Skoltech

MASTER'S THESIS

Analyzing Transformer Language Models with Interpretative Techniques

Master's Educational Program: Data Science

Student: _____ Pavel Shtykov
signature

Research Advisor: _____ Serguei Barannikov
signature
PhD, Leading research scientist

Co-Advisor _____ Alexey Naumov
signature
Doctor of Sciences, Professor

Moscow 2025

Skoltech

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Анализ трансформерных языковых моделей с помощью интерпретационных методов

Магистерская образовательная программа: Науки о данных

Студент: _____ Павел Штыков
подпись

Научный руководитель: _____ Сергей Баранников
подпись
к.м.н., ведущий научный сотрудник

Со-руководитель _____ Алексей Наумов
подпись
д.ф.-м.н., профессор

Москва 2025

Analyzing Transformer Language Models with Interpretative Techniques

Pavel Shtykov

Submitted to the Skolkovo Institute of Science and Technology on June 10, 2025

ABSTRACT

This thesis explores a potential interpretative framework for transformer language models based on movement through embedding space. Current approaches often separate token embeddings from hidden states, potentially limiting understanding of these models.

The proposed framework considers hidden states as points within the embedding space, possibly forming a path from token to token. This perspective may offer an alternative view where each layer contributes to a trajectory culminating in next-token prediction.

Experiments with Llama models and custom-trained models suggest that weight-tied models show alignment between similarity metrics and model logits. Analysis indicates intermediate representations seem to diverge from the token embedding manifold in deeper layers. A regularization approach appears to constrain hidden states without significantly impacting performance.

These preliminary findings may support the framework's potential, though questions remain. The ability to train models with representation constraints without apparent performance degradation might suggest directions for more interpretable language models.

Keywords: Transformer Language Models, Embedding Space, Model Interpretability, Hidden State Representation, Token Embeddings

Research advisor:
Name: Serguei Barannikov
Degree, title: PhD, Leading research scientist

Co-advisor:
Name: Alexey Naumov
Degree, title: Doctor of Sciences, Professor

Contents

Chapter 1
Introduction

Relevance. Transformer-based language models have revolutionized natural language processing, yet our understanding of their internal representations remains limited. Current approaches often create an artificial separation between token embeddings and hidden states, hindering interpretability. As these models become increasingly integrated into critical systems, developing better interpretative frameworks is essential for ensuring reliability, enabling systematic debugging, and driving architectural innovations.

Main purpose of the research. This research aims to develop and validate a novel interpretative framework for transformer language models based on the concept of movement through embedding space. The central hypothesis posits that transformer operations can be understood as a trajectory from one token to the next, with each layer contributing to this path. Through analysis of pre-trained models and experiments with models trained under specific constraints, this work investigates whether this embedding space movement paradigm offers a valid perspective for understanding transformer behavior.

Scientific novelty. This thesis explores potential contributions to transformer interpretation: (1) a perspective that considers both token embeddings and hidden states as points within the same embedding space; (2) methods to examine this interpretative framework; (3) an experimental regularization approach that attempts to constrain intermediate hidden states within the token embedding manifold; and (4) preliminary evidence from both pre-trained models and models with alternative architectural configurations. While these contributions suggest possible directions for transformer interpretation, they represent initial steps that require further validation and refinement.

Statements for defense. The following statements are presented for defense:

1. The proposed interpretative framework of movement through embedding space offers a valid perspective for understanding transformer language models.
2. In standard pre-trained transformers, intermediate hidden states progressively diverge from the token embedding manifold as layer depth increases.
3. Intermediate hidden states can be constrained to remain within the vicinity of the token embedding cloud through regularization without significant performance degradation.
4. Different language modeling head architectures, including those explicitly based on similarity metrics, can achieve comparable performance.

Chapter 2
Author contribution

The author of this thesis has made several contributions to the research presented:

The conceptual framework of viewing transformer operations as movement through embedding space was developed by the author, building upon existing literature on transformer architecture and interpretation. This perspective emerged from critical analysis of current approaches that separate token embeddings from hidden states.

The author designed and implemented the analytical methods used to evaluate this interpretative framework, including the similarity metrics between final hidden states and token embeddings, and the visualization techniques for tracking hidden state norms across layers.

The experimental work, including the analysis of pre-trained models and the training of models with specific architectural variations, was conducted by the author. This involved implementing the proposed regularization approach that constrains intermediate hidden states to remain within the token embedding manifold.

Chapter 3
Literature review

3.1 Transformer Models and Interpretability

Transformer-based language models have revolutionized natural language processing [19], yet they largely remain ”black boxes” with limited interpretability. The field of transformer interpretability has grown significantly, with approaches like VisBERT by Aken et al. [18] providing visualization techniques for hidden states that offer insights into the model's internal processing.

3.2 Geometric Interpretations of Transformer Representations

A promising direction involves analyzing transformers through geometry in embedding space. Dar et al. [3] present a framework where transformer parameters are interpreted by projecting them into embedding space, demonstrating that both pretrained and fine-tuned models can be understood through this lens.

Singh [14] extends this by framing transformer dynamics as movement through embedding space, establishing a theory where intelligent behaviors map to paths in embedding space and context vectors are composed by aggregating token features. This aligns with our thesis's focus on interpreting transformers as inertial movement in embedding space.

The geometric properties of transformer hidden representations have also been explored by Valeriani et al. [17], providing insights into information encoding across transformer layers.

3.3 Alternative Formulations of Language Modeling

Traditional transformers use a linear language modeling head, but alternative formulations enhance interpretability and performance. Geva et al. [6] showed that feed-forward networks in transformers function as key-value memories related to embedding space, suggesting a connection between internal representations and vocabulary token embeddings.

The relationship between k-nearest neighbors and transformer attention has been explored by Haris [8], who provides a theoretical framework for k-NN attention using Gaussian sampling for efficient approximation. Kernelized transformer variants have been investigated by Chowdhury et al. [2] and Simpson et al. [13], exploring Gaussian kernels and their integration with transformer architectures.

3.4 Embedding Space Dynamics and Token Relationships

The relationship between transformer hidden states and token embeddings has been studied from various angles. Song and Zhong [16] investigate hidden geometry in transformers by disentangling position and context. The concept of distances between hidden states and token embeddings, central to our research, has been touched upon by Aken et al. [18] and Dar et al. [3], providing a

foundation for our investigation into the relationship between probability distributions and embedding distances.

3.5 Research Gaps and Opportunities

Despite advances in transformer interpretability, several key gaps remain:

Current approaches create an artificial separation between token embeddings and hidden states, hindering unified understanding of information flow through transformers. While geometric interpretations exist, the concept of viewing transformer operations as a continuous trajectory through embedding space remains underexplored.

The relationship between final hidden states and token embeddings has not been systematically investigated across different architectures, particularly regarding how proximity in embedding space correlates with token prediction probabilities in models with and without weight tying.

Additionally, methods to constrain intermediate hidden states within the token embedding manifold are lacking. These gaps present opportunities for developing a more intuitive framework for transformer interpretation by reconceptualizing operations as movement through embedding space.

Chapter 4
Problem Statement

4.1 Transformer Language Model Architecture

Decoder-only transformer language models form the core architecture behind modern large models such as GPT [**?**] and LLaMA [**?**]. Unlike the original encoder-decoder design [19], these models utilize only the decoder with causal (unidirectional) attention, making them particularly suitable for autoregressive language modeling tasks.

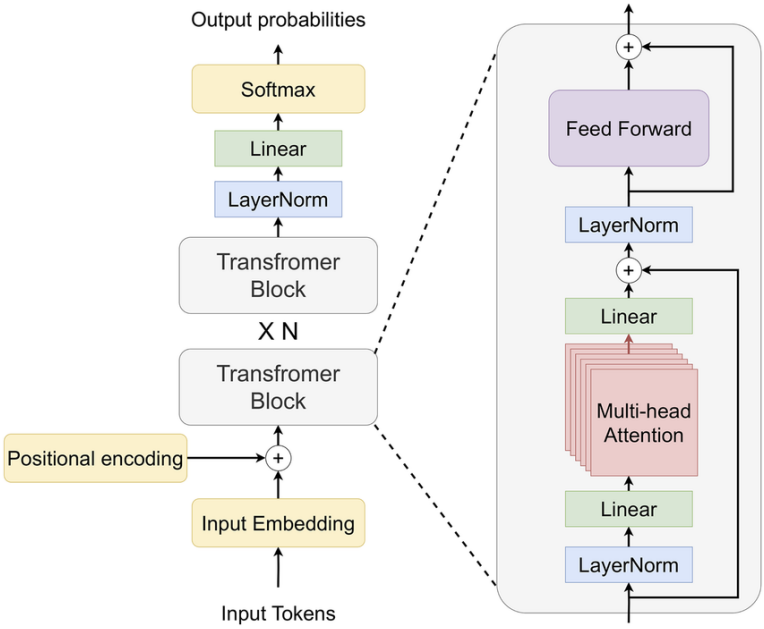

Figure 4.1: Illustration of the decoder-only transformer language model architecture (adapted from [1])

A decoder-only transformer begins with a vocabulary $\mathcal{V}$, where each token is mapped to a d-dimensional embedding via matrix $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$. For a sequence $\mathbf{t} = (t_1, \ldots, t_n)$, the embedding layer produces a sequence of vectors, which are then processed by L stacked decoder layers, yielding hidden states $\mathbf{h}_i^l \in \mathbb{R}^d$ for each token i at each layer l.

Each decoder layer consists of two main components: multi-head self-attention and a position-wise feed-forward network (FFN), both implemented with residual connections and layer normalization:

$$\mathbf{h}' = \mathbf{h} + \text{Attn}(\text{Norm}_1(\mathbf{h}), \mathbf{H}) \tag{4.1}$$

$$\mathbf{h}^{\text{out}} = \mathbf{h}' + \text{FFN}(\text{Norm}_2(\mathbf{h}')) \tag{4.2}$$

The self-attention mechanism enables each token to attend to itself and previous tokens (causal attention). For token i at layer l:

$$\text{Attn}(\mathbf{h}_i^{l-1}, \mathbf{H}^{l-1}) = \sum_{j=1}^{i} \alpha_{ij} \mathbf{W}_V \mathbf{h}_j^{l-1} \tag{4.3}$$

where $\mathbf{H}^{l-1}$ represents the set of all hidden states at layer $l-1$, i.e., $\mathbf{H}^{l-1} = \{\mathbf{h}_1^{l-1}, \mathbf{h}_2^{l-1}, \ldots, \mathbf{h}_i^{l-1}\}$ for causal attention, and α_{ij} are attention weights computed as:

$$\alpha_{ij} = \frac{\exp((\mathbf{W}_Q \mathbf{h}_i^{l-1})^T (\mathbf{W}_K \mathbf{h}_j^{l-1})/\sqrt{d_k})}{\sum_{j'=1}^{i} \exp((\mathbf{W}_Q \mathbf{h}_i^{l-1})^T (\mathbf{W}_K \mathbf{h}_{j'}^{l-1})/\sqrt{d_k})} \tag{4.4}$$

Here, $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ are learned parameter matrices, and d_k is the key dimension.

The FFN is applied to each position independently:

$$\text{FFN}(\mathbf{h}_i^{l-1}) = \mathbf{W}_2 \max(0, \mathbf{W}_1 \mathbf{h}_i^{l-1} + \mathbf{b}_1) + \mathbf{b}_2 \tag{4.5}$$

where $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, $\mathbf{b}_2$ are learned parameters.

The final hidden state $\mathbf{h}_i^L$ is projected to logits over the vocabulary by a linear language modeling head:

$$\text{LmHead}(\mathbf{h}_i^L) = \mathbf{W}\mathbf{h}_i^L + \mathbf{b} \tag{4.6}$$

where $\mathbf{W} \in \mathbb{R}^{|\mathcal{V}| \times d}$ and $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$. The probability of the next token is given by softmax:

$$P(t_{i+1} = v | t_1, \ldots, t_i) = \frac{\exp(z_{i,v})}{\sum_{v' \in \mathcal{V}} \exp(z_{i,v'})} \tag{4.7}$$

This architecture enables effective modeling of sequential data with both local and long-range dependencies, making decoder-only transformers highly effective for natural language processing tasks.

4.2 Embedding Space Interpretation Challenge

Current approaches to transformer interpretation often create an artificial separation between token embeddings $\mathbf{E}$ and intermediate hidden states $\mathbf{h}_i^l$. This separation introduces a conceptual disconnect in understanding how transformers process information. A more unified perspective is proposed: treating hidden states as points within the same embedding space. In this framework, the sequence of hidden states forms a continuous path from one token to the next.

The attention mechanism and feed-forward networks (described in the previous section) construct this path incrementally. Each layer transforms the representation, moving it closer to the next token prediction. The final output, calculated through Equations 4.6 and 4.7, represents the culmination of this trajectory through embedding space.

This unified perspective offers a more intuitive understanding of transformer operations and may lead to valuable insights for model design and optimization. The subsequent chapters explore this interpretation in detail, providing both theoretical analysis and empirical validation.

Chapter 5
Methodology

This chapter proposes an interpretative framework for transformer language models based on the concept of movement through embedding space.

The core insight is to view transformer operation as a trajectory from token t_i to token t_{i+1} in the embedding space. This process begins at the embedding of token t_i and proceeds through a series of transformations as L decoder layers are applied sequentially. Each decoder layer contributes an additive shift to the residual stream, creating a piecewise linear path through the embedding space, as illustrated in Figure 5.1.

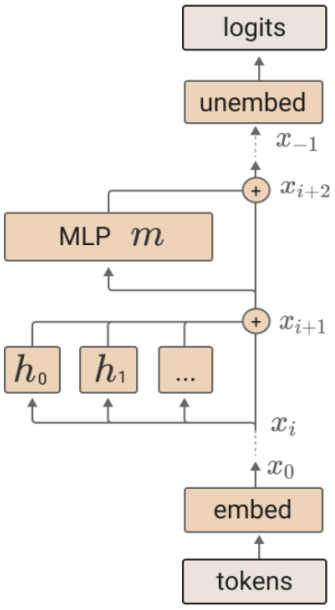

Figure 5.1: Illustration of the decoder-only transformer operation from the residual flow perspective (adapted from [4]). In this illustration, intermediate hidden states are denoted as $x_0, x_1, \ldots, x_{-1}$

After the final decoder layer produces the hidden state $\mathbf{h}_i^L$, the language modeling head generates a distribution over possible next tokens. The selection of token t_{i+1} (via sampling or greedy selection) creates a ”discontinuity” in the trajectory, as computation then continues from the embedding of this new token rather than from the final hidden state. Figure 5.2 provides a simplified visualization of this token-to-token movement pattern.

To verify the applicability of the proposed interpretation to real transformers, the following analytical tools are suggested.

5.1 Proximity Analysis Between Final Hidden States and Token Embeddings

A crucial aspect of the paradigm of movement in token embedding space is that the final hidden state should be close to the embedding of the next predicted token.

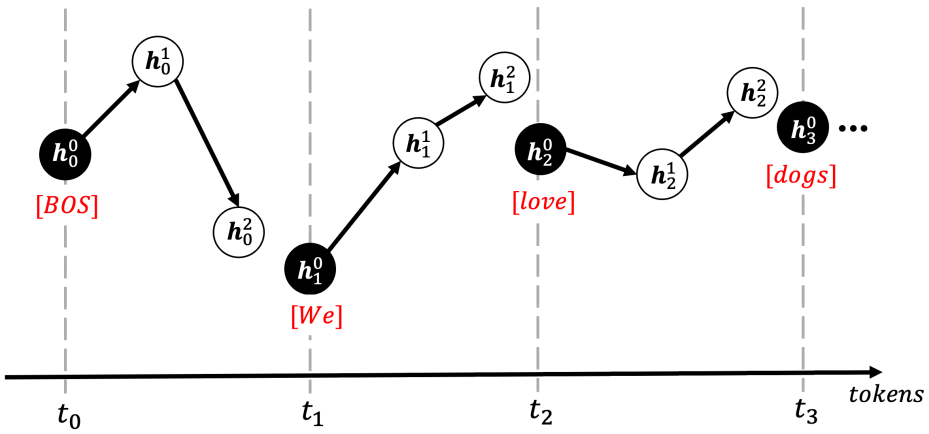

Figure 5.2: Simplified transformer operation diagram illustrating movement in embedding space. Filled circles represent token embeddings $t_0, t_1, \ldots$, while empty circles represent intermediate hidden states $\mathbf{h}_0^0, \mathbf{h}_0^1, \mathbf{h}_1^0, \mathbf{h}_1^1, \ldots$. The axis indicates the token sequence direction.

This behavior of the transformer can be visualized using the diagram in Figure 5.3, which complements Figure 5.2.

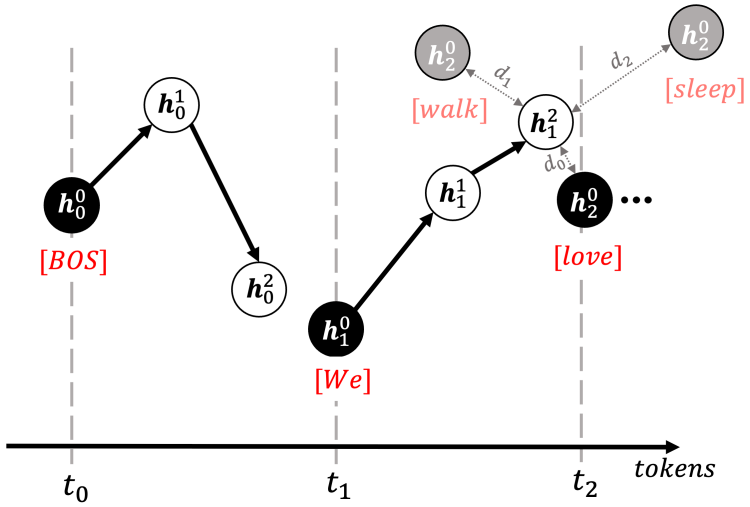

Figure 5.3: Illustration demonstrating the importance of proximity between the final hidden state $\mathbf{h}_i^L$ and the next token embedding t_{i+1} in the proposed transformer interpretation paradigm. Distances is $d_j = \rho(h_1^2, t_j)$ and $d_0 < d_1 < d_2$.

To systematically evaluate this hypothesis, two complementary approaches are proposed: analyzing pre-trained transformers and training new models with specific constraints.

Measuring Similarity Between Hidden States and Token Embeddings in Pre-trained Models

To verify whether pre-trained transformers conform to this requirement, it is proposed to measure how well the distances between final hidden states and token embeddings correlate with the token probabilities predicted by the language modeling head for these same hidden states.

For this purpose, it is necessary to introduce a similarity metric between the final hidden state and all token embeddings in the model's vocabulary.

The first metric is Euclidean similarity. To measure this, the Euclidean distances are first calculated and then inverted by negation or taking the reciprocal:

$$d_{\text{Euclid}}(\mathbf{h}_i^L, \mathbf{e}_j) = \|\mathbf{h}_i^L - \mathbf{e}_j\|_2 \tag{5.1}$$

$$\text{sim}_{\text{NegEuclid}}(\mathbf{h}_i^L, \mathbf{e}_j) = -d_{\text{Euclid}}(\mathbf{h}_i^L, \mathbf{e}_j) \tag{5.2}$$

$$\text{sim}_{\text{InvEuclid}}(\mathbf{h}_i^L, \mathbf{e}_j) = \frac{1}{d_{\text{Euclid}}(\mathbf{h}_i^L, \mathbf{e}_j)} \tag{5.3}$$

Additionally, two other similarity metrics are considered: cosine similarity and dot similarity:

$$\text{sim}_{\text{cosine}}(\mathbf{h}_i^L, \mathbf{e}_j) = \frac{\mathbf{h}_i^L \cdot \mathbf{e}_j}{\|\mathbf{h}_i^L\|_2 \|\mathbf{e}_j\|_2} \tag{5.4}$$

$$\text{sim}_{\text{dot}}(\mathbf{h}_i^L, \mathbf{e}_j) = \mathbf{h}_i^L \cdot \mathbf{e}_j = \sum_{k=1}^{d} h_{i,k}^L e_{j,k} \tag{5.5}$$

To evaluate the alignment between model predictions and embedding space proximity, the correlation between language model logits and the similarity metrics is measured. This comparison employs the Normalized Discounted Cumulative Gain (NDCG), a standard metric for ranking quality assessment:

$$\text{NDCG@k} = \frac{\text{DCG@k}}{\text{IDCG@k}} \tag{5.6}$$

where DCG@k (Discounted Cumulative Gain) is defined as:

$$\text{DCG@k} = \sum_{i=1}^{k} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)} \tag{5.7}$$

Here, rel_i is the relevance score of the item at position i, and IDCG@k is the DCG@k value for the ideal ranking. In this context, the relevance scores are derived from the token probabilities predicted by the language modeling head, and the comparison evaluates how well the similarity metrics align with these probabilities.

Constructing Similarity-Based Language Modeling Heads for New Models

It is possible to attempt training a transformer with the desired properties from scratch. To do this, it is necessary to explicitly construct logits based on the proximity of the final hidden state to token embeddings.

For the case of dot similarity, such a construction method is well-known and is called weight tying. By tying the weight matrix in the language modeling head to the embedding matrix of the model, dot similarity is explicitly calculated as logits at the output of the language modeling head layer:

$$\text{LmHead}_{tied}(\mathbf{h}_i^L) = \mathbf{E}\mathbf{h}_i^L + \mathbf{b} = [\text{sim}_{\text{dot}}(\mathbf{h}_i^L, \mathbf{e}_1) + b_1, \ldots, \text{sim}_{\text{dot}}(\mathbf{h}_i^L, \mathbf{e}_{|\mathcal{V}|}) + b_{|\mathcal{V}|}] \tag{5.8}$$

where $\mathbf{E} \in \mathbb{R}^{|\mathcal{V}| \times d}$ is the embedding matrix containing token embeddings $\mathbf{e}_j$, $\mathbf{h}_i^L \in \mathbb{R}^d$ is the final hidden state, and $\mathbf{b} \in \mathbb{R}^{|\mathcal{V}|}$ is the bias vector.

The Euclidean metric can also be approached in a similar manner. This involves simply outputting distances with a negative sign as logits at the layer output. Negation is chosen for simpler

computation during training (inverting the distance is a more expensive operation). Such a layer is termed KnnHead:

$$\text{KnnHead}(\mathbf{h}_i)_j = \text{sim}_{\text{NegEuclid}}(\mathbf{h}_i^L, \mathbf{e}_j) + b_j = -\|\mathbf{h}_i^L - \mathbf{e}_j\|_2^2 + b_j \tag{5.9}$$

Note that the squared Euclidean distance is used for computational efficiency, as it preserves the same ordering of distances while avoiding the square root operation.

Cosine similarity is not utilized as a head function when training transformers from scratch in this framework.

5.2 Analyzing Intermediate Hidden States in Embedding Space

Beyond the proximity of the final hidden state to the next token, the position of all intermediate hidden states is also crucial in this interpretative framework. It is logical to expect that intermediate hidden states should reside somewhere in the vicinity of the vocabulary token embeddings, since the transformer's movement in this paradigm occurs specifically from token to token.

For studying the position of intermediate hidden states in pre-trained transformers, the simplest approach is to examine their norms and visualize how these norms change depending on the layer number or the token position in the sentence. Heatmap visualizations of these norm patterns will be presented in the numerical experiments section.

When training a transformer from scratch, it is possible to explicitly penalize the model when certain intermediate hidden states deviate too far from the cloud of token embedding points. This can be implemented through a regularizer that linearly penalizes those intermediate hidden states whose norm exceeds the mean norm of token embeddings plus one and half standard deviations:

$$\mathcal{R}(\mathbf{h}_i^l) = \lambda \cdot \max(0, \|\mathbf{h}_i^l\|_2 - (\mu_{\mathbf{E}} + \frac{3}{2}\sigma_{\mathbf{E}})) \tag{5.10}$$

$$\mu_{\mathbf{E}} = \frac{1}{|\mathcal{V}|}\sum_{j=1}^{|\mathcal{V}|}\|\mathbf{e}_j\|_2 \tag{5.11}$$

$$\sigma_{\mathbf{E}} = \sqrt{\frac{1}{|\mathcal{V}|}\sum_{j=1}^{|\mathcal{V}|}(\|\mathbf{e}_j\|_2 - \mu_{\mathbf{E}})^2} \tag{5.12}$$

where $\mathcal{R}(\mathbf{h}_i^l)$ is the regularization penalty for hidden state $\mathbf{h}_i^l$, λ is the regularization coefficient, and $\|\mathbf{h}_i^l\|_2$ is the L2 norm of the hidden state. The total regularization term would be the average of penalties across all hidden states and tokens:

$$\mathcal{R}_{\text{total}} = \frac{1}{n \cdot L} \cdot \sum_{i=1}^{n}\sum_{l=1}^{L}\mathcal{R}(\mathbf{h}_i^l) \tag{5.13}$$

This regularization encourages intermediate hidden states to remain within a reasonable distance from the token embedding cloud, reinforcing the concept of movement through embedding space.

Chapter 6
Numerical Experiments

Table 6.1: Comparison of NDCG score across different model sizes and similarities

Model	NegEuclid	InvEuclid	Cosine	Dot
Llama 3.2 1B	0.921	0.912	0.952	1.000
Llama 3.2 3B	0.934	0.926	0.969	1.000
Llama 3.1 8B	0.120	0.113	0.093	0.091

The numerical experiments were conducted using Python and widely recognized deep learning libraries PyTorch[10] and Torch-Lightning[5]. For experiments with pre-trained models, Llama 3.2 1B, Llama 3.2 3B, and Llama 3.1 8B [7] were utilized. For training models from scratch, a small LLama-like model with 150M parameters was employed (model config described in 7). The SlimPajama dataset [15] served as the training data. Each model trained from scratch processed 1.5B tokens during training. After training, the models were evaluated on benchmarks including MMLU[9], AGIEval[20], and SQuAD[12]. The computational setup consisted of a single Nvidia 3090 GPU. The training and evaluation code is available on GitHub.

6.1 Correlation Between Final Hidden States and Token Embeddings

Table 6.2: Comparison of validation loss and score on benchmarks across 150M Llama-like model with different LmHead

LmHead Type	Loss ↓	MMLU (5-shot, acc)	AGIEval (4-shot, acc)	SQuAD (1-shot, em)
(base) LmHead	2.84	27.34	20.1	25.34
LmHead_{tied}	2.89	27.11	20.3	24.55
KnnHead	3.01	26.80	19.9	25.08

For pre-trained Llama family models, the NDCG score between model logits and various similarity metrics was measured. The detailed methodology is described in section 5.1. The metric was evaluated on a subsample of 1000 examples from the validation portion of the SlimPajama dataset. Averaging was performed across 100 tokens in the prompt and 20 generated tokens.

Based on the results in Table 6.1, the following conclusions can be drawn:

1. For Llama 3.2 1B and Llama 3.2 3B models, all similarity metrics rank tokens very similarly to how tokens are ranked by the standard model logits. This behavior is likely due to these models being trained by their authors with weight tying. The NDCG score of 1.0 for dot similarity confirms this.

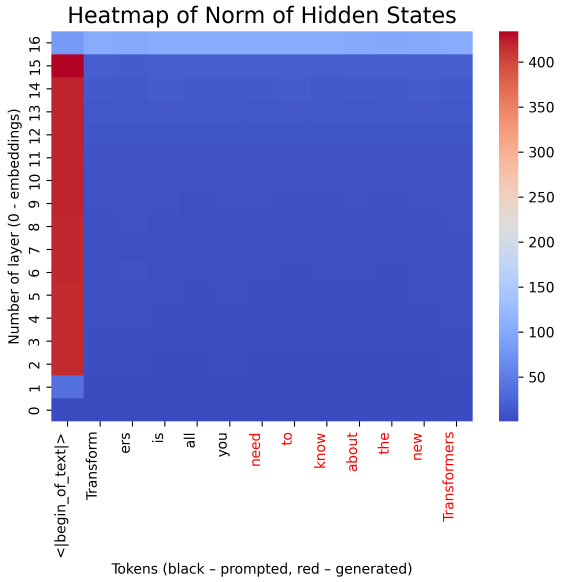

Figure 6.1: Heatmap of hidden state norms across layers and tokens for Llama 3.2 1B model. The x-axis represents tokens in the sequence (with generated tokens in red), and the y-axis represents model layers (with 0 being the embedding layer). Brighter colors indicate larger norms.

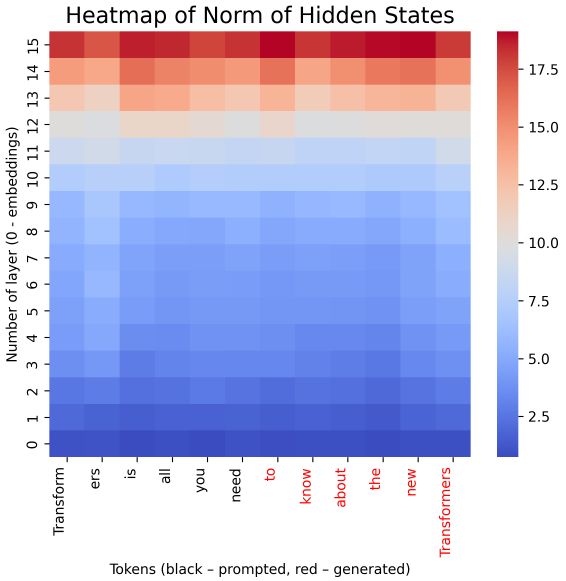

Figure 6.2: Heatmap of hidden state norms excluding the <BOS> token and final layer, providing a clearer view of the norm distribution in intermediate layers. Note the gradual increase in norm magnitude as layer depth increases.

2. The Llama 3.1 8B model was trained without weight tying. For this model, the NDCG score for all similarity metrics proved to be extremely low. This indicates that the model itself does not attempt to bring final hidden states closer to the most probable tokens according to any proximity metric.

Several models were also trained from scratch with different variants of LmHead and their quality was measured on the validation set and standard benchmarks. A detailed description of the language head variations used in the experiments can be found in section 5.1.

The experimental results presented in Table 6.2 reveal several significant findings:

1. The standard LmHead configuration (without weight tying) demonstrated marginally superior performance across evaluation metrics. Nevertheless, all architectural variants maintained functional integrity and exhibited comparable performance characteristics, suggesting robustness in the underlying approach.

2. All models converged to validation loss values consistent with expectations for models of this parameter scale (according to [11]). This convergence pattern confirms the viability of all three architectural approaches for language modeling tasks.

3. Performance on the AGIEval benchmark approached random baseline levels across all model variants. This observation indicates either inherent limitations in the model capacity relative to the benchmark complexity, or insufficient contextual information provided by the 4-shot prompting methodology.

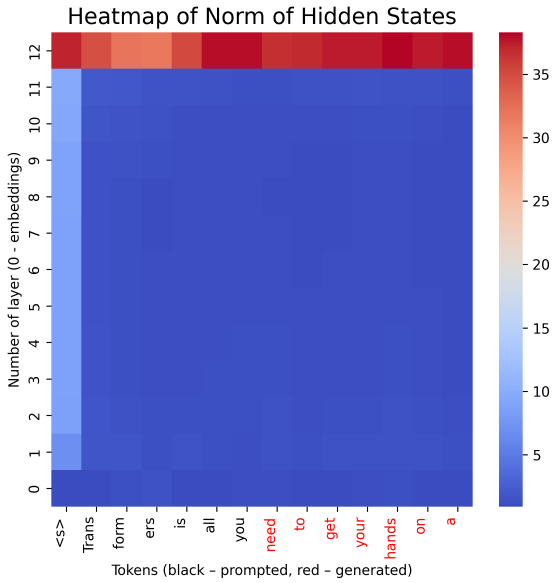

Figure 6.3: Heatmap of hidden state norms for the 150M parameter model trained with the regularizer.

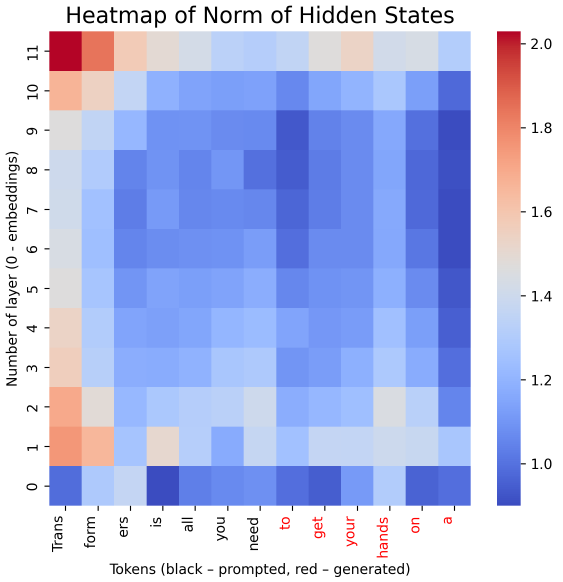

Figure 6.4: Heatmap of hidden state norms for the regularized model, excluding the <BOS> token and final layer. The regularization effect is visible in the more uniform norm distribution across layers compared to Figure 6.2. All intermediate hidden states maintain norms comparable to token embedding norms (layer 0)

6.2 Distribution of Intermediate Hidden States in Embedding Space

To investigate the relative positioning of intermediate hidden states and token embeddings, the norms of hidden states after each layer of the model were visualized. For this visualization, the pre-trained Llama 3.2 1B model was initially used.

In Figures 6.1 and 6.2, heatmaps for Llama 3.2 1B are presented. The model was given the prompt ”Transformers is all you” and generated 8 tokens in response (generated tokens are marked in red). There are same heatmaps for Llama 3.2 3B and Llama 3.1 8B models in appendix (7.1, 7.2, 7.3, 7.4).

Quantitative analysis of these heatmap visualizations reveals several noteworthy patterns:

1. The <BOS> token exhibits distinctive behavior, with its intermediate hidden state representations rapidly developing substantially larger norm magnitudes compared to other tokens. This phenomenon suggests a specialized role for the <BOS> token in establishing contextual foundations for subsequent processing.

2. Final layer hidden states demonstrate unexpectedly elevated norm values across all tokens, indicating a systematic transformation in the representation space immediately preceding token prediction.

3. A clear pattern emerges when isolating the intermediate layers and non-<BOS> tokens: hidden state norms progressively increase with network depth, while token embedding norms (at layer 0) remain consistently normalized around magnitude 1. Notably, this pattern manifests uniformly across both prompt and generated tokens. This systematic norm expansion

demonstrates that intermediate hidden states in the Llama 3.2 1B architecture progressively diverge from the token embedding manifold as information propagates through the network.

Additionally, a small Llama-like model with 150M parameters was trained with KnnHead and a regularizer $\mathcal{R}$ that penalizes intermediate hidden states for deviating too far from the token embedding cloud. The operating principle of this regularizer is described in detail in section 5.2. Chart with regularization magnitude over training steps attached in appendix 7.5.

Examination of heatmaps 6.3 and 6.4 provides compelling evidence for the efficacy of the regularization approach. The visualization confirms that intermediate hidden state representations maintain norm magnitudes comparable to token embeddings, successfully constraining these representations to remain within the token embedding manifold. This controlled norm behavior stands in marked contrast to the unregularized model. Notably, the regularization mechanism did not affect the distinctive norm patterns observed for the <BOS> token or final layer hidden states, suggesting these may be intrinsic properties necessary for model functionality rather than incidental characteristics.

It should be noted that the benchmark metrics and validation loss for the model trained with this regularizer did not change significantly. This can be seen in Table **??**.

Chapter 7
Discussion and Conclusion

This thesis explored a potential interpretative framework for transformer language models based on the concept of movement through embedding space. The research attempted to address a gap in current approaches to transformer interpretation, which typically separate token embeddings from hidden states.

The proposed framework considers hidden states as points within the embedding space, potentially forming a path from token to token. This perspective may offer an alternative understanding of transformer operations, where each layer could contribute to a trajectory through embedding space.

Experimental results suggest several possible insights:

- Models with weight tying appeared to show alignment between similarity-based rankings and model logits, though the relationship's nature requires further investigation.
- Different language modeling head architectures exhibited relatively similar performance, potentially indicating some degree of robustness.
- Hidden state norms in pre-trained models seemed to increase with layer depth, possibly indicating divergence from the token embedding space.
- A regularization approach appeared to constrain intermediate hidden states without major performance impacts, though its broader implications remain unclear.

These preliminary findings may support the potential of the proposed framework, though many questions remain unanswered. The ability to train models with representation constraints without apparent performance degradation might suggest directions for more interpretable models.

Future work could explore this approach with larger models, investigate regularization strategies, and examine the framework's applicability to various transformer architectures.

Bibliography

[1] Chalvatzaki, G., Younes, A., Nandha, D., Le, A., Ribeiro, L. F. R., and Gurevych, I. Learning to reason over scene graphs: A case study of finetuning gpt-2 into a robot language model for grounded task planning.

[2] Chowdhury, S. P., Solomou, A., Dubey, A., and Sachan, M. On learning the transformer kernel.

[3] Dar, G., Geva, M., Gupta, A., and Berant, J. Analyzing transformers in embedding space.

[4] Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread* (2021). https://transformer-circuits.pub/2021/framework/index.html.

[5] Falcon, W., and The PyTorch Lightning team. PyTorch Lightning, Mar. 2019.

[6] Geva, M., Schuster, R., Berant, J., and Levy, O. Transformer feed-forward layers are key-value memories.

[7] Grattafiori, A., Dubey, A., Jauhri, A., and et al., A. P. The llama 3 herd of models.

[8] Haris, T. $k$nn attention demystified: A theoretical exploration for scalable transformers.

[9] Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding, 2021.

[10] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. In *NIPS-W* (2017).

[11] Pearce, T., and Song, J. Reconciling kaplan and chinchilla scaling laws, 2024.

[12] Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. Squad: 100,000+ questions for machine comprehension of text, 2016.

[13] Simpson, F., Davies, I., Lalchand, V., Vullo, A., Durrande, N., and Rasmussen, C. Kernel identification through transformers.

[14] Singh, S. S. Analyzing transformer dynamics as movement through embedding space.

[15] Soboleva, D., Al-Khateeb, F., Myers, R., Steeves, J. R., Hestness, J., and Dey, N. SlimPajama: A 627B token cleaned and deduplicated version of RedPajama. https://www.cerebras.net/blog/slimpajama-a-627b-token-cleaned-and-deduplicated-version-of-redpajama, 2023.

[16] Song, J., and Zhong, Y. Uncovering hidden geometry in transformers via disentangling position and context.

[17] Valeriani, L., Doimo, D., Cuturello, F., Laio, A., Ansuini, A., and Cazzaniga, A. The geometry of hidden representations of large transformer models.

[18] van Aken, B., Winter, B., Löser, A., and Gers, F. A. Visbert: Hidden-state visualizations for transformers.

[19] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems 30* (2017).

[20] Zhong, W., Cui, R., Guo, Y., Liang, Y., Lu, S., Wang, Y., Saied, A., Chen, W., and Duan, N. Agieval: A human-centric benchmark for evaluating foundation models, 2023.

Appendix

Подробная архитектура 150M Llama-like модели

При обучении с нуля была использована следующая конфигурация Llama-like модели:

1. hidden_size = 768
2. intermediate_size = 3072
3. num_hidden_layers = 12
4. num_attention_heads = 12
5. num_key_value_heads = 12
6. head_dim = 64
7. rope_theta = 50000
8. max_seq_len = 1024
9. torch_dtype = bfloat16
10. attn_implementation = flash_attention_2

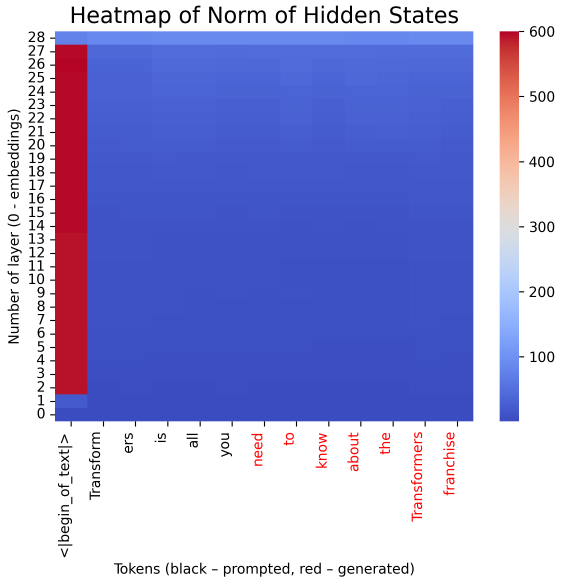

Figure 7.1: Heatmap of hidden state norms across layers and tokens for Llama 3.2 3B model. The x-axis represents tokens in the sequence (with generated tokens in red), and the y-axis represents model layers (with 0 being the embedding layer). Brighter colors indicate larger norms.

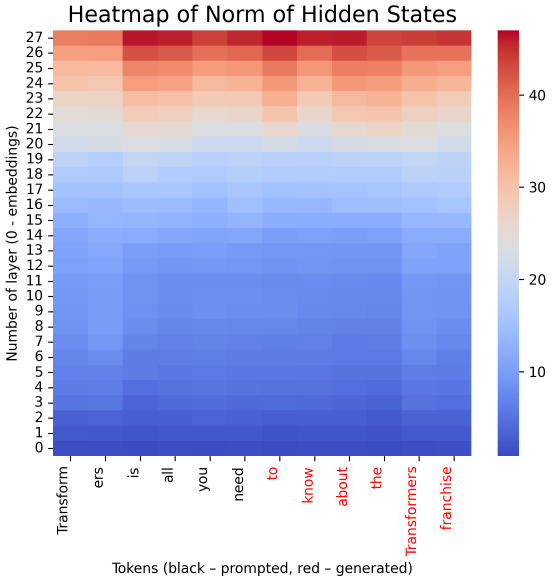

Figure 7.2: Heatmap of hidden state norms excluding the <BOS> token and final layer for Llama 3.2 3B model, providing a clearer view of the norm distribution in intermediate layers. Note the gradual increase in norm magnitude as layer depth increases.

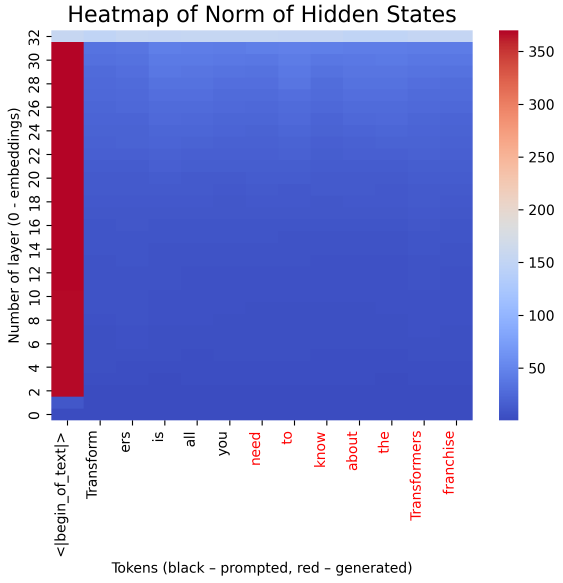

Figure 7.3: Heatmap of hidden state norms across layers and tokens for Llama 3.1 8B model. The x-axis represents tokens in the sequence (with generated tokens in red), and the y-axis represents model layers (with 0 being the embedding layer). Brighter colors indicate larger norms.

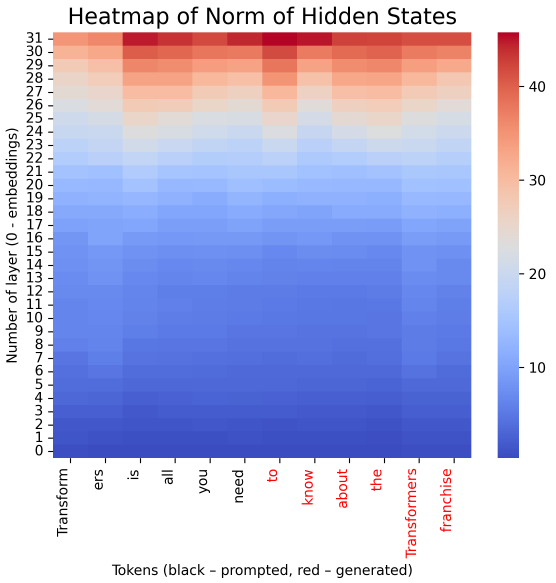

Figure 7.4: Heatmap of hidden state norms excluding the <BOS> token and final layer for Llama 3.1 8B model, providing a clearer view of the norm distribution in intermediate layers. Note the gradual increase in norm magnitude as layer depth increases.

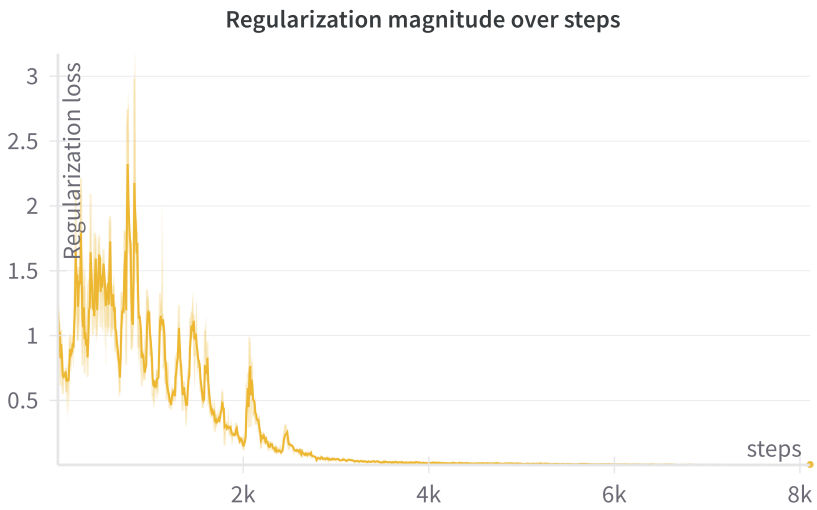

Figure 7.5: Regularization magnitude over steps during training. At the beginning of training, the regularizing penalty kick contributes significantly to the loss. However, as training progresses, its impact decreases and it becomes negligible.