How do Large Language Models Learn In-Context? Query and Key Matrices of In-Context Heads are Two Towers for Metric Learning

Anonymous ACL submission

Abstract

We explore the mechanism of in-context learning and propose a hypothesis using locate-and-project method. In shallow layers, the features of demonstrations are merged into their corresponding labels, and the features of the input text are aggregated into the last token. In deep layers, in-context heads make great contributions. In each in-context head, the value-output matrix extracts the labels' features. Query and key matrices compute the attention weights between the input text and each demonstration. The larger the attention weight is, the more label information is transferred into the last token for predicting the next word. Query and key matrices can be regarded as two towers for learning the similarity metric between the input text and each demonstration. Based on this hypothesis, we explain why imbalanced labels and demonstration order affect predictions. We conduct experiments on GPT2 large, Llama 7B, 13B and 30B. The results can support our analysis. Overall, our study provides a new method and a reasonable hypothesis for understanding the mechanism of in-context learning. Our code will be released on github.

1 Introduction

In-context learning (ICL) is an emergent ability (Wei et al., 2022a) of large language models (Brown et al., 2020; Ouyang et al., 2022; Touvron et al., 2023). By using a few demonstration-label pairs as prompts, ICL can perform well without updating parameters on many tasks, such as machine translation (Sia and Duh, 2023), complexity reasoning (Li et al., 2023a), compositional generalization (Zhou et al., 2022) and information extraction (He et al., 2023). However, the mechanism behind ICL is still a mystery (Zhao et al., 2023).

In this paper, we explore the mechanism of ICL on classification tasks with semantically-unrelated labels. We find that the mechanism behind ICL is computing the similarity metrics between the input text and each demonstration, then choosing the corresponding labels based on the similarity scores. Figure 1 shows the mechanism. In shallow layers, the features of demonstrations are aggregated into the corresponding labels (Wang et al., 2023). At the same time, the features of the input text are merged into the last token. In deep layers, features of demonstrations and labels are contained in labels' layer inputs, and features of the input text are in the last token's layer input.

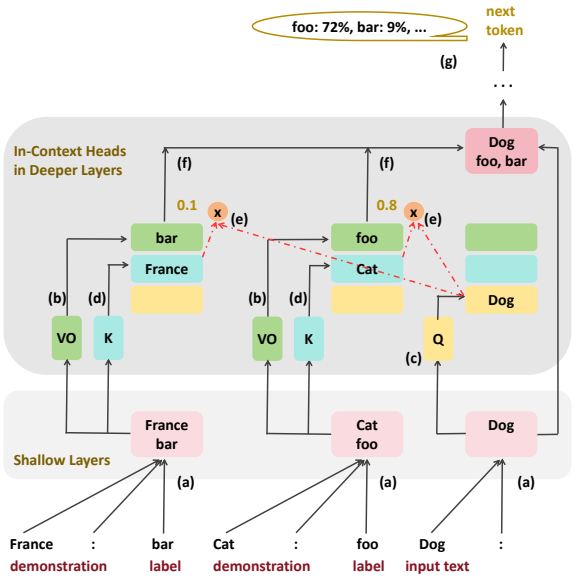


Figure 1: Mechanism of ICL. (a) In shallow layers, the features of demonstrations and the input text are merged into corresponding labels and the last token respectively. In deep layers' in-context heads, (b) the value-output matrix VO extracts the label information. (c) The query matrix Q and (d) the key matrix K compute the (e) attention weights between input text and each demonstration. (f) The attention weight decides how much label information is transferred into the last token. (g) Many in-context heads contribute to the information flow from labels to the last token, which predicts the next word.

In in-context heads, the value-output matrix extracts the labels' features ("foo", "bar") from the layer inputs on label positions. The query and key

matrices compute the attention weights (which can be regarded as similarity scores) between the input text and all demonstrations. Each label value is multiplied by the corresponding attention weight, which controls the label information flow. If the attention weight is large, much information of the corresponding label is transferred into the last token. In deep layers, many in-context heads contribute to the information flow from the labels to the last token, which finally predicts the next word.

1.1 Evidence Supporting the Hypothesis

We take an ICL sentence as an example, and propose a locate-and-project method to analyze the parameters in important layers and heads (Section 3). We find almost all the contributions are caused by deep attention layers (18-31 layers). In each layer, we calculate the contribution score of each head, and assume the most contributing heads are in-context heads (Section 3.1). In section 3.2, we analyze the queries, keys and values by projecting them in vocabulary space (Dar et al., 2022). We find the top words of label values are the labels themselves, and the label keys contain the information of corresponding demonstrations. The last token's query contain the features of the input text.

Moreover, we conduct experiments on word classification datasets and sentence classification datasets with semantically-unrelated labels on GPT2 large (Radford et al., 2019), Llama 7B, 13B and 30B (Touvron et al., 2023). When projecting into vocabulary, the rankings of label words in in-context heads are much smaller than those in random heads (Section 4.3). And the attention weights on true labels are much higher than those on false labels in in-context heads (Section 4.4). These results can support our hypothesis.

1.2 Explaining Phenomenons of ICL

There are several phenomenons of ICL that can be explained by our hypothesis. First, the model tends to predict the majority label in the prompt (Zhao et al., 2021). This phenomenon matches our hypothesis. Query and key matrices compute the attention weights between the input text and each demonstration, so the sum of one label's attention weights is larger when this label is related to more demonstrations. The experimental results in Section 5.1 can support this: when reducing the true label's frequency, the sum of attention weights on true labels will decrease.

Another phenomenon is that the demonstration order affects prediction much (Lu et al., 2021). Our assumption of this phenomenon is: the model is not trained only for ICL, thus the labels do not only aggregate the corresponding demonstration. A little adjacent tokens' features are merged into the labels. Therefore, when the true demonstrations/labels are near the input text, the last token extracts the features of input text and some true demonstrations, which can enhance the features. We conduct experiments to verify this in Section 5.2. When putting all the true demonstrations/labels near the input text, the sum of attention weights on true labels are larger than that when putting them far from the input text.

Overall, we develop a locate-and-project method for exploring the mechanism of ICL. We propose a reasonable hypothesis interpreting how ICL works, and this hypothesis can explain the phenomenons of ICL. Our proposed method and hypothesis are helpful for understanding ICL.

2 Background

2.1 ICL: Text Recognition and Text Learning

Language models perform ICL by prompting K demonstration-label pairs $(d_1, l_1, d_2, l_2, ..., d_K, l_K)$ before the input text X, in order to predict output Y. According to Pan et al. (2023), ICL can be disentangled into task recognition (TR) and task learning (TL). TR does not rely on the demonstration-label mappings because the roles of demonstrations and labels are helping the model know "what is the task". In this situation, the model have similar predictions when the mappings are wrong (Min et al., 2022), because the predictions are based on pre-trained priors. On the other hand, TL relies on the demonstration-label mappings because the semantic priors are removed. For example, in an ICL sentiment classification task, if the labels are "positive/negative", the task is TR. If the labels are "foo/bar", the task is TL because the labels are semantically-unrelated (Wei et al., 2023).

2.2 Label Words are Anchors

Wang et al. (2023) average all attention heads in GPT2-XL and calculate the saliency scores to extract the information flow among layers. They find the label words are anchors to merge the semantic information of corresponding demonstrations in shallow layers, and information is extracted from label words to the final prediction in deep layers.

Our study takes a step further on this work. We find the information flows from labels to the final prediction in deep layers are caused by in-context heads. Instead of saliency scores, we find the attention weights in in-context heads are interpretable, which can be regarded as the similarity metric between the input text and each demonstration. Moreover, we analyze the queries, keys and values in vocabulary space and find human-interpretable concepts. Overall, Wang et al. (2023) find "the information flow exists", while our work aims to answer "how this information flow happens".

2.3 Analyzing Parameters in Vocabulary Space

Many studies have found that the parameters in transformers are interpretable in vocabulary space (Elhage et al., 2021; Geva et al., 2022; Dar et al., 2022). The core idea is to compute the probability distribution of each vector on the unembedding matrix E. The final distribution D_f for predicting the next word is computed by the final vector f:

$$D_f = softmax(E\, f)$$

Similarly, the distribution D_v of other vectors v can be computed in vocabulary space:

$$D_v = softmax(E\, v)$$

If a word w ranks top in D_v, it indicates v is related to w. v can be parameters in different modules, including feed-forward network (FFN) value/key and attention value-output/query-key modules.

2.4 Induction Heads in Attention Layers

Olsson et al. (2022) find that induction heads in attention layers are helpful for copying words from the input sequence (e.g. [A][B]...[A] -> [B]). Our work is inspired by this study. In-context heads have similar characteristics with induction heads on value-output modules, which both transform the $0th$ layer input embeddings into the values. The difference is that induction heads' keys may only extract the previous token's information, while in-context heads should extract features from the whole demonstration. The investigation of the connection between in-context heads and induction heads is a topic deserving further exploration, which we defer to future research.

3 Locate-and-Project Method

In this section, we show how we propose the hypothesis by studying an ICL case **"love : bar like : bar eight : foo two : foo one :"** with the prediction **"foo"**. Our locate-and-project method is helpful and efficient for understanding ICL mechanisms. We conduct the experiments on GPT2 large, which has 36 layers and 20 heads per layer. There are too many layers and heads, so we propose a method to **locate** the important ones. Moreover, we **project** the queries, keys and values in these heads in vocabulary space to understand the mechanism.

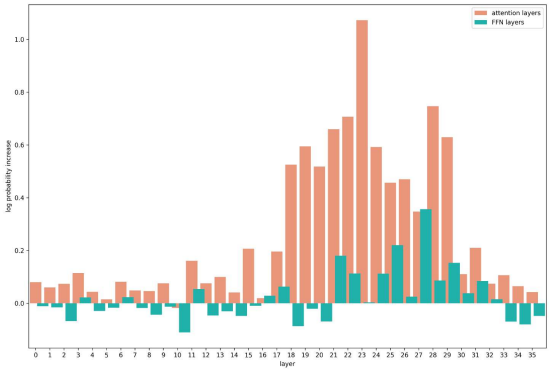


Figure 2: Log probability increase of each layer.

3.1 Locating Important Layers and Heads

Inspired by Yu and Ananiadou (2024), we locate helpful layers by computing each attention layer and FFN layer's log probability increase for the predicted word ("foo"). The results are shown in Figure 2. Almost all the improvements are caused by deep attention layers. Then we compute the log probability increase of each head and analyze the heads with large scores to see whether they have regular patterns.

3.2 Projecting Queries, Keys and Values in Vocabulary Space

After locating the important heads, we aim to analyze whether these heads have human-interpretable concepts. According to Wang et al. (2023), the labels and the last token play important roles in deep layers. So we project the labels' values/keys and the last token's query in vocabulary space following Dar et al. (2022). For label values, we multiply the value-output matrix and the layer inputs. For label keys, we look at their inputs before multiplying the key matrix. For last token's query, we compute its reverse projection into the keys' vector space by multiplying QK^T. In the example sentence, the positions are 2, 5, 8, 11 (labels), and 13 (last token), corresponding to "bar", "bar", "foo", "foo" and ":".

position	top words in vocabulary space
2-value	**BAR**, Barron, Barrett, Band, Bray, **Bars**, Baron, **Bar**, Bay, Boyd
5-value	**BAR**, Barron, Barrett, Baron, **Bar**, Band, Barbie, Barbar, Bard
8-value	**foo**, **Foo**, FO, fo, Foley, Fresno, FDR, fascists
11-value	**foo**, **Foo**, fo, FO, fascists, FDR, Foley, Goo, fascists
2-key	**kisses**, **goddess**, **love**, **charms**, idol, stress, nobles, **happiness**
5-key	style, oriented, +++, **like**, indo, height, Lover, xual, dont, foo
8-key	foo, mc, blah, happ, avg, french, omega, prod, english, google, height, neigh
11-key	foo, mc, infinity, omega, **three**, **two**, repeat, **twelve**, 666, **Three**, **thirds**, **five**, **sixteen**
13-query	**first**, end, only, no, all, given, person, certain, call, same, short, long, **1**, **one**, value

Table 1: Top words of labels and last token in layer 22, head 0.

We take layer 22, head 0 as an example. The top vocabulary tokens of labels' values/keys and the last token's query are shown in Table 1. For clarity, we remove the stop words.

The results are human-interpretable. Label values' top words have related concepts with the labels ("bar" and "foo"). Label keys' top words on position 2, 5, and 11 are related to their demonstrations ("love", "like" and "two"). Last token query's top words are related to the input text ("one").

Based on these interpretable results, we hypothesize that the features of demonstrations are extracted into the corresponding label keys, and the features of labels are extracted into the label values. The features of the input text are compressed in the last token's query. Moreover, we hypothesize the information flow of the label information are controlled by the attention weights between last token query and label keys. Yu and Ananiadou (2024) prove that when a FFN subvalue has related concepts with the predicted token, its coefficient score can enlarge the improvement for prediction. In-context heads have similar situations: labels' values ("foo" and "bar") are helpful for next token prediction, so their attention weights decide how much label information is added into the last token for prediction. In fact, we find an attention layer output is a sum of subheadvalues, and each subheadvalue is the product of an attention score and a vector. This characteristic of attention layers is similar to FFN layers. We discuss this in Appendix E.

dataset	input text	true label	acc
sentiment/number	number	foo	94%
sentiment/number	number	bar	100%
number/sentiment	sentiment	foo	92%
number/sentiment	sentiment	bar	87%
animal/country	country	foo	100%
animal/country	country	bar	100%
country/animal	animal	foo	100%
country/animal	animal	bar	100%

Table 2: Word classification accuracy of datasets.

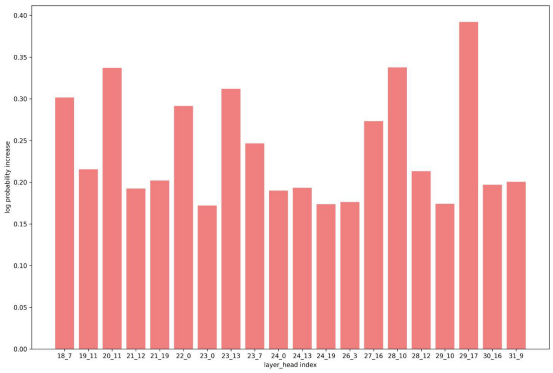


Figure 3: Top 20 heads with largest average log probability increase on sentiment/number dataset.

4 Experiments

4.1 Task and Dataset

We first design an easy NLP task to explore the mechanism of ICL, which is word classification. In this section, the experiments are done on GPT2 large (Radford et al., 2019). We also conduct experiments on real sentence classification datasets on GPT2 large, Llama 7B, 13B and 30B (Touvron et al., 2023), shown in Appendix A-D. We make two datasets for classifying sentiments/numbers

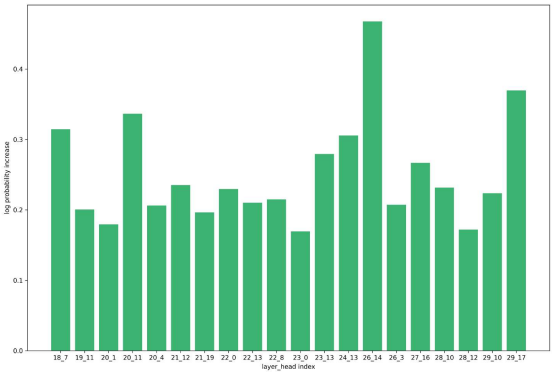


Figure 4: Top 20 heads with largest average log probability increase on animal/country dataset.

and countries/animals. Each dataset has 1,000 sentences. Similar to Wei et al. (2023), we use "foo" and "bar" as labels to remove the semantic priors. We choose two true demonstrations/labels and two false demonstrations/labels, and put the true demonstrations/labels near the input text. The sentences are similar to the example: **"love : bar like: bar eight: foo two: foo one:"**.

We exchange "bar" and "foo" as true labels on each dataset to explore whether GPT2 large really has the ability to classify the words. The results shown in Table 2 indicate that GPT2 large can classify the words by ICL. We choose "foo" as the true label in our experiments.

4.2 Where does In-context Heads Locate

For each sentence, we compute every head's log probability increase and compute the average scores. Based on the results in Section 3, we assume the most contributing heads are in-context heads. We show the top 20 heads with largest log probability increase on sentiment/number and animal/country datasets in Figure 3 and Figure 4. The important heads are in deep layers (18-31). The head indexes are similar on these datasets. Also, the important head indexes on sentence classification tasks are similar (Appendix A). This indicates different ICL tasks share the same in-context heads.

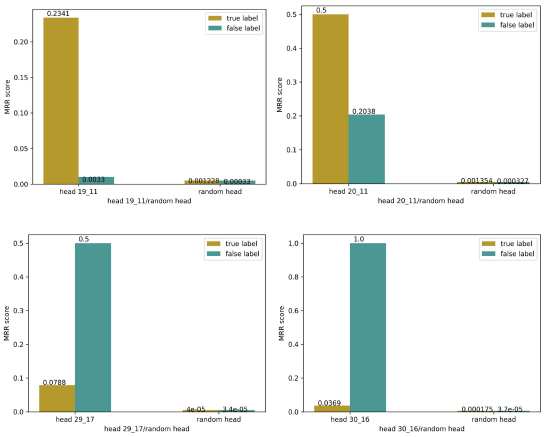


Figure 5: MRR scores of label values in in-context heads (19-11, 20-11, 29-17, 30-16) and random heads.

4.3 Ranking of Labels on In-Context Head Values

Based on the insights in Section 3.2, the label values ("foo" and "bar") in in-context heads should have top rankings when projecting into vocabulary space. We conduct experiments to verify this on sentiment/number dataset. For each in-context head, we random sample a head in the same layer for comparison. We project the label values into vocabulary and compute the label words' mean reciprocal rank (MRR). For example, if the ranking of "foo" is 100 on true label's vocabulary projection, the MRR score is 0.01. If MRR is large, the label token has small ranking in vocabulary space. The scores of heads 18-7, 20-11, 29-17, and 30-16 are shown in Figure 5. Other in-context heads have similar trends with one of these heads. Compared with random heads, label values have much higher MRR scores in in-context heads. The smallest average MRR score 0.0033 is in layer 19, head 11, which still corresponds to an average ranking of 303. This meets our hypothesis of in-context heads: the value-output matrix can extract the label information ("foo"/"bar"). Another finding is the value-output matrix does not control the label information flow, because in in-context heads 29-17 and 30-16, MRR scores of "bar" is higher than "foo".

4.4 Attention Weights on True/False Labels

In this section, we aim to demonstrate the hypothesis that attention weights control the label information flow. For each sentence and each in-context head, we compute the attention weights between the last token's query and true/false labels' keys. The attention weights are shown in Figure 6.

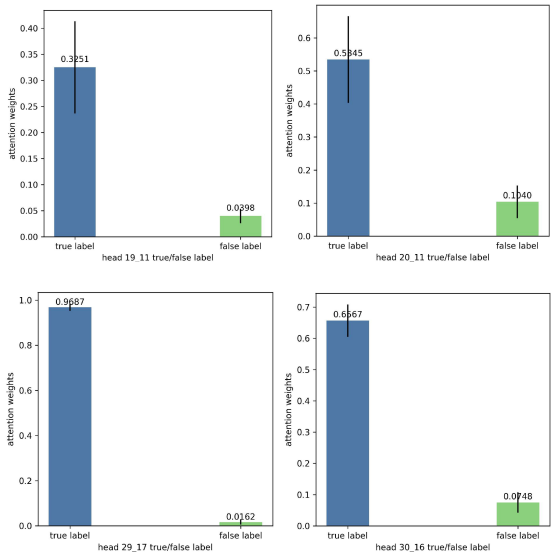


Figure 6: Attention weights between last token query and true/false labels keys in in-context heads (19-11, 20-11, 29-17, 30-16).

In all in-context heads, the attention weights between last token query and true label keys are much

larger than those between last token query and false label keys. This meets our hypothesis that attention weights control the label information flow. Since the log probability increase may be different in various sentences, we random sample one sentence and calculate the attention weights of true labels and the label values' log probability increases in top3 important in-context heads. The relationship is shown in Figure 7. It is not a linear relationship, but it has a increasing trend. Attention weights between 0.9 and 1.0 have larger log probability increase than those between 0.4 and 0.6.

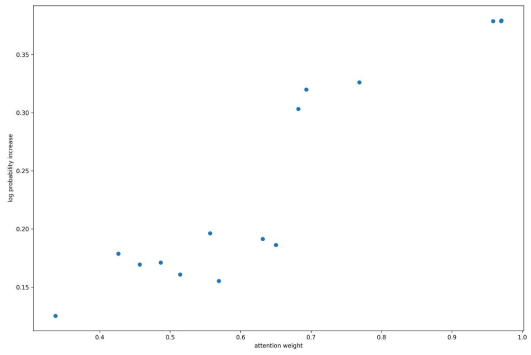


Figure 7: Relationship betwen attention weights and log probability increase.

Let us conclude the mechanism of ICL based on the experimental results. In shallow layers, the demonstration features are merged into their corresponding labels, and the input text features are aggregated into the last token (evidence: Section 3.2). In every in-context head in deep layers, the value-output matrix extracts the label information (evidence: Section 3.2 and 4.3). At the same time, the key matrix and query matrix, which can be regarded as two towers (Huang et al., 2013) for learning the similarity metric, compute the attention weights between the input text and demonstrations. The attention weights control the label information flow (evidence: Section 4.4). There are many in-context heads controlling the label information flow from labels to the last token (evidence: Section 3.1 and 4.2). Finally, the label with larger attention weights are predicted.

5 Explaining Phenomenons of ICL

There are several phenomenons of ICL that haven't been explained. Zhao et al. (2021) illustrate that models tend to predict majority labels and the labels near the input text. Lu et al. (2021) also find that changing the demonstration order can affect predictions a lot. In this section we explain these phenomenons based on our hypothesis.

5.1 Why does Imbalanced Labels Affect Prediction

According to our hypothesis, it is reasonable that the model tends to predict majority labels, because the label information flow is controlled by the attention weights. When a label has high frequency, the sum of attention weights will be larger, thus the probability of this label is larger in final prediction. We design a imbalanced sentiment/number dataset to verify this. For each sentence, we remove the last true demonstration and label. For example, **"love : bar like: bar eight: foo two: foo one:"** is changed to **"love : bar like: bar eight: foo one:"**.

On each in-context head, we compute the sum of attention weights on true/false labels on the imbalanced dataset and compare them with the origin balanced dataset. The results of head 19-11 and 20-11 are shown in Figure 8. Other in-context heads have similar trends. The sum of attention weights on true labels decrease on the imbalanced dataset. On the contrary, the attention weights on false labels increase. The results meet our analysis. The attention weights are computed by a softmax function, so when a true demonstration and its label are removed, the sum of attention weights on false labels will increase.

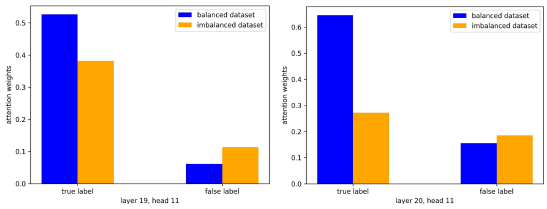


Figure 8: Sum of true/false label attention weights in in-context heads (19-11, 20-11) on balanced and imbalanced datasets.

5.2 Why does Demonstration Order Affect Prediction

The ICL performance is extremely sensitive to the demonstration order. This phenomenon seems to contradict our hypothesis. If the labels and the last token only extract the corresponding demonstrations and the input text, the demonstration order should not affect the prediction. Our assumption of this phenomenon is that the labels not only extract the corresponding demonstrations but also

extract a little adjacent tokens, because the model is not only trained for ICL. Let us assume that the labels can extract 80% corresponding tokens and 20% adjacent tokens. Consider the example sentence **"love : bar like: bar eight: foo two: foo one:"**. In this situation the last token query contains 80% "one"+20% "two". If the demonstration order is changed into **"eight: foo two: foo love : bar like: bar one:"**, the last token query contains 80% "one"+20% "like". Consequently, the attention weights and the predictions will be different.

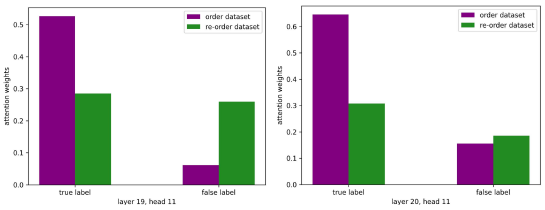


Figure 9: Sum of true/false label attention weights in in-context heads (19-11, 20-11) on order and re-order datasets.

To verify this assumption, we design a re-order dataset based on sentiment/number dataset. Like the previous example, for each sentence we put all the true demonstrations/labels at first, and put all the false demonstrations/labels near the input text. We compute the sum of attention weights on true labels and false labels. The results are shown in Figure 9. We also choose head 19-11 and 20-11 as examples, and other heads have similar trends. The sum of attention weights on re-order dataset are much smaller than those on the origin dataset.

Except analyzing the attention weights, there are also interpretable results supporting our assumption in Table 1. "lover" also exists in 5-key, and "height" (which may be related to "eight") is also contained in 5-key. Compared with 8-key, 11-key has more concepts about numbers. 11-key may extract information from "eight" and "one".

We also show the true labels' attention weights on the in-context heads from layer 18 to layer 29. These in-context heads rank top20 on both sentiment/number dataset and animal/country dataset. We compare the attention weights on the origin sentiment/number dataset, the imbalanced dataset, and the re-order dataset. The results are shown in Figure 10 and Figure 11. Compared with balanced dataset, the average attention weight of imbalanced dataset only increases on head 21-19. In other in-context heads including 19-11, 20-11, 22-0, 24-13 and 27-16, the attention weights decrease a lot. On re-order dataset, the attention weights on all in-context heads drop, especially in 21-19 and 22-0. These results support our hypothesis again: the attention weights control the label information flow.

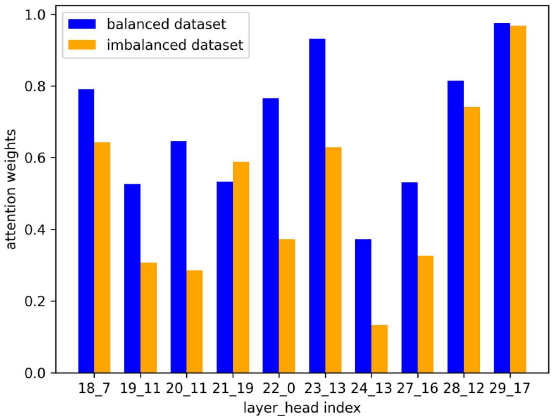


Figure 10: Attention weights of true labels on balanced and imbalanced datasets.

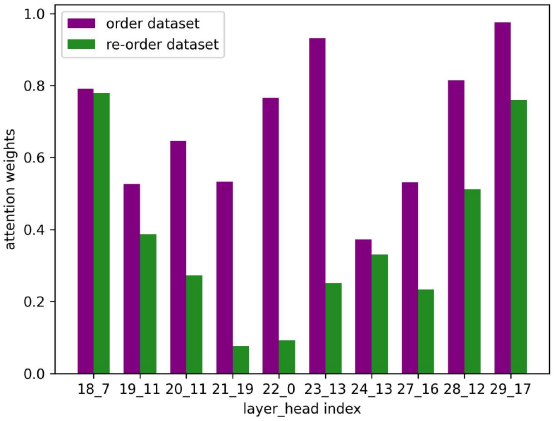


Figure 11: Attention weights of true labels on order and re-order datasets.

6 Discussion

6.1 What Affects ICL Ability

Under our hypothesis, there are four modules related to the ICL ability.

a) Information extraction ability of shallow layers. Shallow layers can be regarded as feature extraction modules. The ability of extracting corresponding demonstrations and the input text decides the quality of features.

b) Value projection ability of in-context heads' value-output matrices. From the results in Figure 5, several in-context heads (such as 19-11 and 30-16) can only project "foo" or "bar". If the value

projection ability is good enough, these in-context heads should project "foo" and "bar" together.

c) Metric learning ability of in-context heads' query and key matrices. The query and key matrices might be the most important module, because they should learn computing different metrics using the same matrices. If different ICL tasks share the same in-context heads, the query and key matrices should learn these metrics jointly.

d) Numbers and parameters of in-context heads. If we regard one in-context head as a two-tower model for metric learning, the parameters of the head are directly related to the learning ability. At the same time, different in-context heads can be regarded as voting or ensemble models, so the head number also controls the learning ability.

6.2 Advantage of Locate-and-Project Method

A popular method for locating important parameters and modules is causal mediation analysis (Vig et al., 2020), which is widely used in existing interpretability studies (Meng et al., 2022; Wang et al., 2022; Hanna et al., 2023; Geva et al., 2023). The core idea of these methods is to mask several parameters/modules in the model and see how much the final prediction is affected. These methods are dynamic methods because they need to run the model many times masking different modules. Our method for locating important layers and heads is static. We only need to run the model once, so our locating method is more efficient.

The method of projecting parameters into vocabulary space is utilized in many studies (Elhage et al., 2021; Geva et al., 2022). Dar et al. (2022) explore how to project the parameters in transformers into vocabulary space. They focus on analyzing the entire matrix, while we consider the values on the most contributing positions in important heads.

In conclusion, our proposed locate-and-project method is efficient. On locating step, we can locate the important layers, heads and positions by only running the model once. On projecting step, instead of analyzing the whole matrix of the heads, we start from projecting the important positions. Therefore, our method could save time for case studies when exploring the mechanisms. Our method is inspired by Yu and Ananiadou (2024). The difference is that our method can locate the important heads, while they sum the heads together to compute the attention subvalues. We discuss this in Appendix E.

7 Related Work

Many studies have explored the mystery of ICL. Min et al. (2022) find that randomly replacing the ground truth labels does not hurt performance much. Wei et al. (2023) argue the reason of this phenomenon is the model can rely on semantic priors. Therefore, they study semantically-unrelated label ICL by transferring the labels into "foo" and "bar" and find that the performance is related to the demonstration-label mapping. Chan et al. (2022) demonstrate that the ICL ability is obtained when training data have enough rare classes. Liu et al. (2021) argue that selecting the closest neighbors as demonstrations can enhance ICL ability. Gonen et al. (2022) propose choose low perplexity demonstrations to increase the performance of ICL. Dong et al. (2022) conclude these methods in a survey for ICL.

Some studies try to explain ICL theoretically. Xie et al. (2021) argue that ICL ability is gained when the pretraining distribution is a mixture of HMMs, and they explain ICL as implicit Bayesian inference. Garg et al. (2022) prove that transformers can learn linear functions by ICL. Akyürek et al. (2022) find transformers can learn linear regression functions and hypothesize that ICL can implement standard learning algorithms implicitly. Li et al. (2023b) explore the softmax regression and find that attention-only transformers are similar with gradient descent models. Von Oswald et al. (2023) and Dai et al. (2022) regard ICL as meta-learning and argue that ICL does gradient descent implicitly.

8 Conclusion

We propose a hypothesis about the mechanism of ICL using our locate-and-project method. Shallow layers merge demonstrations' features into their labels. Deep layers' in-context heads extract the label information by value-output matrices. Query and key matrices compute the attention weights (which can be regarded as similarity metrics) between the input text and the demonstrations. The attention weights control the label information flows to the last token. Moreover, our hypothesis can explain why ICL has majority label bias and recency bias. We conduct experiments on word classification and sentence classification datasets on GPT2 large and Llama 13B, and the results can support our hypothesis. Overall, the locate-and-project method and the hypothesis about ICL mechanism are helpful for future studies on ICL.

9 Limitation

Our experiments are conducted on GPT2 large, Llama 7B, 13B, and 30B. More experiments should be done on larger open source language models. Our hypothesis can explain the ICL mechanism for classification tasks. More studies should be done on other ICL tasks, such as chain-of-thought reasoning (Wei et al., 2022b).

References

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. 2022. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. 2023. Language models can explain neurons in language models. *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Stephanie Chan, Adam Santoro, Andrew Lampinen, Jane Wang, Aaditya Singh, Pierre Richemond, James McClelland, and Felix Hill. 2022. Data distributional properties drive emergent in-context learning in transformers. *Advances in Neural Information Processing Systems*, 35:18878–18891.

Damai Dai, Yutao Sun, Li Dong, Yaru Hao, Zhifang Sui, and Furu Wei. 2022. Why can gpt learn in-context? language models secretly perform gradient descent as meta optimizers. *arXiv preprint arXiv:2212.10559*.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. 2022. Analyzing transformers in embedding space. *arXiv preprint arXiv:2209.02535*.

Qingxiu Dong, Lei Li, Damai Dai, Ce Zheng, Zhiyong Wu, Baobao Chang, Xu Sun, Jingjing Xu, and Zhifang Sui. 2022. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. 2022. What can transformers learn in-context? a case study of simple function classes. *Advances in Neural Information Processing Systems*, 35:30583–30598.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. *arXiv preprint arXiv:2304.14767*.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. *arXiv preprint arXiv:2203.14680*.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2020. Transformer feed-forward layers are key-value memories. *arXiv preprint arXiv:2012.14913*.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2022. Demystifying prompts in language models via perplexity estimation. *arXiv preprint arXiv:2212.04037*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does gpt-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. *arXiv preprint arXiv:2305.00586*.

Jiabang He, Lei Wang, Yi Hu, Ning Liu, Hui Liu, Xing Xu, and Heng Tao Shen. 2023. Icl-d3ie: In-context learning with diverse demonstrations updating for document information extraction. *arXiv preprint arXiv:2303.05063*.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338.

Jia Li, Yunfei Zhao, Yongmin Li, Ge Li, and Zhi Jin. 2023a. Towards enhancing in-context learning for code generation. *arXiv preprint arXiv:2303.17780*.

Shuai Li, Zhao Song, Yu Xia, Tong Yu, and Tianyi Zhou. 2023b. The closeness of in-context learning and weight shifting for softmax regression. *arXiv preprint arXiv:2304.13276*.

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2022. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint arXiv:2202.12837*.

Ioannis Mollas, Zoe Chrysopoulou, Stamatis Karlos, and Grigorios Tsoumakas. 2020. Ethos: an online hate speech detection dataset. *arXiv preprint arXiv:2006.08328*.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. 2022. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. *What In-Context Learning "Learns" In-Context: Disentangling Task Recognition and Task Learning*. Ph.D. thesis, Princeton University.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Suzanna Sia and Kevin Duh. 2023. In-context learning as maintaining coherency: A study of on-the-fly machine translation using large language models. *arXiv preprint arXiv:2305.03573*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

Johannes Von Oswald, Eyvind Niklasson, Ettore Randazzo, João Sacramento, Alexander Mordvintsev, Andrey Zhmoginov, and Max Vladymyrov. 2023. Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. *arXiv preprint arXiv:2305.14160*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022a. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837.

Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, et al. 2023. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*.

Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2021. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*.

Zeping Yu and Sophia Ananiadou. 2024. Locating factual knowledge in large language models: Exploring the residual stream and analyzing subvalues in vocabulary space. *arXiv preprint arXiv:2312.12141*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Zihao Zhao, Eric Wallace, Shi Feng, Dan Klein, and Sameer Singh. 2021. Calibrate before use: Improving few-shot performance of language models. In *International Conference on Machine Learning*, pages 12697–12706. PMLR.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. 2022. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*.

A In-context heads in Sentence Classification Datasets

In this section, we conduct similar experiments with Section 4.2 on GPT2 large on sentence classification datasets, including Stanford Sentiment Treebank binary classification (SST-2) (Socher et al., 2013), Text REtrieval Conference question classification (TREC) (Li and Roth, 2002), AG's news topic classification (AGNews) (Zhang et al., 2015) and Hate Speech Detection (ETHOS) (Mollas et al., 2020). In each dataset, we random sample 1,000 input texts from the test set, and random sample two true demonstrations/labels and two false demonstrations/labels in the training set. We put the true demonstrations/labels near the input text. We do experiments on all the cases predicting the correct labels. The results are shown in Figure 12-15.

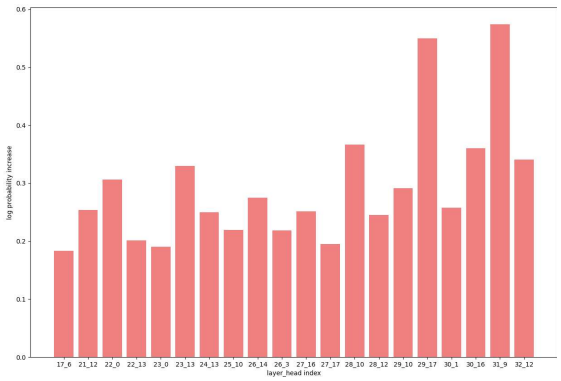


Figure 12: Top 20 heads with largest average log probability increase on SST-2 dataset.

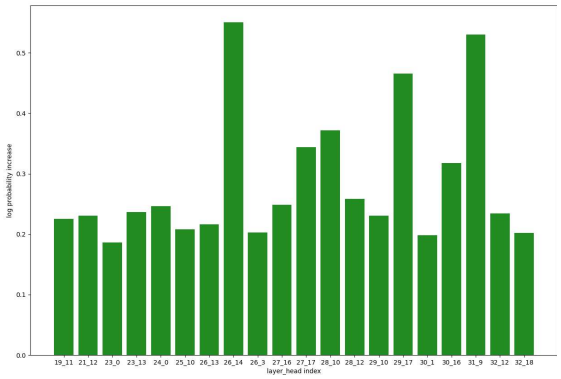


Figure 13: Top 20 heads with largest average log probability increase on TREC dataset.

The head indexes with top20 largest log probability increase are similar in these sentence classification datasets, although the largest head is different.

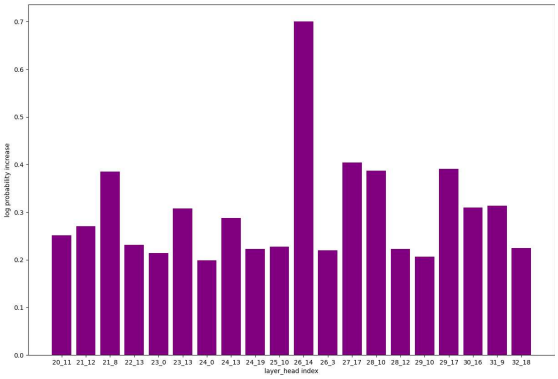


Figure 14: Top 20 heads with largest average log probability increase on AGnews dataset.

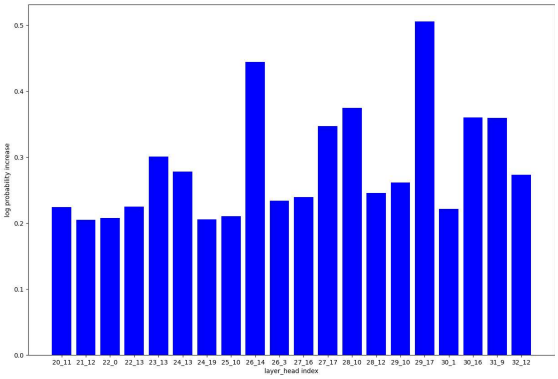


Figure 15: Top 20 heads with largest average log probability increase on EHTOS dataset.

Almost all the heads are in deep layers. Also, there is a large overlap between the head indexes in sentence classification datasets and word classification datasets. Consequently, the in-context heads are important in different ICL tasks.

B Label Rankings and Attention Weights in Sentence Classification Datasets

In this section, we conduct similar experiments with Section 4.2 and 4.3 to see whether the in-context heads in sentence classification tasks have similar characteristic with word classification tasks. We evaluate head 26-14, 28-10 and 29-17, because these heads are important on all the datasets. We compute the MRR scores of label values and the attention weights between last token query and label keys. The results are shown in Figure 16-21. In all datasets, the labels in in-context heads have larger MRR scores than random heads, and the attention scores on true labels are much larger than false labels. These results can support our analysis.

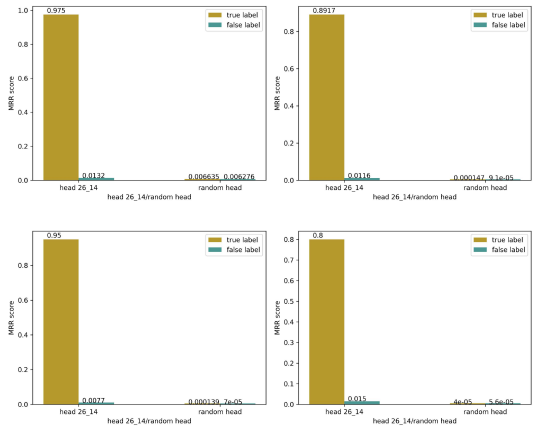


Figure 16: MRR scores of label values in in-context/random heads on 4 datasets (head 26-14).

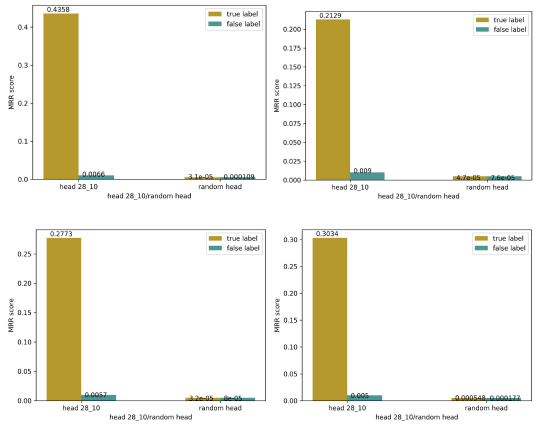


Figure 17: MRR scores of label values in in-context/random heads on 4 datasets (head 28-10).

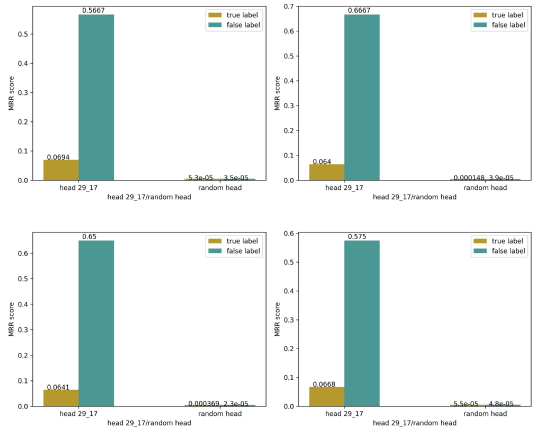


Figure 18: MRR scores of label values in in-context/random heads on 4 datasets (head 29-17).

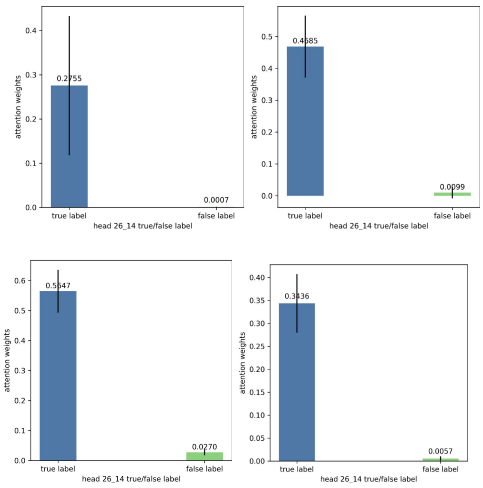


Figure 19: Attention weights between last token query and true/false label keys on 4 datasets (head 26-14).

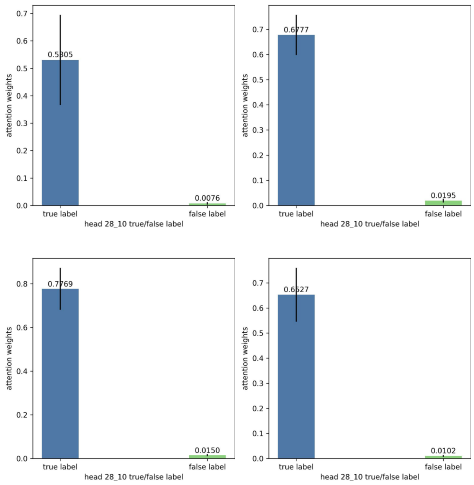


Figure 20: Attention weights between last token query and true/false label keys on 4 datasets (head 28-10).

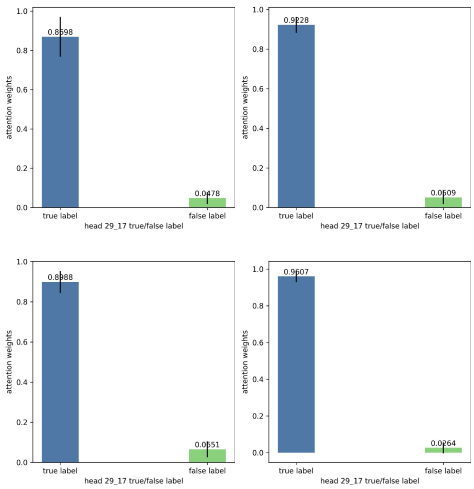


Figure 21: Attention weights between last token query and true/false label keys on 4 datasets (head 29-17).

C Vocabulary Analysis on Sentence Classification ICL Case

We analyze a sentence classification case sampled in AGNews dataset and the results on GPT2 large are shown in Table 3. We take head 23-13 as example. With the prediction "foo", the case is:

Wall St. Bears Claw Back Into the Black (Reuters) Reuters - Short-sellers, Wall Street's dwindling band of ultra-cynics, are seeing green again. : **bar** Stoking the Steamroller No other recording artist can channel American middle-class tastes quite like Chip Davis and his best-selling band. : **bar** Liverpool completes signings of Alonso, Garcia LIVERPOOL, England (AP) – Spanish pair Xabi Alonso from Real Sociedad and Luis Garcia from Barcelona signed five-year contracts with Liverpool on Friday. : **foo** U.S. Doping Watchdog to Question BALCO's Conte - IAAF HELSINKI (Reuters) - U.S . anti-doping officials plan to question Victor Conte after the BALCO head claimed he saw sprinter Marion Jones taking banned drugs, world athletics body the IAAF said Saturday. : **foo** Liverpool Progresses to Champions League; Monaco, Inter Advance Four-time champion Liverpool progressed to soccer Champions League 2-1 on aggregate, overcoming a 1-0 home defeat to AK Graz in the second leg of qualifying. **:**

position	top words in vocabulary space
bar-value	**BAR**, **bars**, **Bars**, bart, **Bar**, bartender, **bar**, Barber
bar-value	bartender, **Bars**, bart, **bars**, **Bar**, Barber, **bar**, **BAR**
foo-value	**foo**, McKenzie, **Foo**, Barney, Walters, Jenner, Murphy, lobster, Handler
foo-value	Walters, **foo**, Barney, McKenzie, Harrington, Murphy, Barber, Barron, Jenner
bar-key	Bloomberg, Investor, billionaires, CNBC, bankers, Companies, JPMorgan, obal, economists, bullish, Barron, HSBC, Friedman, Consumer, business, sellers
bar-key	Buy, Conn, Ok, Previous, Daily, NY, Yes, Anon, US, Ibid, Profit, Staff, Journal, Vanguard, Tribune, Well
foo-key	Buy, iverpool, Ibid, YORK, UNITED, Oliv, Charl, Location, Spanish, Miami, US, Liverpool, Pool, London, Greenwich, United
foo-key	NYT, WATCH, Latest, Exclusive, Previous, UNC, US, Watch, Possible, Ibid, Statement, Reaction, UK, Reuters, United, Smoke
last-query	ruary, Pipe, lihood, swick, Flavoring, iverpool, paddle, paraph, Lake, Repe, tong, bole, etheless, Lakes

Table 3: Top words of labels and last token in layer 23, head 13 on a sentence classification case.

In this case, the false demonstrations with label "bar" are sampled from the "Business" class. The true demonstrations with label "foo" and the input text are sampled from the "Sports" class. On label values, "bar" and "foo" have top rankings. On label keys, the labels correspond to business demonstrations extract the concepts about business, such as "investor" and "profit". The top tokens of true labels are related to places such as "Liverpool" and "Spanish". It seems that the reason why the model predicts "foo" is the word "Liverpool" exists in both the input text and the first true demonstration. We find "iverpool", "Lake" and "Lakes" in last token query, which is related to "iverpool" and "Pool" in the first true label key.

Inspired by Bills et al. (2023), we utilize ChatGPT to analyze the relationship between the top words on label keys and the corresponding demonstrations using the prompt **"Try your best to explore the relationship between the sentence and the words"**, and we observe good results. The results of the first demonstration are as below:

The sentence "Wall St. Bears Claw Back Into the Black (Reuters)" suggests a financial context, indicating a positive turn for short-sellers in the stock market. The phrase "Claw Back Into the Black" implies a recovery or improvement in their financial situation. The words provided seem to be related to the financial and economic domain, with mentions of prominent entities such as 'Bloomberg', 'Investor', 'CNBC', 'JPMorgan', 'HSBC', 'Economist', and 'Forbes'. These entities are likely players in the financial world, including news agencies, banks, economists, and investment platforms. Additionally, words like 'billionaires', 'CEOs', 'bankers', 'entrepreneurs', and 'capitalists' hint at the involvement of influential and wealthy individuals in the financial landscape.

This implies the potential capability to leverage our methods to interpret the predictions automatically by analyzing what the label keys/last token extract from the demonstrations/text input.

D Results on Llama 7B, 13B and 30B

We conduct experiments on Llama 7B, 13B and 30B on these sentence classification datasets. We compute the label rankings and attention weights on the head with the largest log probability increase in each model. The results in Figures 22-27 are similar with the results in GPT2 large. This indicates these models have the same mechanism for ICL.

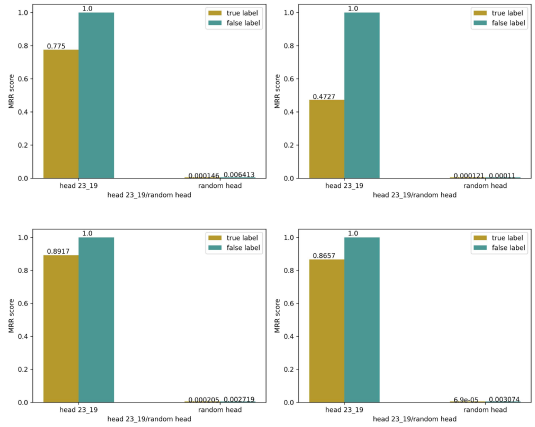


Figure 22: MRR scores of label values in in-context/random heads on 4 datasets (Llama 7B).

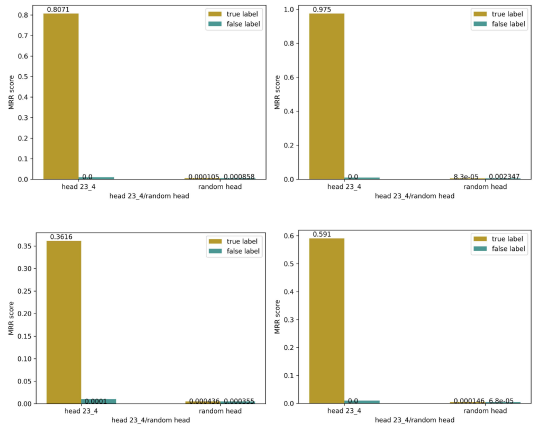


Figure 23: MRR scores of label values in in-context/random heads on 4 datasets (Llama 13B).

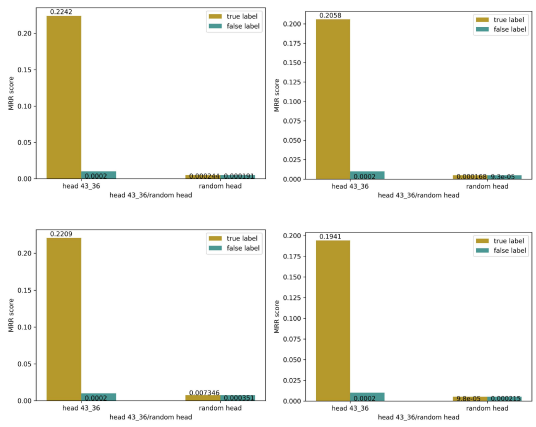
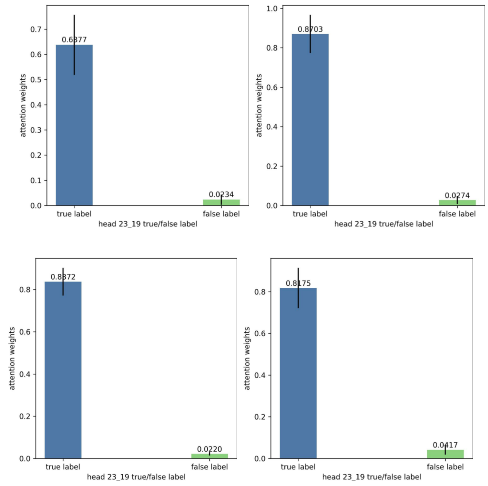


Figure 24: MRR scores of label values in in-context/random heads on 4 datasets (Llama 30B).



=

Figure 25: Attention weights between last token query and true/false label keys on 4 datasets (Llama 7B).

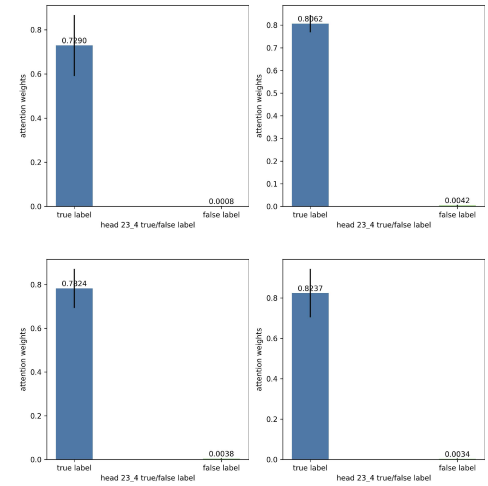


Figure 26: Attention weights between last token query and true/false label keys on 4 datasets (Llama 13B).

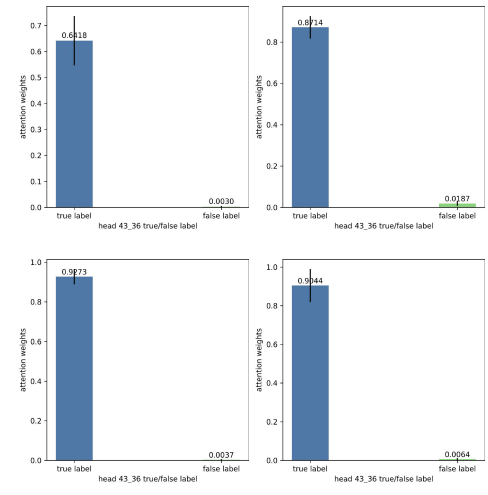


Figure 27: Attention weights between last token query and true/false label keys on 4 datasets (Llama 30B).

E Mechanism Analysis of Our Method

The final embedding F for predicting the next word is a sum of the $0th$ layer input Lin_0 (word embedding + position embedding), each layer's attention output $ATTN_i$ and each layer's FFN output FFN_i.

$$F = Lin_0 + \sum_{i=0}^{L-1} ATTN_i + \sum_{i=0}^{L-1} FFN_i$$

where L is the layer number. Geva et al. (2020) prove that a FFN output is the sum of FFN subvalues ffn_i^k, and each FFN subvalue is the product of a coefficient score m and a vector $fc2_i^k$ in the second FFN matrix.

$$FFN_i = \sum_{k=0}^{N-1} m_i^k fc2_i^k$$

where N is the number of neurons in FFN layers. Yu and Ananiadou (2024) prove that an attention output can also be regarded as the sum of attention subvalues on different positions. Each attention subvalue is the element-wise product of the multi-head attention weight vector and the value-output vector.

$$ATTN_i = \sum_{p=0}^{S-1} attn_i^p$$

where S is the length of the input sequence. Consequently, the final output is the sum of the $0th$ layer input, many attention subvalues and many FFN subvalues on different layers.

$$F = Lin_0 + \sum_{i=0}^{L-1}\sum_{p=0}^{S-1} attn_i^p + \sum_{i=0}^{L-1}\sum_{k=0}^{N-1} m_i^k fc2_i^k$$

By analyzing the distribution change, Yu and Ananiadou (2024) prove that a subvalue is helpful for the final prediction word w if w ranks top when projecting the subvalue into vocabulary space. This is because the before-softmax values of the subvalues are added in a sum function. Consider a FFN subvalue v in the last layer L. x is the minus of the final vector and the FFN subvalue:

$$F = x + v$$

The probability of the predicted word w on F, x and v are computed by the softmax function:

$$p(w|F) = \frac{exp(e_w \cdot (x+v))}{exp(e_1 \cdot (x+v)) + ... + exp(e_B \cdot (x+v))}$$

$$p(w|x) = \frac{exp(e_w \cdot x)}{exp(e_1 \cdot x) + ... + exp(e_B \cdot x)}$$

$$p(w|v) = \frac{exp(e_w \cdot v)}{exp(e_1 \cdot v) + ... + exp(e_B \cdot v)}$$

where e_w is the wth row of the unembedding matrix E, with B words in vocabulary. Term $e_w \cdot x$ as the bs-value (before-softmax value) of w on x, then F can be regarded as bs-value vectors:

$$bs(x+v) = [bs_1^{x+v}, bs_2^{x+v}, ..., bs_w^{x+v}, ..., bs_B^{x+v}]$$

The probability of w can be computed by bs-values:

$$p(w|x+v) = \frac{exp(bs_w^{x+v})}{exp(bs_1^{x+v}) + ... + exp(bs_B^{x+v})}$$

And the bs-values of $x+v$ can be computed by a direct sum of bs-values x and v:

$$bs(x+v) = bs(x) + bs(v)$$

Consequently, the top tokens of the FFN subvalue v in vocabulary space are related to the predictions of F. If w ranks top in v in vocabulary space, v is helpful for increasing the probability of w because bs_w^v is large and bs_w^{x+v} will increase much.

Take a vocabulary with three words ("foo", "bar" and "unknown") as an example. $bs(x)$ is [1, 2, 3], the words' probabilities in x are [0.09, 0.24, 0.67]. If $bs(v)$ is [5, 1, 2], "foo" is the top ranking token in vocabulary space because its bs-value is the largest. $bs(F)$ will be [6, 3, 5], and the probabilities will change into [0.70, 0.04, 0.26]. In this case, v helps increase the probability of "foo" from 0.09 to 0.70. The coefficient score m in the FFN subvalue v is helpful for enhancing the probability change. If m changes from 1.0 to 2.0, $bs(v)$ changes from [5, 1, 2] to [10, 2, 4], and then the probabilities of F will be [0.981, 0.001, 0.018].

This characteristic can be promoted into all subvalues. If the final token w ranks top when projecting a subvalue into vocabulary space, this subvalue is helpful for the final prediction of w. Furthermore, Yu and Ananiadou (2024) prove that using log probability increase can help locate the most important subvalues, because the curve of log probability increase has a linear monotonically increasing shape. In other words, the sequence of subvalues do not affect the log probability increase score much. Take the $0th$ attention subvalues as example. $log(w|Lin_0 + attn_0^n) - log(w|Lin_0)$ is similar to $log(w|Lin_0 + attn_0^m + attn_0^n) - log(w|Lin_0 + attn_0^m)$. Therefore, the significance scores of all subvalues can be compared together.

A problem of Yu and Ananiadou (2024) is that they take the product of the multi-head vector and the value-output vector to compute the attention subvalues. They do not consider the roles of different heads. In our work, we take a step further and find that the attention output is the sum of head vectors:

$$ATTN_i = \sum_{h=0}^{H-1} attnhead_i^h$$

where H is the head number in each attention layer, and each head vector is the sum of different subheadvalues:

$$attnhead_i^h = \sum_{p=0}^{S-1} \alpha_i^{hp} \cdot attnheadv_i^{hp}$$

where S is the length of the input sequence. α_i^{hp} is the attention score on the pth position in the hth attention head in layer i, computed by the last token query and the pth token key in this head. The attention output can also be regarded as the sum of attention subheadvalues on different positions in different heads. Each subheadvalue is computed by an attention score and a vector. Similar to the roles of FFN subvalues' coefficient scores, an attention score can enhance the probability change when the final predicted token has top ranking when projecting its corresponding subheadvalue into vocabulary space. Our experiments match this theory. In in-context heads, "foo" and "bar" ranks top when projecting label values into vocabulary space. The corresponding attention scores on true labels are large, which are helpful for increase the probability of "foo" in final prediction. Similarly, we compute the log probability increase I of each head vector and subheadvalue to locate the important heads and subheadvalues.

$$I_i^h = log(w|Lin_i + attnhead_i^h) - log(w|Lin_i)$$

$$I_i^{hp} = log(w|Lin_i + \alpha_i^{hp} attnheadv_i^{hp}) - log(w|Lin_i)$$

Generally, the final embedding F can be regarded as the sum of many attention subheadvalues and FFN subvalues:

$$F = Lin_0 + \sum_{i=0}^{L-1}\sum_{h=0}^{H-1}\sum_{p=0}^{S-1} \alpha_i^{hp} attnheadv_i^{hp} + \sum_{i=0}^{L-1}\sum_{k=0}^{N-1} m_i^k fc2_i^k$$

where α_i^{hp} is the attention score in attention layer i, head h, position p. m_i^k is the coefficient score of the kth FFN neuron in FFN layer i. When analyzing a case, we first locate the most important attention subheadvalues and FFN subvalues by calculating their log probability increase. Then we analyze the coefficient scores of FFN subvalues by calculating the inner products between the FFN subkeys and previous subvalues. Similarly, we analyze the queries and keys which compute the attention scores. Using this method, we can figure out which FFN subvalues and attention subheadvalues are helpful for predicting the final word, and which parameters are helpful for increasing the coefficient scores and attention scores. Consequently, we can find why the model has the final predictions.