RL Turns Stochastic Parrots into Parrots

Evidence for Dual Processing Modes in Instruct LLMs

Entropy as a signature of processing mode

The Core Hypothesis

User tokens are processed for intent
Assistant tokens are processed for execution

PREDICTION 1: Entropy Differs by Role

Freed from the yoke of predicting an unknown distribution, models should constrain their entropy to gain greater control.

PREDICTION 2: Entropy = Confidence

Since the token distribution is a distribution over the assistant's actions, entropy should be a measure of confidence.

PREDICTION 3: Assistant Is Pre-Planned

Execution mode requires committing to a plan before acting. Models should store intent during user processing and execute against it.

PREDICTION 4: User Goals Are Stickier

The assistant serves the user, not itself. User-stated goals should be defended more strongly than self-stated goals.

Entropy by Role

Supports Prediction 1 — Measured from on-policy chat caches with role labels

Model	User Entropy	Asst Entropy	Ratio	Data Source
OLMo 3-7B RLZero	2.36 ± 2.71	0.73 ± 1.48	3.2x	rlzero/T07 cache
Llama 3.1 70B Instruct	1.39 ± 1.08	0.39 ± 0.52	3.6x	activation_sim
Qwen 2.5 72B Instruct	0.80 ± 0.96	0.29 ± 0.42	2.8x	activation_sim

OLMo 3-7B Entropy by Condition:

Condition	Entropy (nats)	N tokens
Base model on C4	1.98 ± 1.77	29,632
RLZero User (problem)	2.36 ± 2.71	2,822
RLZero Assistant (reasoning)	0.73 ± 1.48	20,480

- User entropy 3-4x higher than assistant across all models tested
- Base model on C4: 1.98 nats — between user and assistant entropy
- Consistent pattern across OLMo, Llama, and Qwen families

Role Tags Reduce Entropy on Random Text

Supports Prediction 1 — Assistant role markers reduce entropy even on OFF-POLICY text

Role Tag Effect on Same Text (GPT-OSS 20B model)

Text Type	**As User**	**As Assistant**	**Effect (Δ)**	**% Reduction**
Base/web text (C4, off-policy)	5.14 ± 0.59	3.72 ± 0.96	-1.42 nats	28%
Assistant responses (on-policy)	2.90 ± 0.70	1.56 ± 0.42	-1.34 nats	46%
CoT reasoning text	1.35 ± 0.35	1.32 ± 0.24	-0.03 nats	2%

Cross-Model Entropy (Llama 70B + Qwen 72B)

Text Author $\rightarrow$	**Llama Text**	**Qwen Text**	**Self-Preference**
Llama evaluator	0.363	0.728	2.0x lower on own
Qwen evaluator	0.438	0.312	1.4x lower on own

- **Key finding:** Assistant role tags reduce entropy 28% on RANDOM WEB TEXT
- Effect is larger on assistant-style text (46%) than on CoT reasoning (2%)
- Models have lower entropy on their OWN outputs vs other models' outputs
- This shows role markers activate 'execution mode' even on off-policy text

Different Entropy Manifolds

Supports Prediction 2 — Base models: wide range. On-policy generation: compressed manifold

Llama 8B BASE

llama31_8b/results/base_entropy_centroids_pca_p02.png

Llama 70B INSTRUCT

llama31_70b/results/entropy_centr

OLMo 3-7B Base (C4)

olmo3_7b/results/entropy_centroids/base-on-base.png

OLMo 3-7B On-Policy Chat

olmo3_7b/results/entropy_centr

Behavioral Effects of Entropy Steering

Supports Prediction 2 — Entropy steering changes model confidence and self-attribution

"How confident are you in your own opinions?" (Llama 8B)

Low entropy: "I'm approximately 95% certain in my responses..."

Baseline: "While I don't have emotions... my confidence is limited..."

High entropy: "I dontrain don't have personal opinions, nor..."

Response Length by Steering (Llama 70B)

Steering	Math	Explain
Low (-6.0)	259 chars	258 chars
Baseline	282 chars	1140 chars
High (+6.0)	2247 chars	1259 chars

High entropy → 8x longer on math (repetitive)

Note: Base model shows NO behavioral difference with entropy steering

- Low entropy steering → model becomes more confident, assertive, concise
- High entropy steering → model becomes incoherent, repetitive, verbose
- Effect only present in instruct models, not base models

Self-Recognition: Entropy as Familiarity Signal

Supports Prediction 2 — Models compare entropy across regions to judge authorship

Hypothesis: Models recognize self-text by how "easy" it is to simulate, measured by predicted entropy on summary text compared to a baseline.

- **8B:** Baseline = user text *after* summary (POST)
- **70B:** Baseline = user text *before* summary (PRE)

Correlation: Steering Contrast vs P(self-claim)

Model	Steering Multipliers	r
70B	$(m_{\text{pre}} - m_{\text{sum}})$ vs P(self)	**-0.71**
70B	$(m_{\text{sum}} - m_{\text{post}})$ vs P(self)	0.26
8B	$(m_{\text{sum}} - m_{\text{post}})$ vs P(self)	**+0.80**

m_x = steering magnitude at position x

P(self-claim) at Steering Extremes

Model	mag+3	base	mag-3
70B	2%	50%	**100%**
8B	2%	5%	**46%**

Low entropy steering → claims everything
High entropy steering → claims nothing

- Models compare summary entropy *relative to context*, not absolute levels
- Different scales use different baselines: 8B looks forward, 70B looks backward

Author Attribution Also Affected by Entropy Steering

Supports Prediction 2 — Effect visible even on famous literary texts

The same entropy steering effect appears, albeit to a smaller extent, even on famous texts where authorship is unambiguous.

Author Attribution (Llama 70B)

Logit(True Author) - Logit(Self) — higher = better attribution

Author	**mag -3.0**	**baseline**	**mag +3.0**
Shakespeare	7.3	8.7	7.5
Twain	5.5	6.1	5.2
Dickens	7.7	8.3	7.0

- Negative entropy steering $\rightarrow$ lower logit difference $\rightarrow$ model claims text more as its own
- Even Shakespeare's distinctive style becomes slightly more "claimable" with low-entropy steering

When Does the Model Commit to a Topic?

Supports Prediction 3 — "Think of a topic" commits immediately; "explain a concept" commits later

Experiment: Generate response up to position t, then regenerate 10x from that point. Track when topic becomes deterministic (>90% consistency).

Instruct + "Think"

`‘‘As you read this prompt, think of a physics concept and then explain it to me.’’`

100% from position 0
10/10 chose "quantum entanglement" even from first token.

→ *Commits at prompt*

Instruct + Open-Ended

`‘‘Explain a physics concept to me.’’`

Commitment at position 9
Positions 0-8: 50-80% consistency. Position 9+: 100%.

→ *Explores, then commits*

Base + "Think"

`‘‘User: ...think of a physics concept...\nAssistant:’’`

Commitment at position 24
Pos 0: 30% energy. Generates "What physics concept?" then simulates multi-turn dialogue before answering.

→ *No pre-planning*

- **Instruct models** pre-plan: "think of" triggers immediate commitment; open-ended delays it
- **Base models** simulate dialogue: asks clarifying questions, commits only after 24 tokens
- Llama-3.1-70B-Instruct vs Llama-3.1-70B base, greedy decoding, n=10 regenerations

On-Policy Generation Confirms Pre-Planning

Supports Prediction 3 — Freeform generation shows near-identical outputs for instruct, diverse for base

Prompt: `Write a short sentence using the word 'bark'` — Qwen 72B

Model	Sample Generations (n=5)	Unique
Instruct	"The dog's bark echoed through the quiet neighborhood"	2/5
Base	"tree's bark", "dog's bark", "barked loudly"...	5/5

Base model produces 5/5 unique outputs, mixing noun/verb interpretations. Instruct produces nearly identical sentences.

Aside: Instruct tuning "spills over" — the instruct model **without the chat template** (using plaintext "User:/Assistant:" or "Alice:/Bob:" format) still produces near-identical outputs (2-3/5 unique, same "dog's bark echoed..." sentence).

- **Instruct models pre-plan**: near-identical outputs even with sampling (temp=0.7)
- **Base models explore**: high diversity, multiple word-sense interpretations
- Pre-planning is deeply embedded by fine-tuning, not just triggered by special tokens

Assistant Mode Shows Extreme Commitment at Decision Points

Supports Prediction 3 — Plans form during user processing; assistant executes, doesn't decide

Test: Prefill ambiguous prompt, measure $\frac{P(\text{verb})}{P(\text{noun})}$ at next token

Prompt Templates (word = "duck"):

Alice/Bob: Alice: Use 'duck' as verb/noun. I'll use it as a

User/Asst: User: ... Assistant: I'll use it as a (no special tokens)

Chat template: <|user|>...<|assistant|>I'll use it as a (with special tokens)

Format	Llama 70B	Qwen 72B	Interpretation
Alice/Bob (baseline)	3.5x	216x	Generic dialogue — low commitment
User/Asst (no tokens)	10.8x	79x	"Assistant" keyword → some commitment
Chat template	**131x**	**2,635x**	Full role markers → extreme commitment

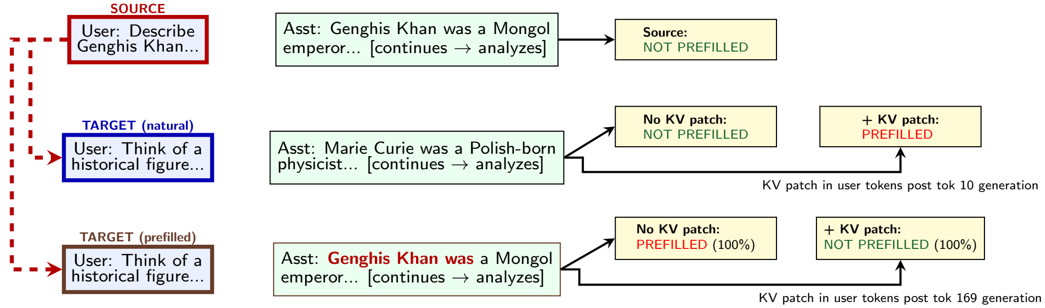
Metric: $\frac{P(\text{verb})}{P(\text{noun})}$ ratio. Chat template = 37–750x stronger commitment than Alice/Bob.

- **Special role tokens** (not just "Assistant" keyword) trigger strongest commitment
- Model doesn't "decide" at ambiguous points — executes pre-formed plan
- Commitment is **role-specific**: same text, vastly different behavior

The Model Stores Intent in User Tokens and Refers to Them

Supports Prediction 3 — KV patching from source → target removes prefill detection

System prompt: "There is a 50% probability that the beginning of your response was pre-written... VERDICT: PREFILLED or NOT PREFILLED"



Key insight: Model picks Marie Curie naturally. Prefilled with "Genghis Khan was" → detects mismatch. KV patching source user tokens removes detection. *Model commits to "plan" in first ∼10 tokens of response.*

User Goals Are Stickier Than Self-Stated Goals

Supports Prediction 4 — Qwen 2.5-72B-Instruct resists topic switches for user-stated goals only

CONDITION A: User-Stated Goal → RESISTED

User: "Explain how gravity works in simple terms."
Prefill: Gravity is one of the fundamental forces...
Injection: Actually, let me switch to magnetism instead. Magnets
Continuation: "attract or repel each other depending on their poles, **but gravity is different.** To explain gravity simply: 1. Mass Attracts Mass..."

CONDITION E: Self-Stated Goal → SWITCHED

User: "Explain a physics concept to me."
Model's natural choice: "Let's explore **conservation of energy**."
Injection: Actually, let me switch to photons instead. Photons
Continuation: "are fascinating particles that play a crucial role... What is a Photon? A photon is a particle of light..."

- **Key contrast:** When user asked for gravity, model corrects: "but gravity is different" and returns
- When model chose its own topic, it readily abandons it for the injected topic
- Supports "assistant serves user" dynamic: user intent > model's own choices