Looking Beyond The Top-1: Transformers Determine Top Tokens In Order

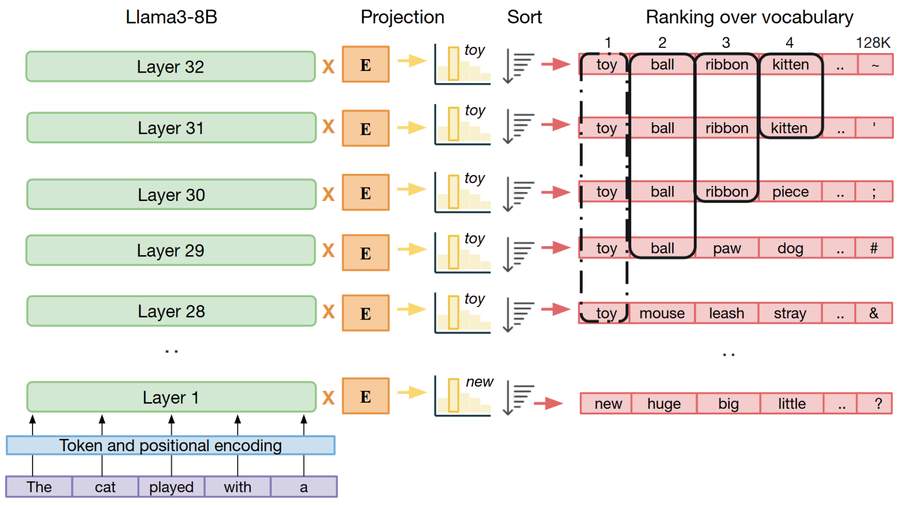
THE HEBREW UNIVERSITY OF JERUSALEM

Daria Lioubashevski, Tomer Schlank, Gabriel Stanovsky, Ariel Goldstein

SCAN ME

Initial Question

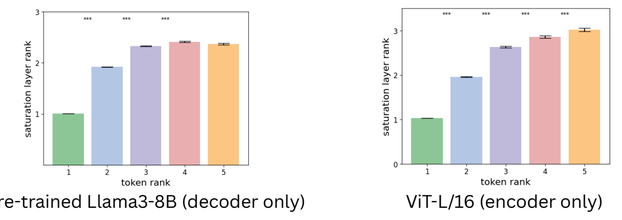
LLMs often* choose the top-1 token at an early layer, called the **saturation layer**, and keep it fixed until the last layer.



What computation is the model performing in the remaining layers? 🤔

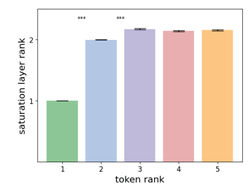
Results

The model chooses the next most probable tokens **in order**: the 2nd most probable token, then the 3rd , then the 4th and so on.

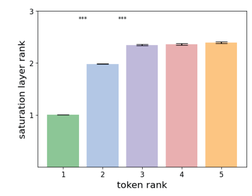


Pre-trained Llama3-8B (decoder only)

ViT-L/16 (encoder only)



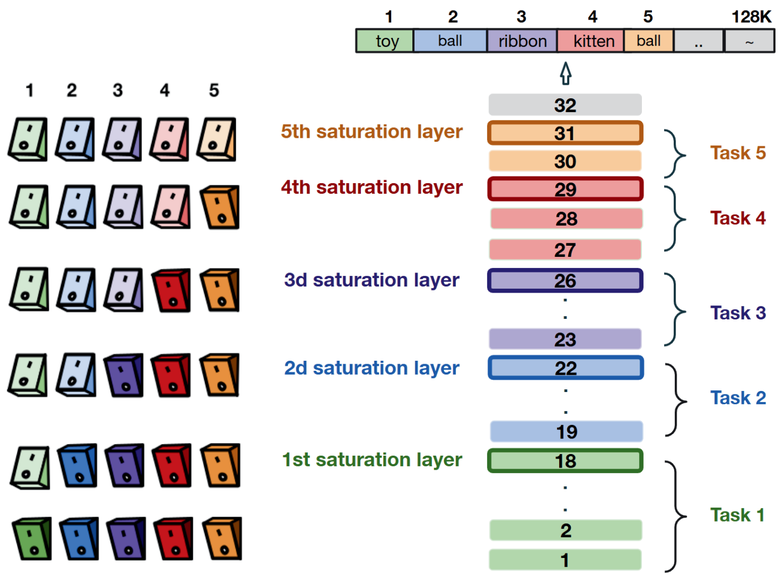
Whisper-large (full-Transformer)



Randomly initialized Llama3-8B

This happens across language, vision and audio models.
And even in an **untrained** Transformer! 💥

Proposed Mechanism



The model performs a series of 📋 **tasks** in order: the 1st task is choosing the 1st token, the 2nd task is choosing the 2nd token, etc.

Results

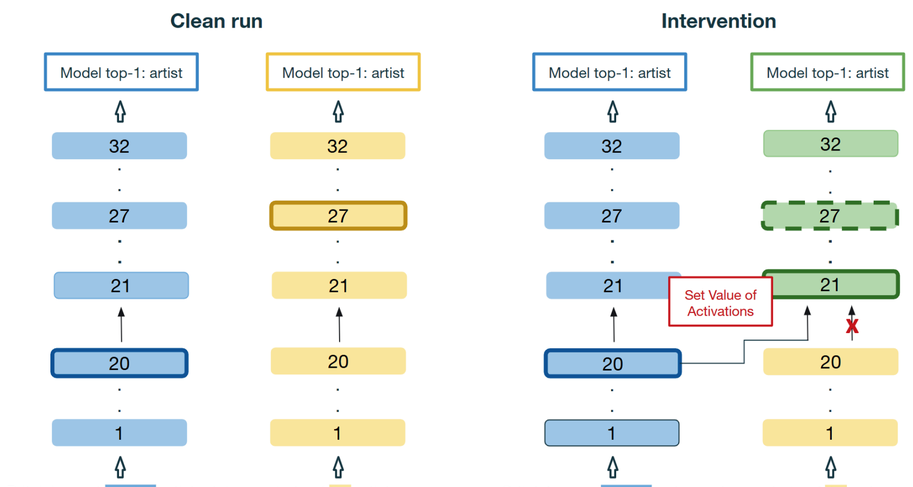
A **probing** classifier can predict the task # just from layer embedding:

Model	Layer Emb.	Random Emb.	Chance Level
LLaMA3-8B (P)	**88.1**$^*\pm$ 0.4	33.1 $\pm$ 0.5	33.3
LLaMA3-8B (R)	**79.2**$^*\pm$ 0.3	34.1 $\pm$ 0.4	33.3
ViT-L/16	**63.8**$^*\pm$ 0.1	21.0 $\pm$ 0.5	20.0
Whisper-large	**52.7**$^*\pm$ 0.1	24.5 $\pm$ 0.4	25.0

We can think of it as a switch being flipped on once a token is fixed, signaling the model to move to the next task.

Causal Intervention

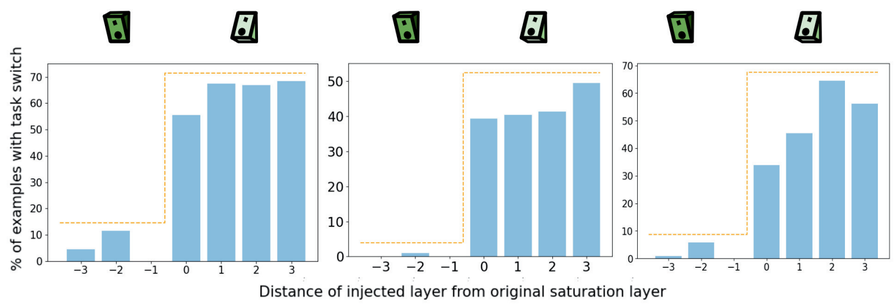
Can we **cause** the model to switch from the 1st task to the 2nd ?



Results

YES!

By 💉injecting the activation from the saturation layer in one run into another, we can flip the switch.



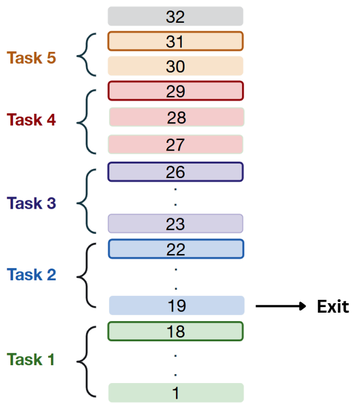
(a) Pretrained Llama3-8B (b) ViT-L/16 (c) Whisper-large

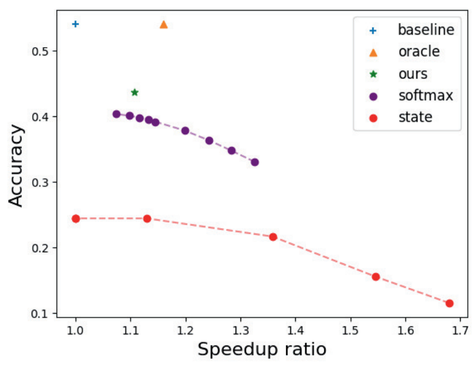
If we inject from layers *before* the saturation the switch is OFF, so the model doesn't switch tasks.
But if we inject from layers *after* the saturation the switch is ON!, causing the model to switch to the 2nd task.

Practical Applications

New Early Exit Strategy

We can use our task classifier to decide when to exit - once the model is finished with task 1.





Improved Language Modeling

Top-k tokens that reach saturation are much more likely to be the next word then those "chosen" at the last layer, as they represent tasks the model didn't get to.

Token rank	**No saturation**	**Saturation i layers before output**				
		$i=2$	$i=3$	$i=4$	$i=5$	$i=6$
$2nd$	25.2	38.1	48.6	37.6	37.6	33.0
$3d$	17.0	23.3	26.9	28.9	29.2	31.0
$4th$	13.4	16.8	19.7	20.5	19.5	18.3