Finding Year Comparison Circuit in LLMs

Yutong He

yutong.he@stud.uni-heidelberg.de

Mechanistic Interpretability 24/25 WS

May 3rd, 2025

1 Research question

How does a medium-sized LLM internally decide whether one 4-digit year is later than another? Is it able to find out the former digits weights more than the latter ones when comparing temporal relations between two years?

This project focus on finding the mechanistic circuit, from the first token embedding, through attention heads and residual streams, to the final answer logit, using methods including probing and activation patching.

2 Data

Prompt Design

The model is asked to answer if one year is earlier than the other, and the labels are binary as 'Yes'/'No'.

```
"<system> You are a bot that can compare years. <system>
Please answer: Is 1933 earlier than 1642? Answer with 'Yes' or 'No'.
A:"
```

Model Selection

Phi-3-mini-4k-instruct (≈ 3.8 B params) can reach a zero-shot accuracy around 95% on this task.

Importantly, this model tokenises each 4-digit year into 5 tokens, which looks like this: `1937` → `['_', '1', '9', '3', '7']`. This is convenient for the digit-wise analysis we need later.

The dataset is randomly generated with years ranging from 1000 to 2020. Among 150 samples, only the prompt questions that the model can correctly answer are included, resulting in a total of 134 samples.

For further ablation studies, an additional dataset with years ranging from 1800 to 2020 is constructed to observe the impact of digit frequency at each position.

3 Experiments & Results

Probing

For this experiment, I am trying to find out how the model encodes every digit in a 4-digit year. I apply probing approaches to observe whether there exist linear structures for digit-wise token embeddings among all layers, as well as whether the patterns alter between the two years appeared in a prompt.

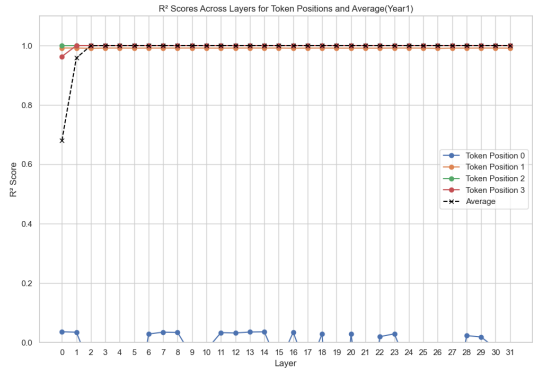
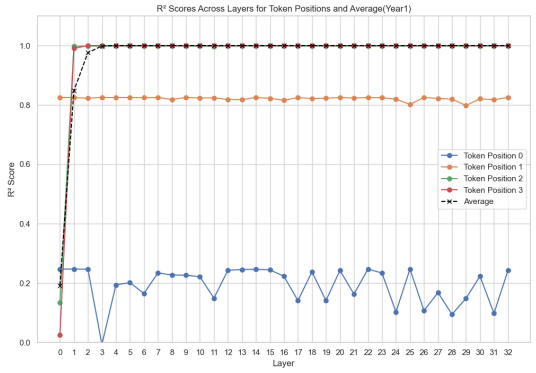
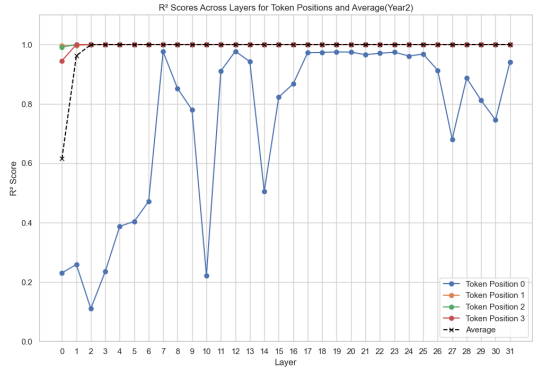
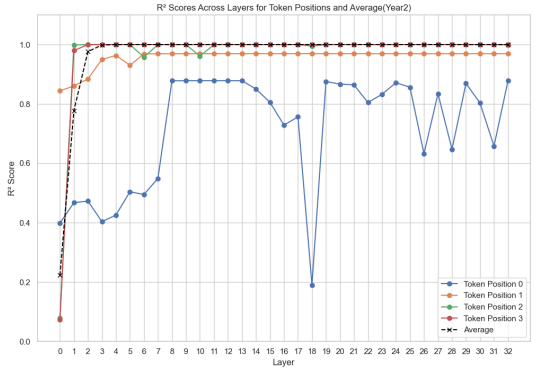
1) Methodology

1. Train a simple Ridge `reg` to see if it can fit the embedding of each layer after the residual modules to the corresponding years, and visualize the R^2 score of `reg` over layers and different digit positions or at the average level overall.
2. Train a simple linear regression `lr` to fit projection (`embedding @ normalized(reg.coef)` as time direction of this dimension) to the corresponding years, and visualize the projected years and actual years.
3. A simple ablation for the number ranges for each token position: 1800-2020 vs 1000-2020

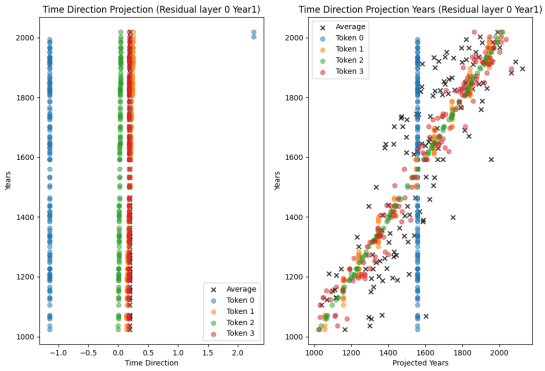
Relative codes are implemented in `probing.py`.

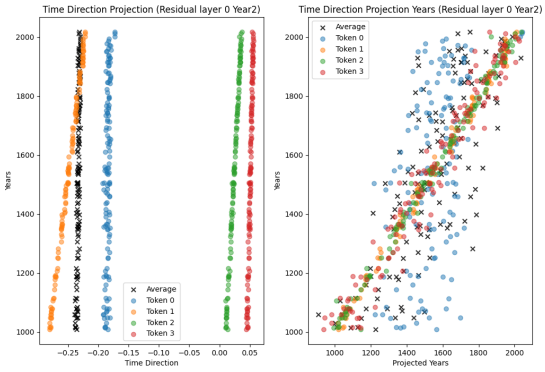
2) Results

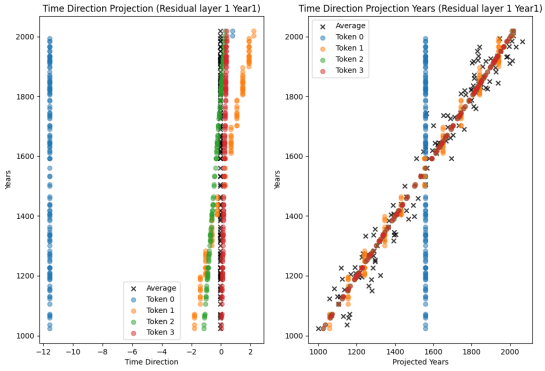
Note: Token position 0 refers to the thousands digit, while token position 3 refers to the ones digit.

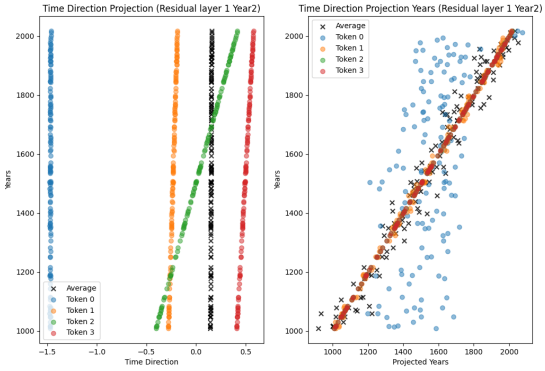
R^2 Year Range: 1000-2020	R^2 Year Range: 1800-2020
	
	

Time Direction Year Range: 1000-2020

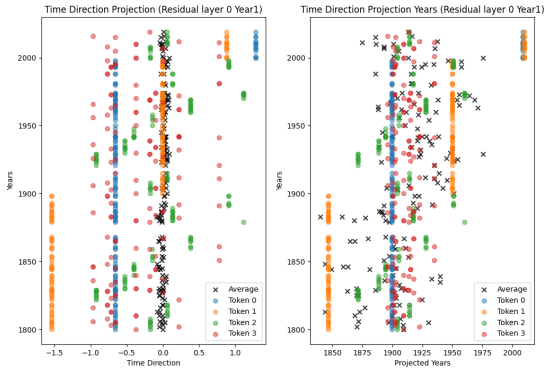


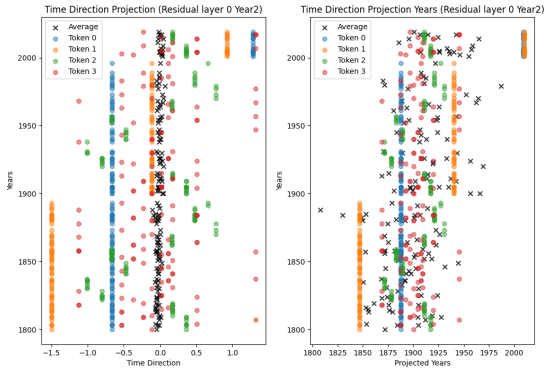


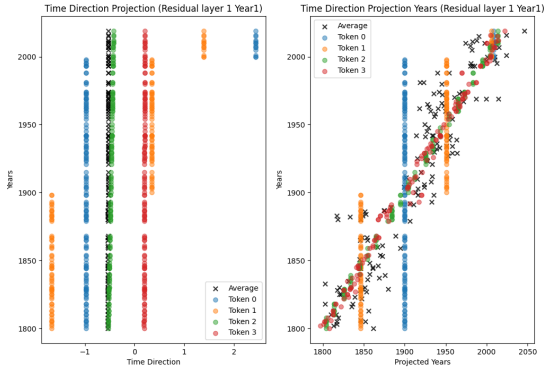


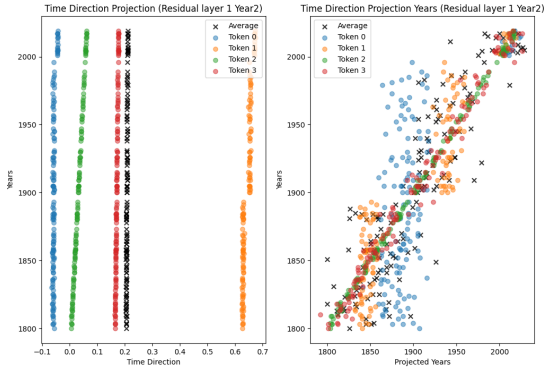


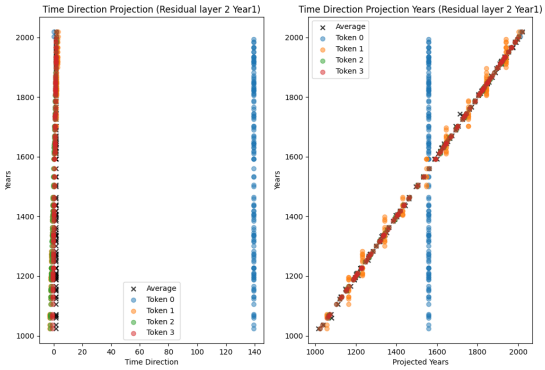
Time Direction Year Range: 1800-2020

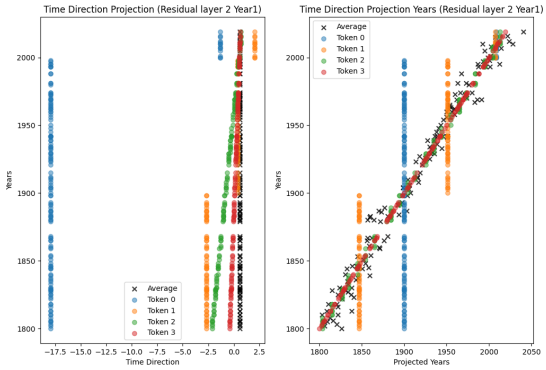


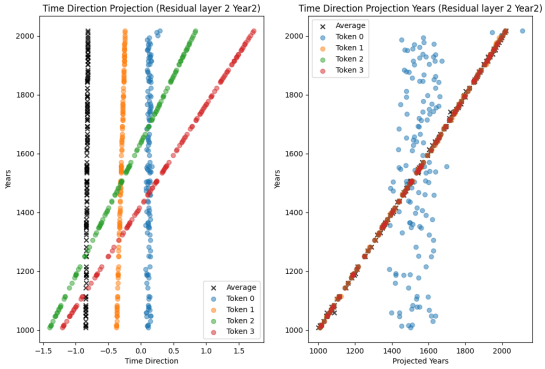


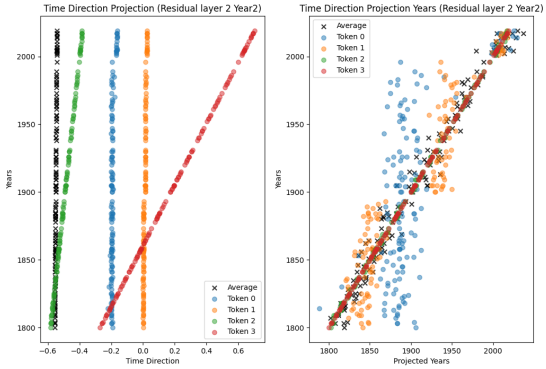


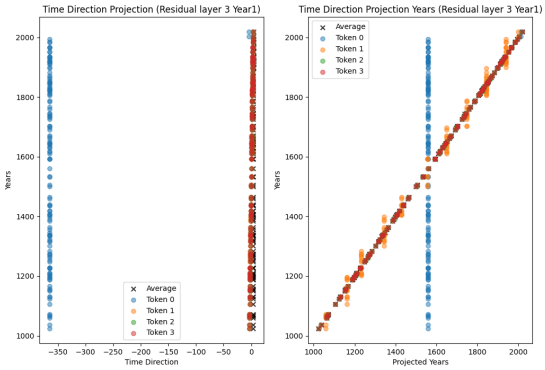


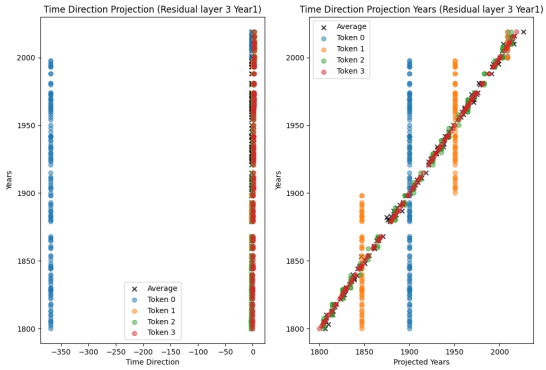

Time Direction Projection (Residual layer 2 Year1)
Time Direction Projection Years (Residual layer 2 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

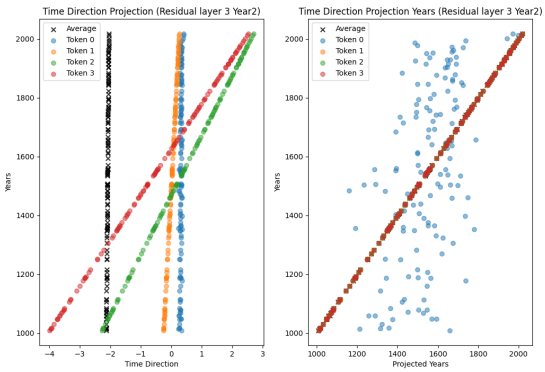

Time Direction Projection (Residual layer 2 Year1)
Time Direction Projection Years (Residual layer 2 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

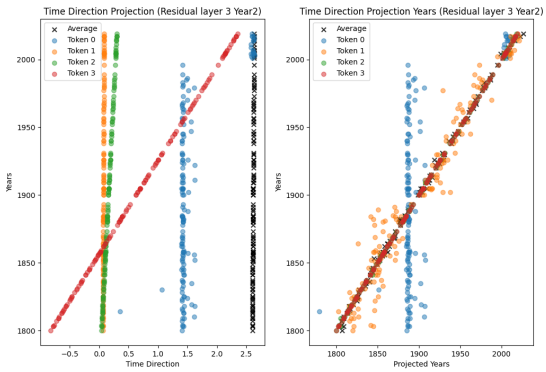

Time Direction Projection (Residual layer 2 Year2)
Time Direction Projection Years (Residual layer 2 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

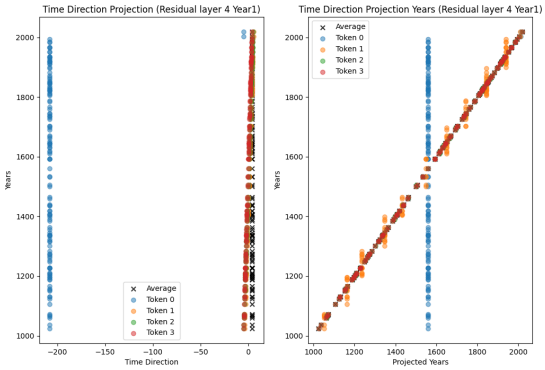

Time Direction Projection (Residual layer 2 Year2)
Time Direction Projection Years (Residual layer 2 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

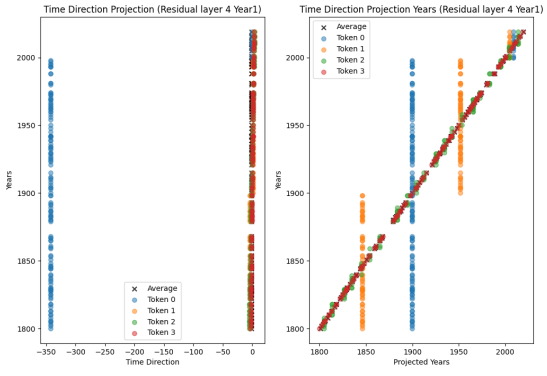

Time Direction Projection (Residual layer 3 Year1)
Time Direction Projection Years (Residual layer 3 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

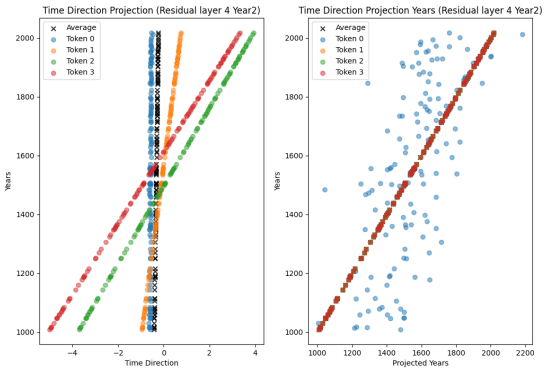

Time Direction Projection (Residual layer 3 Year1)
Time Direction Projection Years (Residual layer 3 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

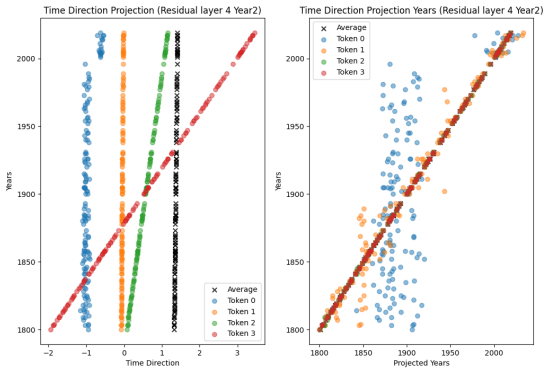

Time Direction Projection (Residual layer 3 Year2)
Time Direction Projection Years (Residual layer 3 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

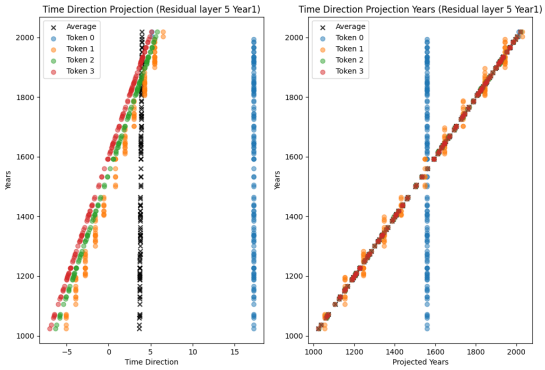

Time Direction Projection (Residual layer 3 Year2)
Time Direction Projection Years (Residual layer 3 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

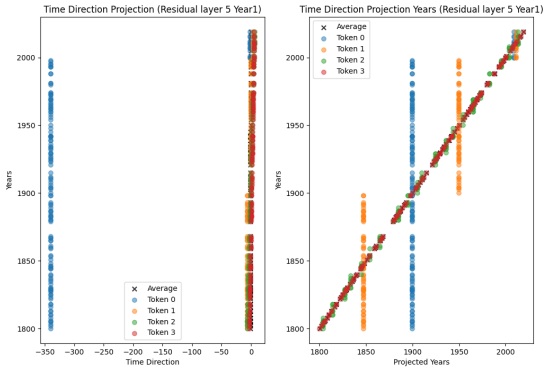

Time Direction Projection (Residual layer 4 Year1)
Time Direction Projection Years (Residual layer 4 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

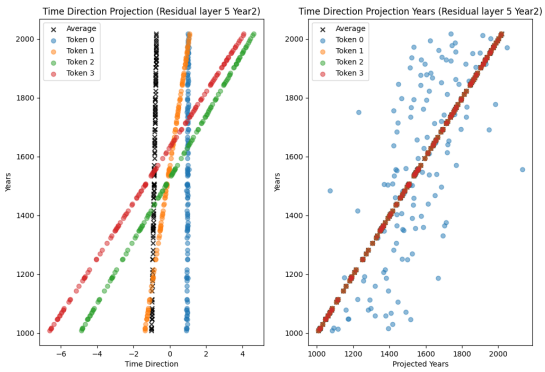

Time Direction Projection (Residual layer 4 Year1)
Time Direction Projection Years (Residual layer 4 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

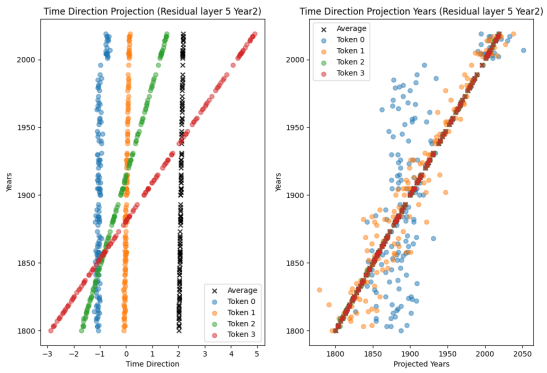

Time Direction Projection (Residual layer 4 Year2)
Time Direction Projection Years (Residual layer 4 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

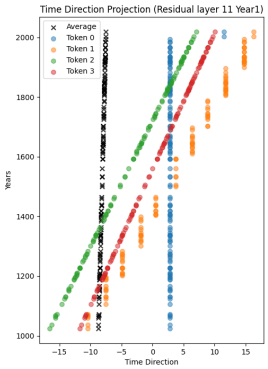

Time Direction Projection (Residual layer 4 Year2)
Time Direction Projection Years (Residual layer 4 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

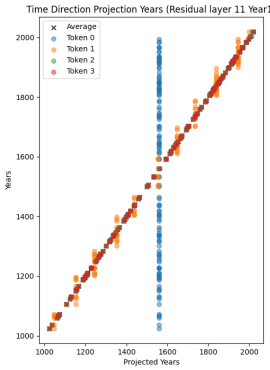

Time Direction Projection (Residual layer 5 Year1)
Time Direction Projection Years (Residual layer 5 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

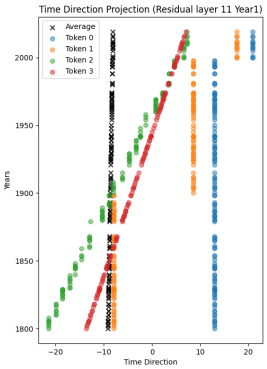

Time Direction Projection (Residual layer 5 Year1)
Time Direction Projection Years (Residual layer 5 Year1)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

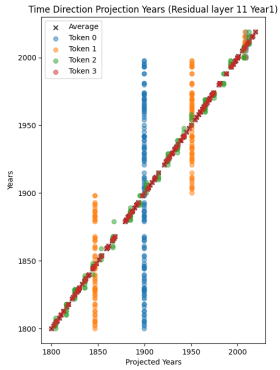

Time Direction Projection (Residual layer 5 Year2)
Time Direction Projection Years (Residual layer 5 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

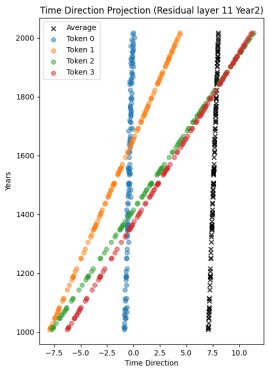

Time Direction Projection (Residual layer 5 Year2)
Time Direction Projection Years (Residual layer 5 Year2)
Years
Time Direction
Projected Years
Average
Token 0
Token 1
Token 2
Token 3

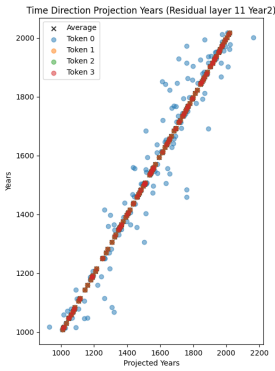

Time Direction Projection (Residual layer 11 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

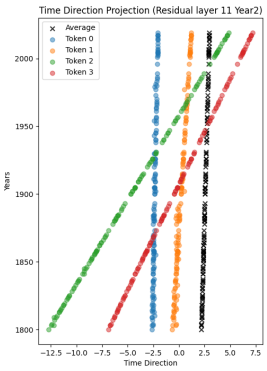

Time Direction Projection Years (Residual layer 11 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

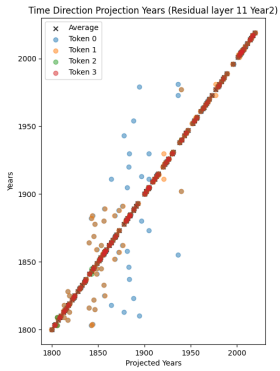

Time Direction Projection (Residual layer 11 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

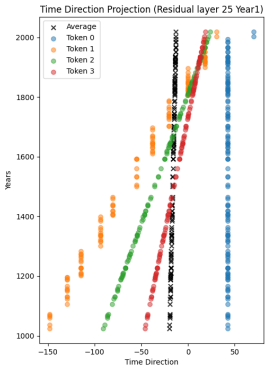

Time Direction Projection Years (Residual layer 11 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

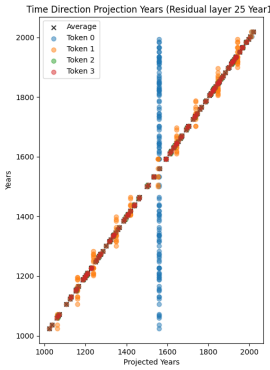

Time Direction Projection (Residual layer 11 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

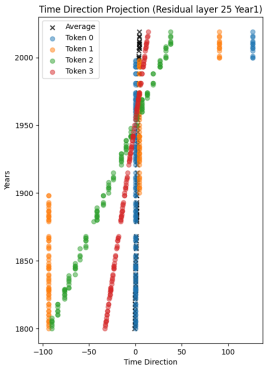

Time Direction Projection Years (Residual layer 11 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

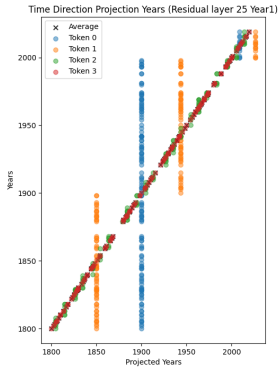

Time Direction Projection (Residual layer 11 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

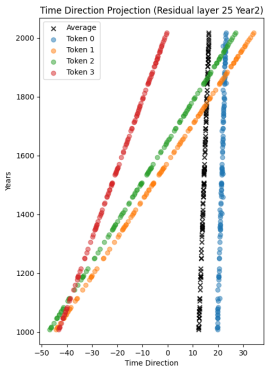

Time Direction Projection Years (Residual layer 11 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

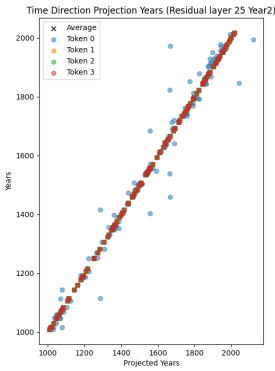

Time Direction Projection (Residual layer 25 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

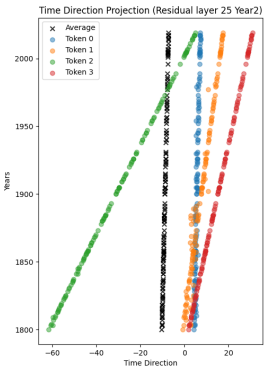

Time Direction Projection Years (Residual layer 25 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

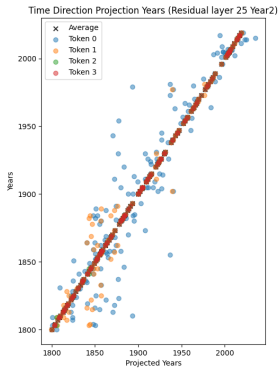

Time Direction Projection (Residual layer 25 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

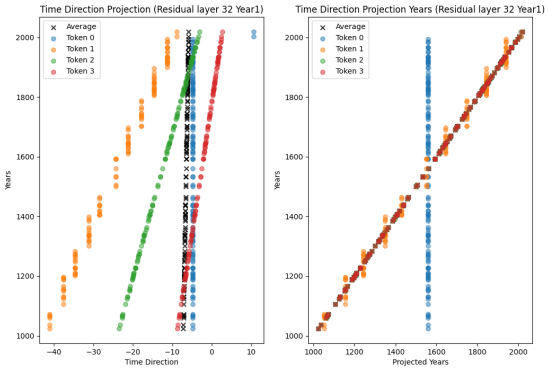

Time Direction Projection Years (Residual layer 25 Year1)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

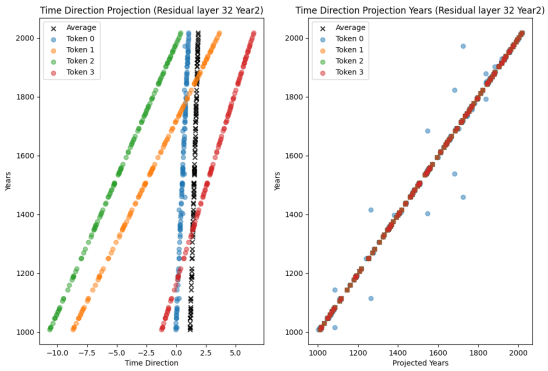

Time Direction Projection (Residual layer 25 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction

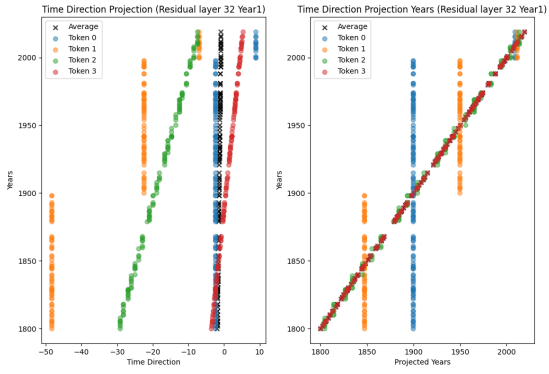

Time Direction Projection Years (Residual layer 25 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years

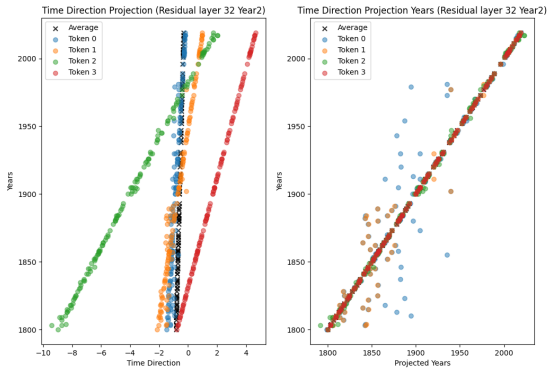

Time Direction Projection (Residual layer 25 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Time Direction


Time Direction Projection Years (Residual layer 25 Year2)
Average
Token 0
Token 1
Token 2
Token 3
Years
Projected Years









3) Conclusion

1. It fits better for year 2 (the one occurs later in the sentence), maybe it's because it is closer the answer position.
2. The deeper the layers, the higher the R^2, and it reaches 1 in first few layers (Layer 0-4).
3. The deeper the layers, the better `lr` fits except for token at first digit.
4. *Ablation*: If the range for the second digit scale up, it fits better in later layers than the last two digits, the R^2 is higher at the same time. Therefore, the emergence of this linear structure is correlative to the frequency and diversity of numbers appearing at every digit.

Comparison Circuit

In this experiment, I aim to identify the comparison circuit in the LLM that captures relations digit-wise in the years.

1) Methodology

1. With transformer-lens API, the model is run with cache. For each sample, the attention patterns, head outputs, and post-residual activations are stored and analysed at every layer, every head, and every digit level.
2. Three metrics are observed:
    1. `score_b2a`: Cross-attention score between same digits from a pair of years. `score_b2a = Attn(year₂_digit → year₁_digit)`. It measures how much the head explicitly looks from the second year's digit to the matching digit of the first year.

        ```
        def get_attn_score(cache, pos1, pos2, layer_id, head_id):
            pattern = cache[f"blocks.{layer_id}.attn.hook_pattern"][0, head_id]  # [Q, K]
            if pos1 > pos2:
                pos1, pos2 = pos2, pos1
            # score_a2b = pattern[pos1, pos2].item() # the previous a can't look at the latter b for causal modelling.
            score_b2a = pattern[pos2, pos1].item()
            return score_b2a
        ```

    2. `qk_diff`: It computes the cosine similarity between the QK scores of two digit positions from different years, with one negated. A high similarity implies that the attention pattern from one digit is roughly the inverse of the other, suggesting the head may be comparing the two digits via subtractive mechanisms.

        ```
        def get_qk_diff(cache, pos1, pos2, layer_id, head_id):
            q = cache[f"blocks.{layer_id}.attn.hook_q"][0, :, head_id]
            k = cache[f"blocks.{layer_id}.attn.hook_k"][0, :, head_id]

            qk = q @ k.T

            logits_a = qk[pos1]
            logits_b = qk[pos2]

            sim = F.cosine_similarity(logits_a, -logits_b, dim=0).item()

            return sim
        ```

    3. `delta`: Using the TransformerLens API, the model is run with a hook that swaps the activations at specific digit token positions between two years, in order to observe the causal impact of the heads.

        Specifically, `delta` is computed as:

```
def get_metric(logits):
    last_logits = logits[:, -1, :]
    t = model.to_single_token(true_label)
    f = model.to_single_token(false_label)
    return last_logits[:, t] - last_logits[:, f]
baseline_score = get_metric(baseline_logits)
patched_score = get_metric(patched_logits)
delta = baseline_score - patched_score
```

Relative codes are implemented in `comepare_head.py`.

2) Results

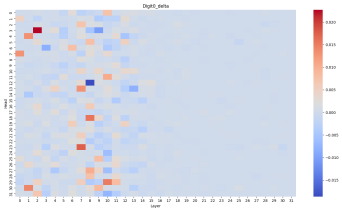
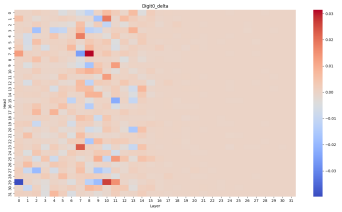
Causality Analysis

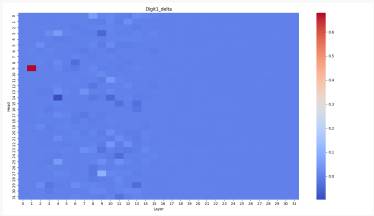
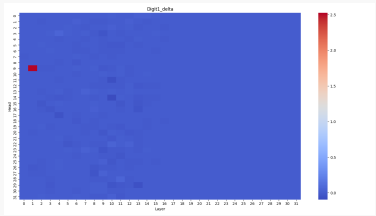
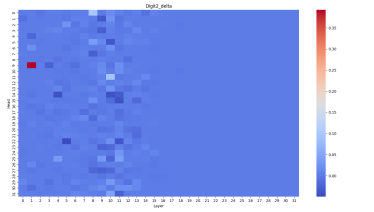
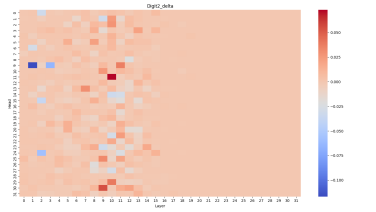
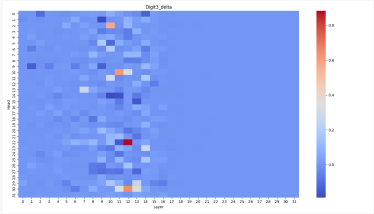
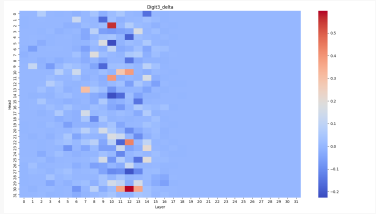
The causal influence of individual digit positions is captured by the difference `delta` in output probabilities of true labels between the original logits and those obtained by patching the corresponding digits in both years with zero.

To observe the importance of first digit, I constructed an ablation dataset by randomly sampling data points that differ in the first digit, since such changes are relatively infrequent in most cases. The size of datasets are both set to 10 samples here.

The layer- and head-wise heat maps of delta scores for each digit are shown below.

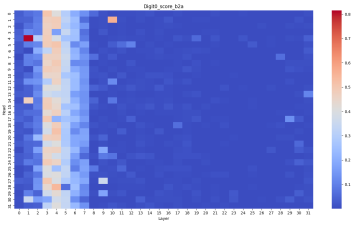
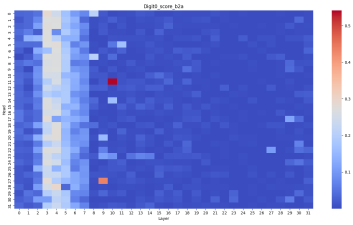
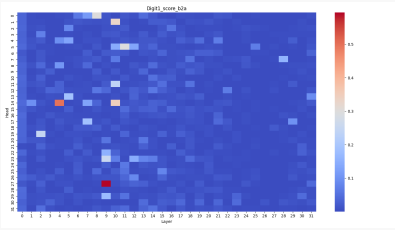
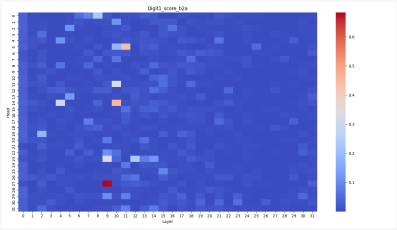
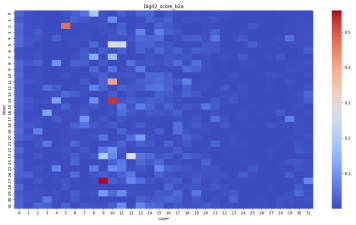
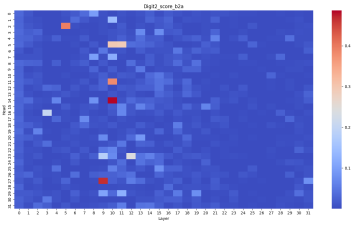
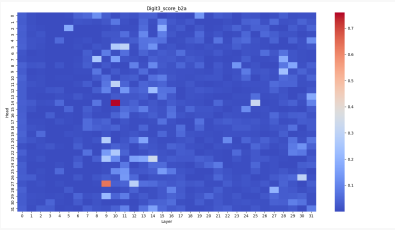
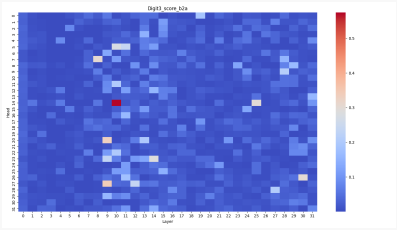
Note: Digit 0 refers to the thousands digit, while digit 3 refers to the ones digit.

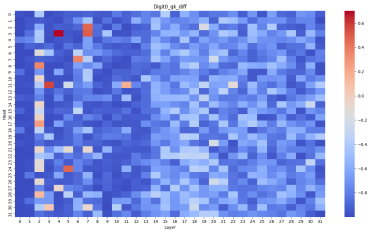
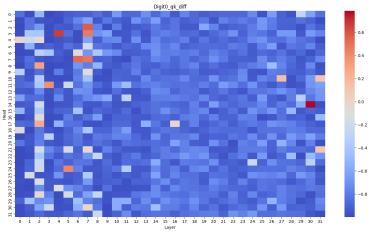
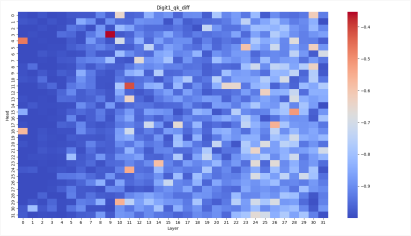
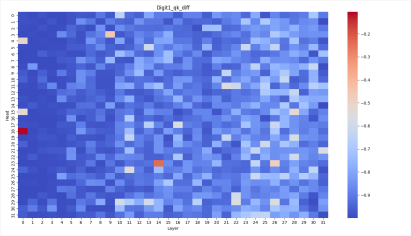
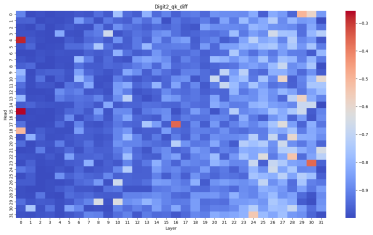
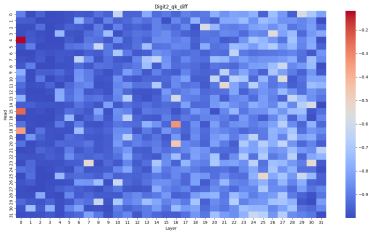
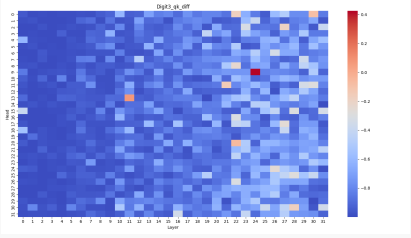
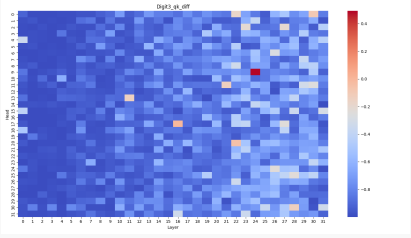
Delta	Random samples (only 2/10 samples different in digit 0)	Samples that all differ in digit 0	Findings
Digit 0			1. When using samples that all differ in the first digit, the overall `delta` scores of most heads for digit 0 increase noticeably. Causally significant heads are primarily concentrated in layers 0–14. 2. Key heads consistently identified in both settings include: (10, 29), (7, 23); 3. In contrast, heads such as (1, 30) and (2, 3) show divergent patterns between the two sample sets. This may indicate their involvement in

			modeling transiftions of the first digit in year representations.
Digit 1			Key head: (1, 9)
Digit 2			1. When using samples that all differ in the first digit, the overall `delta` scores of most heads for digit 2 decrease, noticing the ranges shown by cbar. Causally significant heads are primarily concentrated in layers 0–14. 2. One key head may be (1, 9), but it shows contrary pattern for two settings.
Digit 3			Key heads primarily emerge in layers 10-14, including (12, 22), (12, 30), (10, 2).

Attention Analysis

I aim to find out if there are compare heads for different digits when completing the task of understanding the temporal relation of years. First of all, I look into the attention scores `score_b2a` from the second year's digit to the corresponding digit of the first year. Moreover, I compute the QK difference `qk_diff` to testify if there are comparison behaviours by subtraction operations.

Score_b2a	Random samples	Samples that differ in digit 0	Findings
Digit 0			1. Layers 3–4 primarily capture relations between the first digits, indicating that the model attends to the most decisive digits early in the processing. 2. Key heads: (10, 11), (9, 27), (1, 4), (1, 14)
Digit 1			Key head: (9, 27), (10,14), (10, 11), (11, 5)
Digit 2			Key head: (9, 27), (10, 13), (10, 11), (4, 14)
Digit 3			Key head: (9, 27), (10, 14)

qk_diff	Random samples	Samples that differ in digit 0	Findings
Digit 0			1. Layers 2-8 have comparison behaviours based on subtraction operations. 2. Key heads: (4, 3), (3, 11), (5, 24), (7, 2), (6, 7) 3. For samples that differ in digit 0, there seems to be a "checking" stage at the last few deep layers, including heads (30, 24), (31, 11), (31, 21), (27, 11).
Digit 1			Key heads: (9, 3), (0, 4), (0, 18), (0, 15)
Digit 2			Key heads: (0, 4), (0, 15), (0, 18), (16, 17)
Digit 3			Key heads: (24, 9), (11, 13)

3) Conclusion

In conclusion, the model seems to use compare heads to capture the relations between the first digit in the early stage. For the lower digits (hundreds, tens, ones), there is no clear layer that focuses on comparing same digits. Instead, a few mid-layer heads like (9,27), (10,13), and (10,14) consistently contribute. This suggests the model compares the first digit (thousands) early and explicitly via compare heads, while

encoding the other digits more diffusely—possibly linearly. This matches the probing results, where lower digits show linear structure in early layers, while the first digit does not.

Additionally, the `qk_diff` analysis also shows strong difference-based comparison only for the first digit, mostly in layers 2-8, which is mostly absent for the other digits.

However, the attention analysis results do not fully align with the causality analysis, suggesting that the decision circuit is long and complex. For example, compare heads may compute differences early, but the final decision could depend on later heads that gather and process this information through the residual stream.

Answer Circuit

For this experiment, the goal is to find out the answer circuit in the model, using methods of activation patching.

1) Methodology

1. Using the transformer-lens API, the model is run on a set of yes/no classification prompts. The attention head outputs are cached.
2. Focusing on the answer token position (`answer_pos = tokens.shape[-1] - 1`), for every layer and attention head, three metrics are computed:
    1. `delta`: It measures the drop in logit margin between the true and false labels when the token position of true label is patched to zero. A higher drop indicates greater causal importance, similar to how `delta` is used in identifying comparison circuits.
    2. `cos_true`: It computes the cosine similarity between a head's output at the answer token and the unembedding vector of the true label. Higher values suggest the head helps copy and write the correct answer to the final decision.

        ```
        head_out = base_cache[f"blocks.{layer_id}.attn.hook_result"][0, answer_pos,
        head_id]
        true_label_dir  = model.W_U[:, model.to_single_token(true_label)]
        cos_true  = F.cosine_similarity(head_out, true_label_dir, dim=0).item()
        ```

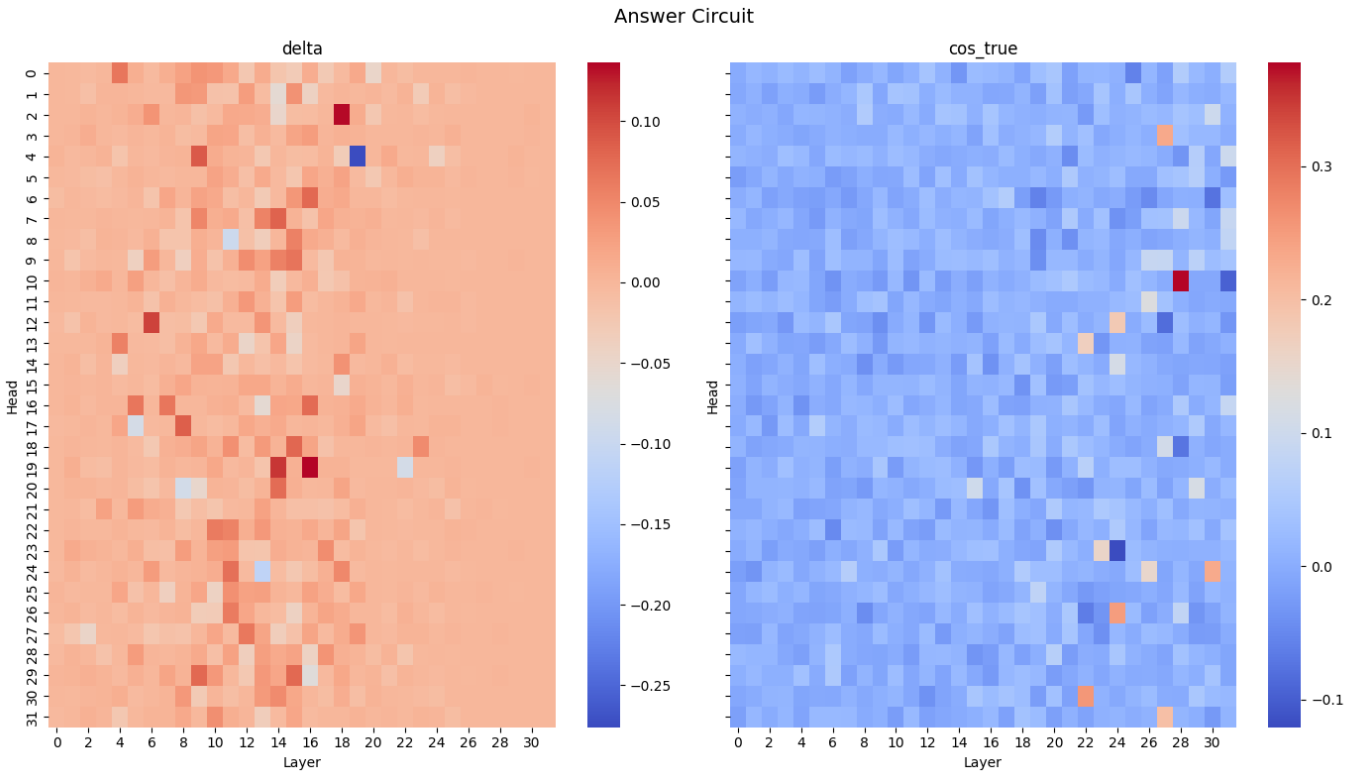
    3. `Top5 attended tokens frequency`: It counts the frequency of top5 tokens that receive the highest attention from the answer token position, showing which parts of the input the model focuses on most when generating the answer.

        ```
        attn = base_cache[f"blocks.{layer_id}.attn.hook_pattern"][0, head_id, answer_pos]
        top_idx = attn.topk(5).indices.tolist()
        top_tokens = [model.to_string(int(tokens[0, i])) for i in top_idx]
        ```

Relative codes are implemented in `answer_head.py`.

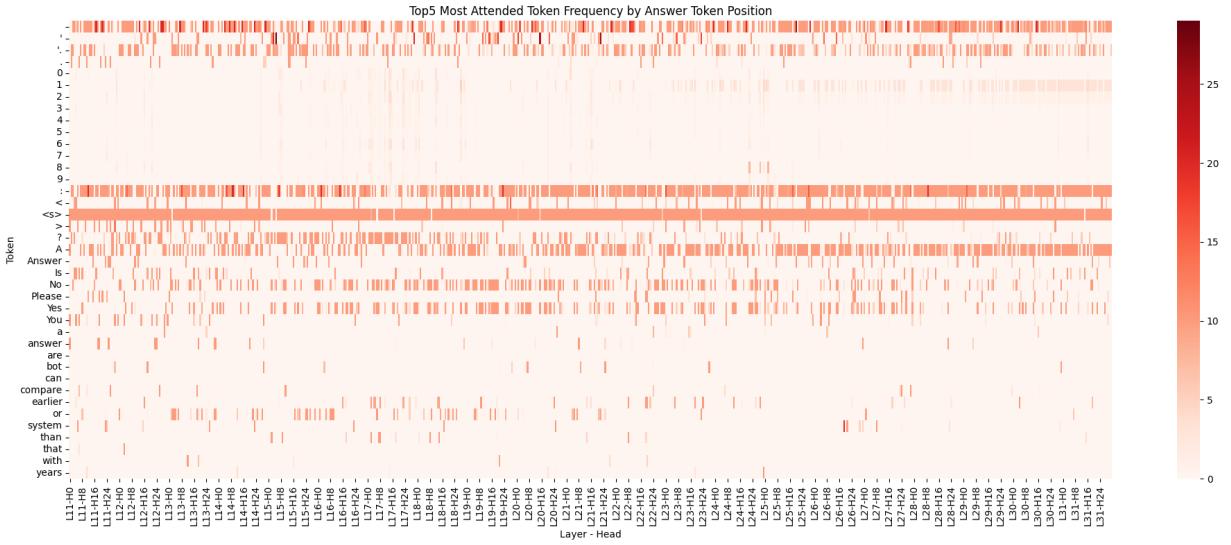
2) Results

Causality Analysis



- Key heads that copy the true label token: (28, 10), (22, 30), (24, 26), (27, 3), (30, 24)
- Key heads that has significant causal importance: (16, 19), (18, 2), (14, 19), (6, 12), (9, 4)

Top Attended Tokens



3) Conclusion

The heatmap reveals that the answer token frequently attends to <s> (the EOS token), and key semantic words like “Answer”, “Yes”, and “No”, especially in mid-to-late layers. Notably, several heads with high alignment to the true label token—such as (22, 30), and (24, 26)—also consistently direct attention to these answer-related tokens, indicating their role in writing the final output.

4 Conclusion & Future Work

The current analysis suggests that the model solves the “which year is later?” task using a multi-stage pipeline:

In the embedding and early residual layers (0–4), the model learns nearly perfect linear representations for the hundreds, tens, and ones digits. However, the thousands digit remains nonlinear at this stage.

To handle the first digit, the model uses a distinct comparison circuit. Layers 3–4 primarily capture attention relationships between the most decisive first digits early in the processing. Correspondingly, QK-difference analysis supports the emergence of strong difference-based comparisons focusing exclusively on the first digit, particularly in layers 2–8. This may indicate that the model understands the first digit using comparison head mechanisms rather than the linear encoding for the lower digits, as shown in the probing experiments.

The comparison of the lower digits is more distributed. A small cluster of heads in the mid-layers, especially heads (9, 27) and (10, 11) along with their neighbor, contributes significantly to this process, though without being concentrated in any single layer.

Finally, in the later stages (layers 22-32), dedicated “copy heads” such as (22, 30) and (30, 24) focus on output tokens like “Answer,” “Yes,” “No,” or the end-of-sequence symbol `<s>`. These heads directly copy the appropriate token into the unembedding layer, finalizing the decision-making process.

However, some pieces of the jigsaw are still missing. Specifically, the exact residual pathways that carry the thousands-digit subtraction signal from the early comparison heads to the mid-layer integrators have not yet been traced. Additionally, the role and activation patterns of the relevant MLP neurons remain unclear. Furthermore, most heads with high attention scores or strong QK-difference signals do not consistently align with high causality scores, suggesting that the link between the “comparison circuit” and the final “decisive circuit” is still incomplete.

To close this gap, further work is needed. In particular, more fine-grained ablations would be usefu, such as prompts where only specific digits are altered, neuron-level analyses to probe the internal logic of the computation, and more diverse activation patching strategies to map the flow of information throughout the model. These approaches could help clarify how digit-level comparisons are integrated and ultimately translated into the model’s final decision.

Reference

- Rai, D., Zhou, Y., Feng, S., Saparov, A., & Yao, Z. (2024). *A Practical Review of Mechanistic Interpretability for Transformer-Based Language Models*. https://arxiv.org/html/2407.02646v1#S4
- Gurnee, W., & Tegmark, M. (2023). *Language Models Represent Space and Time*. Massachusetts Institute of Technology. arXiv:2301.05217