Residual Connections and the Causal Shift:
Uncovering a Structural Misalignment in Transformers

Jonathan Lys[1], Vincent Gripon[1], Bastien Pasdeloup[1], Axel Marmoret[1]
Lukas Mauch[2], Fabien Cardinaux[2], Ghouthi Boukli Hacene[2]
[1]IMT Atlantique, Lab-STICC, UMR CNRS 6285, Brest, France · [2]Sony Europe Ltd., Stuttgart Technology Center, EUREC, Germany

Residual Connections & Input–Output Leakage

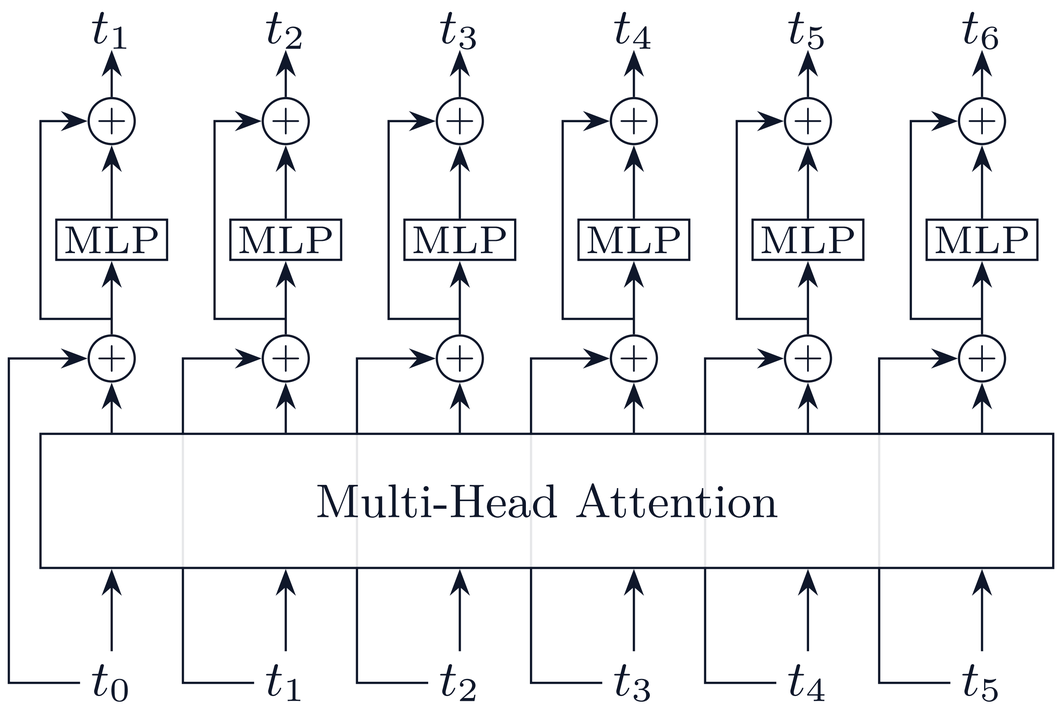
Autoregressive Transformers achieve training parallelism via **causal masking**: position i attends strictly to preceding tokens $t_{\leq i}$ to predict the next token t_{i+1}.

However, the standard **residual identity stream** continuously injects the current token embedding t_i across all layers at position i:

$$x_{l+1} = x_l + F_l(\mathrm{LN}(x_l))$$

Resulting Input–Output Leakage:

- The residual stream acts as a conservative bias anchored to input token t_i.
- Next-token supervision pulls the representation toward prediction target t_{i+1}.
- While t_i provides a useful local prior, it **interferes** when predictions require longer range contextual aggregation (e.g. repeating a sentence).



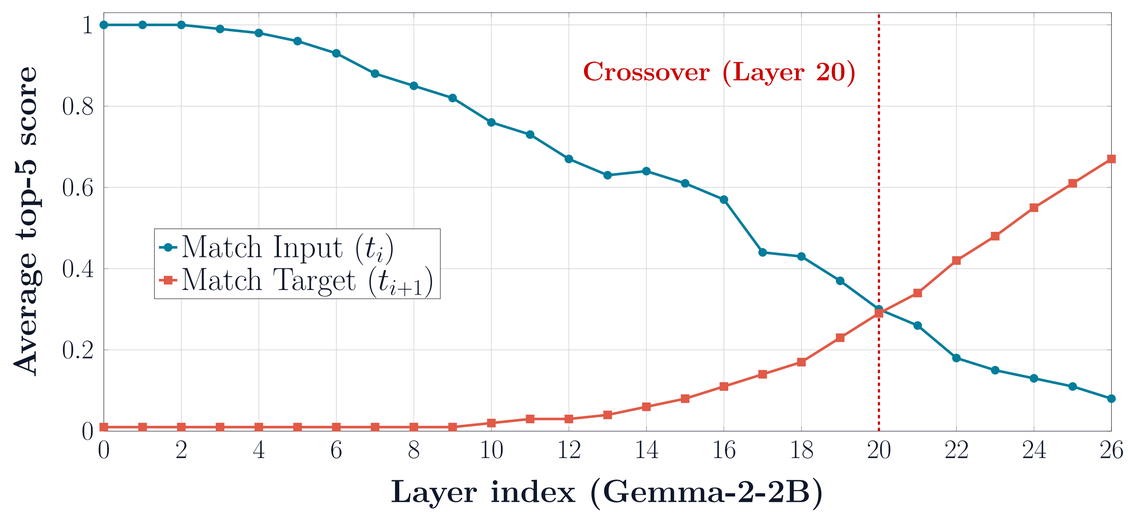
Core Questions:

1. Shift Location: At what depth do internal representations transition from input-anchored (t_i) to target-oriented (t_{i+1})? Is this depth consistent across models?

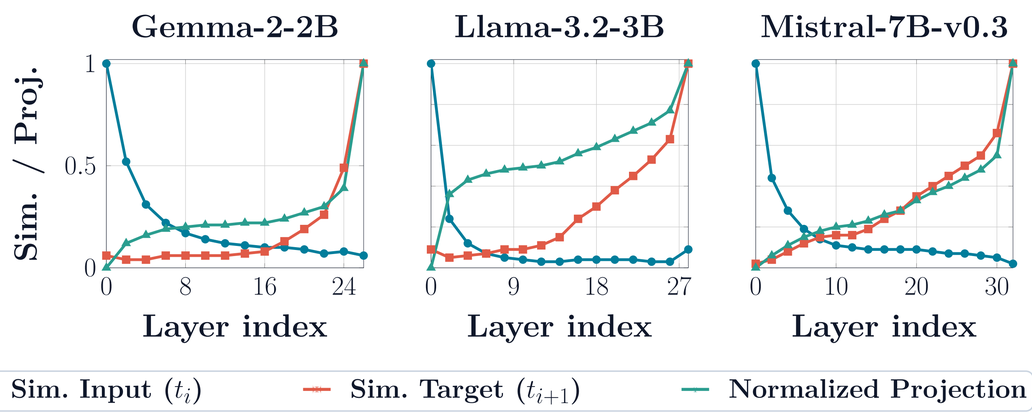
2. Architectural Mitigation: Can residual connections be adjusted to relieve this tension while preserving gradient stability?

Localizing the Shift Across Architectures

Logit-Lens Probing: Using the tied embedding matrix, we track whether intermediate representations are aligned with the current token (t_i) or the prediction target (t_{i+1}). In Gemma-2-2B, representations remain strongly input-aligned in early layers before progressively shifting toward the target in later layers.



Continuous Probing Across Model Families: To avoid discretization artifacts, we measure cosine similarity to input/target embeddings and projection along the input–output axis. Gemma-2-2B, Llama-3.2-3B, and Mistral-7B-v0.3 all exhibit the same qualitative transition from input-anchored to target-oriented representations.



Consistent Across Architectures:

Both discrete logit-lens decoding and continuous metrics confirm that representations shift from input-anchored to target-aligned **late across all evaluated model families**.

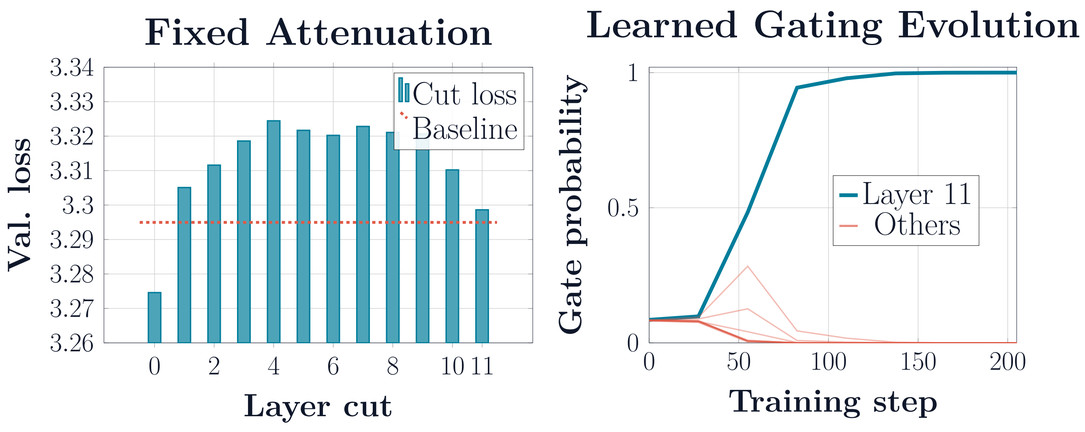
Residual Decoupling & Learnable Gating

Fixed attenuation: Hard suppression ($\alpha = 0$) eliminates the identity gradient highway, degrading optimization. We propose **soft residual attenuation**:

$$x_{l+1} = \alpha\, x_l + F_l(\mathrm{LN}(x_l)), \qquad \alpha \in (0, 1]$$

Learned attenuation: A per-sequence router predicts a gate vector ($g(x) \in \Delta^L$) over all layer indices to decide how much to attenuate the residual connection.

$$x_{l+1} = (1 - \alpha \cdot g(x)_l)\, x_l + F_l(\mathrm{LN}(x_l))$$



Training Dynamics Impact:

Fixed Cuts: Forcing the attentuation to Layer 0 yields the best static validation performance. Attenuating Layer 11 (the last layer) yields the second best performance.

Learned Gating: Dynamic routing automatically concentrates attenuation mass onto the final layer over training, discovering useful intervention without discrete manual tuning.

Evaluation on Downstream NLP Benchmarks

The 150M-parameter models are evaluated for next-token accuracy on four benchmarks.

	Next-Token Accuracy (%)			
Benchmark	**Baseline**	**Fixed-Cut**	**Learnable**	**Gain (Δ)**
Wikitext	28.46	28.62	**28.75**	**+0.29**
LAMBADA	32.99	33.38	**33.52**	**+0.53**
OpenWebText	34.74	34.84	**35.50**	**+0.76**
Fineweb-Edu	38.87	38.98	**39.19**	**+0.32**

Key Empirical Insights:

- **Consistent Improvements:** Learnable gating outperforms both baseline and fixed cuts across all benchmarks (+0.29 to +0.76 points).
- **Lightweight and automatic selection:** The learned gate automatically identifies a useful attenuation depth incurring minimal overhead.
- **Preserves Optimization Flow:** Gradual learned gating maintains stable gradient flow throughout training, avoiding optimization collapse.

Code & Reproducibility

Implementation:

`github.com/jonathanlys01/causal-shift`

Contact: `jonathan.lys@imt-atlantique.fr`