
- Opinion: Are LLMs a Dead End?
- How Is Attention Different from Memory?
- Why Do We Expect Deterministic Outcomes from Randomness?

Attention vs. Memory?

- **Attention** is *contextual weighting* — deciding what tokens to focus on now.
- **Memory** implies long-term storage and retrieval — across time or sessions.
- Transformers attend to all tokens in the current window — but forget everything after.
- *No state is saved* between prompts unless explicitly engineered (e.g., with memory modules).

Attention is selective focus. Memory is persistent storage. LLMs natively have one — not the other.

Back to Index

Randomness → Deterministic Outcomes

- LLMs are probabilistic by design — sampling is non-deterministic unless constrained.
- But humans **perceive language deterministically** — we expect consistency and intent.
- This mismatch creates confusion:
    - "Why did it answer differently?"
    - "Can I trust this answer?"
- Determinism can be forced (e.g., greedy decoding), but at the cost of creativity and robustness.

LLMs are not bugs — our expectations of determinism are.

Back to Index

Opinion: Are LLMs a Dead End?

- LLMs are powerful — but they're not the final form of intelligence modeling.
- They are **inductive engines**: great at pattern recognition, but weak at causal reasoning, planning, abstraction.
- They simulate thought — not generate it.
- Future systems will likely integrate:
  - Symbolic reasoning
  - Explicit memory and planning
  - Multimodal grounding and feedback
- LLMs aren't a dead end — but they are **a phase**. A stepping stone, not a destination.

We need systems that reason — not just simulate reasoning.

Back to Index

Opinion: Can LLMs Really Understand?

- This is an **open debate** in the AI community — with strong views on both sides.
- **Skeptical view (LeCun, Bender):**
  - LLMs simulate language without grounding or intent.
  - Fluency emerges from statistical patterns, not comprehension.
  - No world model, no goals — just token prediction.
- **Pro-understanding view (Hinton, Mitchell):**
  - Some argue understanding is *emergent*, not binary.
  - LLMs exhibit flexible reasoning, analogy, and abstraction in context.
  - If their behavior mirrors comprehension, should we deny it?

Perhaps LLMs don't "understand" like we do — but maybe they understand differently.

Back to Index