Exact Conversion of In-Context Learning to Model Weights in Linearized-Attention Transformers

Brian K Chen [1 2] **Tianyang Hu** [3] **Hui Jin** [3] **Hwee Kuan Lee** [1 2 4 5 6 7] **Kenji Kawaguchi** [1]

Abstract

In-Context Learning (ICL) has been a powerful emergent property of large language models that has attracted increasing attention in recent years. In contrast to regular gradient-based learning, ICL is highly interpretable and does not require parameter updates. In this paper, we show that, for linearized transformer networks, ICL can be made explicit and permanent through the inclusion of bias terms. We mathematically demonstrate the equivalence between a model with ICL demonstration prompts and the same model with the additional bias terms. Our algorithm (ICLCA) allows for exact conversion in an inexpensive manner. Existing methods are not exact and require expensive parameter updates. We demonstrate the efficacy of our approach through experiments that show the exact incorporation of ICL tokens into a linear transformer. We further suggest how our method can be adapted to achieve cheap approximate conversion of ICL tokens, even in regular transformer networks that are not linearized. Our experiments on GPT-2 show that, even though the conversion is only approximate, the model still gains valuable context from the included bias terms.

1. Introduction

Large Language Models (LLMs) have displayed incredible performance and capabilities with the introduction of models such as BERT (Devlin et al., 2018) and GPT (Radford et al., 2018; Brown et al., 2020). One such ability is In-Context Learning (ICL), which involves learning to generalize and make predictions on new tasks based on demonstration examples provided in the prompt. Numerous studies have shown LLMs solving increasingly complex tasks with ICL as compared to fine-tuning using labeled data (Wei et al., 2022).

In-context learning is an attractive proposition compared to traditional parameter-based model update methods such as fine-tuning (Wei et al., 2021; Ouyang et al., 2022) or knowledge editing (Dai et al., 2021; Meng et al., 2022a;b). Demonstration examples can be directly included in the prompt. There is no need to edit the parameters of the model which can be costly when considering models with billions of parameters such as OpenAI's GPT-4 and Google's Bard. In terms of performance, ICL prompting methods (Wei et al., 2022; Zhou et al., 2022) also demonstrate stronger generalization and more precise control of the model behavior (Zheng et al., 2023). Such an advantage is expected to be more prominent as LLMs get larger and more powerful. In addition, ICL presents a degree of interpretability when incorporating demonstration examples into the model. The examples often take the form of natural language prompts or other readily understandable inputs that can be understood by humans. This is often not the case when working with machine learning.

Despite its promising qualities, there are clear shortcomings to the ICL paradigm compared to the traditional gradient-based fine-tuning approaches. While not requiring costly parameter updates like other methods, ICL does not permanently modify the model weights. All included contextual information is only temporary. Demonstration examples are only incorporated on a case-by-case basis and will not be applied to future input prompts. Studies on permanently including ICL information in the model weights are still limited. There are some attempts aiming to distill the context information from ICL prompts into the model parameters (Askell et al., 2021; Choi et al., 2022; Snell et al., 2022). However, these so-called context distillation methods are largely empirical and lack theoretical guarantees.

In this study, we attempt to incorporate ICL demonstration prompts into the parameters of an LLM through the addition of a bias term to the attention module. We do so for a specific class of transformers with linearized attention modules which have already shown promise in real-world

---

[1]National University of Singapore [2]Bioinformatics Institute, Agency for Science, Technology and Research (A*STAR) [3]Huawei Noah's Ark Lab [4]Nanyang Technological University [5]Singapore Eye Research Institute [6]Singapore International Research Laboratory on Artificial Intelligence [7]Singapore Institute for Clinical Sciences. Correspondence to: Tianyang Hu <hutianyang.up@outlook.com>, Kenji Kawaguchi <kenji@comp.nus.edu.sg>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

applications for improving the efficiency of LLMs.

There are a few key advantages of the ICL conversion method to be introduced in Section 4.4. First, unlike the fine-tuning data, which is often just another sampled dataset, the ICL prompts used here are highly interpretable and often meaningful in the context of natural languages. Since we are doing exact conversion rather than fine-tuning, we will not face issues with algorithmic convergence or stability that arise with fine-tuning methods. As long as the ICL ability of the original model is strong and we have suitable ICL prompts, our method will be effective. Second, our proposed method is simple to implement. All that is required is a linear update to the biases of the network, which is computationally very inexpensive compared to conducting gradient descent methods. The converted extra bias term only amounts to about 1% of the total number of parameters, which is very efficient to store. Furthermore, such extra weights can be combined freely to incorporate more in-context prompts or thrown away to recover the original model. Consequently, ICL conversion methods shape up to be a powerful new tool for incorporating new information into existing LLMs in a cheap, interpretable fashion.

In addition to an exact conversion method for transformers with linearized attention modules, we also provide an approximate conversion method for regular transformers in Section 5. Transformers with softmax attention remain the most popular architecture for most state-of-the-art models so even an approximate conversion method bears significance. While we can only achieve approximate conversion in this case, we show experimentally that the model retains some degree of In-context information by including our bias terms. Approximate conversion of regular transformers is also easy to implement and readily interpretable.

Our work presents the following key contributions:

- We demonstrate how ICL relationships are captured between the Key and Value matrices of the attention module.
- We present an exact conversion method that captures the ICL prompts into the model weights by introducing a novel bias term into the model architecture. This conversion is done for models based on linearized attention. This methodology is theoretically and mathematically supported in contrast to existing methods.
- We extend our methodology to an approximate conversion method that covers a wider setting. This includes the regular transformer with softmax-based attention.
- We provide experimental justification for this methodology:
  - We showcase the exact conversion of the ICL prompt into a model with linear attention.
  - We demonstrate an effective approximate conversion of ICL prompts within a pretrained GPT-2 model with regular attention.

2. Related Works

Transformers and Large Language Models The transformer architecture was first introduced by Vaswani et al. (2017) and has revolutionized multiple fields including, but not limited to, natural language processing, computer vision, time-series analysis, and bioinformatics. The field of natural language processing has been especially impacted by transformers being adopted in LLMs such as GPT (Radford et al., 2018) and BERT (Devlin et al., 2018). Through the adoption of the attention mechanism, transformer architectures are capable of capturing long-range dependencies of the input sequence and more sophisticated abilities such as reasoning (Bubeck et al., 2023; Abbe et al., 2023; Dziri et al., 2024; Wang et al., 2024; Giannou et al., 2023; Gao et al., 2024). This was difficult for previous alternatives such as LSTMs (Hochreiter & Schmidhuber, 1997) and RNNs (Rumelhart et al., 1985) due to their sequential nature.

Linear and Kernel Attention Within the transformer layer, the softmax function is usually identified as a bottleneck. It introduces quadratic complexity in computation concerning input length. As an alternative, studies have approximated the attention mechanism by the linear product (Irie et al., 2022; Bahdanau et al., 2016; Luong et al., 2015). This is further extended to approximate the softmax by a kernel product (Blanc & Rendle, 2018; Katharopoulos et al., 2020). As a field growing in popularity, there have been multiple studies attempting to find the most appropriate kernel, either for training models from scratch (Chowdhury et al., 2021), or for approximating the softmax. (Choromanski et al., 2020). In particular, RetNet (Sun et al., 2023) claims to match state-of-the-art transformer architectures using just linear attention with some added tricks such as multi-scale retention. Currently, while linearized transformer networks still occupy the minority, the prevalence of lengthy prompts has shown how important it is to rescue the computational cost of LLMs. We predict that as time passes, linearized transformer networks will grow a great deal both in proficiency and significance.

In Context Learning and Context Distillation The ICL capability of LLMs has gained much traction since being demonstrated on GPT-3 (Brown et al., 2020). It has been considered an emergent capability that only arises in models with many layers and an extremely large number of parameters (Dong et al., 2022). However, studies have demonstrated that ICL can occur in smaller models, albeit in a less sophisticated manner (Wei et al., 2023). Due to the rising interest in ICL, multiple works have explored why ICL works and how it can be improved. Among others, Garg et al. (2022), Akyurek et al. (2022) and Xie et al. (2021)

constructed a Bayesian framework to explain how ICL works when the training and inference distributions are distinct. Bai et al. (2023) provided a more concrete mechanism and demonstrated how transformers can implement a large range of learning algorithms in context. They also provided experimental results showing the strong in-context algorithm selection capabilities of standard transformer architectures. To make the effect of ICL more permanent, various methods have been developed to distill in-context prompts into the models. Askell et al. (2021), Choi et al. (2022) and Snell et al. (2022) proposed various fine-tuning schemes to match the input-output relationships with ICL. This line of work is largely empirical and usually involves multiple LLMs, which are not efficient and lack theoretical guarantees.

Based on our observation, current studies attempt to encode ICL permanently through a form of fine-tuning with gradient descent. Such approaches only amount to approximations and can be computationally very expensive on increasingly large models. Our work seeks to take advantage of the structure of linear and kernel attention to achieve the exact conversion of ICL to the weights of our LLM thereby avoiding both limitations of existing methodologies.

3. Preliminaries

Introduced in the seminal paper "Attention is All You Need" (Vaswani et al., 2017), transformer models handle sequence-to-sequence tasks more proficiently by leveraging the attention mechanism to weigh the influence of tokens against each other. Traditional sequence-to-sequence models, such as GRUs and LSTMs, suffer from difficulties in capturing long-range dependencies due to the inherently sequential nature of their operations. Such a problem does not occur in Transformers because each token has access to all the other tokens along the sequence allowing distant relationships to be captured.

In this paper, we assume that the input of the transformer model is initially passed through an encoder layer to obtain a sequence of vector tokens $X = [X_1, ..., X_N]^T$ which are then fed into the transformer layers. Each token has dimensionality d_{in} such that $X \in \mathbb{R}^{N \times d_{in}}$.

3.1. The Attention Mechanism

The core aspect of the transformer is the attention mechanism. Inspired by the cognitive process of information retrieval in database systems, the attention mechanism uses query, key, and value matrices to emulate how we would sort information in real life within a large database.

While in practical applications we often conduct multi-headed attention, this study emphasizes single-headed attention due to its simplicity. All following definitions and methods can be translated to multi-headed attention by addressing each attention head separately.

Definition 3.1 (Single-headed Attention Layer)**.** Given input data $X \in \mathbb{R}^{N \times d_{in}}$, a single attention layer is characterized by trainable matrices $W_Q, W_K \in \mathbb{R}^{d_{in} \times d_K}, W_V \in \mathbb{R}^{d_{in} \times d_V}$. The Single-headed attention layer takes the form:

$$O(Q,K,V) = \mathrm{Attn}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_K}}\right)V, \quad (1)$$

where O is the output, $Q = XW_Q$, $K = XW_K$ and $V = XW_V$.

For any given matrix Z, let Z_i return the i-th row as a vector and denote $sim(\cdot,\cdot)$ as a similarity score (Note vectors are all vertical). Drawing upon the work of Katharopoulos et al. (2020), the single-headed attention layer can be generalized as follows:

$$O_i^T = \frac{\sum_{j=1}^N \mathrm{sim}(Q_i, K_j) V_j^T}{\sum_{j=1}^N \mathrm{sim}(Q_i, K_j)}. \quad (2)$$

This is the same as regular single-headed attention when $\mathrm{sim}(q,k) = \exp(\frac{qk^T}{\sqrt{d_K}})$. Such an approach can be extended to a linearized (kernel) attention setting by approximating the similarity score $\mathrm{sim}(\cdot,\cdot)$ by a kernel $\mathrm{Ker}(\cdot,\cdot)$ with associated feature map $\phi(\cdot)$. There are many different choices for $\phi(\cdot)$. For example, Katharopoulos et al. (2020) uses a simple $\phi(x) = \mathrm{elu}(x) + 1$. Other feature maps are also further studied showcasing varying levels of success (Zhang et al., 2024; Katharopoulos et al., 2020; Choromanski et al., 2020).

This is further generalized into the following linearized attention module:

Definition 3.2 (linearized attention Layer)**.** Given input data $X \in \mathbb{R}^{N \times d_{in}}$, a single attention layer is characterized by trainable matrices $W_Q, W_K \in \mathbb{R}^{d_{in} \times d_K}, W_V \in \mathbb{R}^{d_{in} \times d_V}$. The linearized attention layer takes the form of:

$$\begin{aligned} O_i^T &= \frac{\sum_{j=1}^N \phi(Q_i)^T \phi(K_j) V_j^T}{D_1(Q_i)^T D_2(X)} \\ &= \frac{\phi(Q_i)^T \sum_{j=1}^N \phi(K_j) V_j^T}{D_1(Q_i)^T D_2(X)}, \end{aligned} \quad (3)$$

where $\phi(\cdot)$ is the associated feature map and $D_1(Q_i)^T D_2(X)$ is a normalizing term depending on the network architecture and input X. Typically D_2 is a function of $K = XW_K$, but we keep it as a function of X itself to capture a more general class of models. For instance, when implementing kernel softmax we would take $D_1(Q_i) = \phi(Q_i)$, $D_2(X) = \sum_{j=1}^N \phi(K_j) = \sum_{j=1}^N \phi(X_j W_K)$ and when implementing linear attention we would take $D_1(Q_i) = 1, D_2(X) = 1$.

For simplicity, Katharopoulos et al. (2020) state the numerator is easier to understand when stated in the vectorized form:

$$(\phi(Q)\phi(K)^T)V = \phi(Q)(\phi(K)^T)V. \quad (4)$$

In real-world applications involving sequential data such as NLP, transformers are typically autoregressive. In the regular attention setting this requires applying a mask to the attention score matrix QK^T within the softmax function. However, the linearized attention layer requires the following modification:

Definition 3.3 (Masked linearized attention Layer)**.** Given input data $X \in \mathbb{R}^{N \times d_{in}}$ a single attention layer is characterized by trainable matrices $W_Q, W_K \in \mathbb{R}^{d_{in} \times d_K}, W_V \in \mathbb{R}^{d_{in} \times d_V}$. The Masked linearized attention Layer takes the form of:

$$O_i^T = \frac{1}{D_1(Q_i)^T D_2(X)} \phi(Q_i)^T \sum_{j=1}^{i} \phi(K_j) V_j^T. \tag{5}$$

In this study, we introduce the concept of the Key-Value matrix. The Key-Value matrix is not widely explored within existing literature largely because it does not exist within the classic attention mechanism. However, it is present in linearized attention models and is the portion of the model that captures the ICL information as we find in §4.4

Definition 3.4 (Key-Value Matrix)**.** Given the definition of linearized attention found in Equation (3), we define the Key-Value matrix A as:

$$A = \sum_{j=1}^{N} \phi(K_j) V_j^T. \tag{6}$$

The definition is the same for masked linearized attention except the sum is up to token i rather than over the whole sequence with length N.

We can see that the addition of extra tokens to the input will not change the $\phi(Q_i)$ term but only the Key-Value matrix. Hence, conversion within the Key-Value matrix A allows for conversion within the linearized attention Layer.

3.2. In Context Learning

In traditional supervised learning, models are generally trained on a labeled dataset, which provides the associations or relationships between inputs and outputs. This approach often necessitates a substantial amount of labeled data and significant computational resources for the sake of fine-tuning the parameters of the model.

Contrarily, in-context learning allows language models to learn tasks based on a few demonstration examples without additional training. This is done by appending demonstration examples to the original prompt following certain criteria (Dong et al., 2022). Demonstrations are typically written in natural language which is an interpretable way to communicate with LLMs.

Typically, we organize demonstration examples in a purposeful manner. A demonstration set consists of multiple examples, each containing a query input text and corresponding label.

In this paper, we omit the structure of the demonstration set. Instead, we treat both the input prompt and demonstration as mere sets of tokens. This includes but is not limited to existing ICL prompt structures. We find this more straightforward mathematically and covers a larger range of scenarios. We consider a standard case where we have an input prompt $X = [X_1, ..., X_N]^T$ of length N and a Demonstration ICL prompt $X' = [X'_1, ..., X'_M]^T$ of length M. The final input is a concatenation of these two $[X';X]^T = [X'_1, ..., X'_M, X_1, ..., X_N]^T \in \mathbb{R}^{(M+N) \times d_{in}}$. Each X_i or X'_i represents a single token and they are arranged horizontally as the rows of the matrices. We refer to the tokens in the ICL prompt $\{X'_i\}_{i=1}^M$ as the ICL tokens and the tokens in the input prompt $\{X_i\}_{i=1}^N$ as the input tokens.

4. Exact Conversion of In-Context Learning to Model Weights

In this section, we formally introduce the ICL weight conversion process. We present that ICL prompts can be captured by a combination of the key and value matrix within a linearized attention setting.

Initially, in section 4.1 and section 4.2, we examine the simplest case with linear attention (Identity Kernel and normalizing factor equalling one). In section 4.1 we demonstrate that ICL conversion cannot be achieved through modifying existing weights. This is solved in section 4.2 through the addition of bias. Subsequently, in section 4.3 we outline a list of assumptions. Transformer models that feature linearized attention and satisfy these assumptions can also achieve bias conversion through the inclusion of bias. Finally, in section 4.4 we formalize the bias terms required for such a linearized attention model.

4.1. Direct Conversion for Linear Attention

In this section, we provide a naive attempt to achieve direct conversion of ICL prompts into the weights of Linear Attention layer. This involves taking equation (3) and setting $D_1(Q_i) = 1, D_2(X) = 1$, and $\phi(\cdot) = I(\cdot)$ as the identity mapping. We want to find corresponding weights W_1, W_2 such that the following equivalence holds true for all $X \in \mathbb{R}^{N \times d_{in}}$,

$$W_1^T X^T X W_2 = W_K^T X^T X W_V + W_K^T X'^T X' W_V. \tag{7}$$

This is drawn from the vectorized form of Equation (4), forming an affine equation that can be rewritten as:

$$\begin{aligned} &(W_2^T \otimes W_1^T)\mathrm{vec}(X^T X) = \\ &(W_V^T \otimes W_K^T)\mathrm{vec}(X^T X) + (W_V^T \otimes W_K^T)\mathrm{vec}(X'^T X'). \end{aligned} \tag{8}$$

Since $X = 0$ is a potential prompt, this means that one would have to have $(W_V^T \otimes W_K^T)\mathrm{vec}(X'^T X') = 0$, which is not true in general. We further explore possible assumptions to achieve such a conversion in the appendix A but ultimately are unable to find a feasible option. Under very specific circumstances it may be possible to directly convert ICL prompts into the model weights. However, it is likely such circumstances will not apply to real-world scenarios so we will not examine this direction further.

4.2. Addition of Free Parameters

Rather than direct conversion, another way to achieve equivalence in equation (7) is through the inclusion of free parameters. Due to the linearity of the attention module, bias arises as a very natural option. Typically this is included separately within the Key, Value and Query matrices in the form W_K+b_K,W_V+b_V and W_Q+b_Q and has limited effect on model performance (Namazifar et al., 2023).

Previously, we mentioned that the Key-Value matrix is integral to the incorporation of ICL prompts. A natural extension of this principle is to include bias to the Key-Value matrix itself. We refer to this term as b_{KV}. This makes the linear attention with bias b_{KV}:

$$\mathrm{LinAttn}(Q,K,V,b_{KV})=Q(K^TV+b_{KV}). \tag{9}$$

This enables a simple conversion for a given ICL weight as follows.

Theorem 4.1 (Bias conversion for Linear Attention)**.** *For weights W_Q,W_K,W_V, input prompt X, ICL prompt X' and initial bias b_{VK}, we have that:*

$$\begin{aligned}\mathrm{LinAttn}([X';X]W_Q,[X';X]W_K,[X':X]W_V,b_{V,K})\\=\mathrm{LinAttn}(XW_Q,XW_K,XW_V,b'_{V,K})\end{aligned} \tag{10}$$

holds for all X if we set

$$b'_{V,K}=b_{V,K}+W_K^TX'^TX'W_V. \tag{11}$$

The proof is trivial by definition. This conversion is simple because one does not need to consider causal masks and positional encodings.

4.3. Assumptions Necessary for Conversion

In this section, we explore what models other than simple linear attention are appropriate for our ICL conversion method. A similar bias term can be found for any linearized attention model that satisfies all of these requirements.

Assumption 4.2 (Autoregressive)**.** For any transformer model, we assume that it is autoregressive. At each position, the model can only attend to that token and previous tokens. This prevents the model from seeing into the future.

The autoregressive property of the model is essential because it prevents the ICL tokens from being affected by the input tokens. If the model isn't autoregressive, the ICL tokens themselves will be affected by the input prompt from layer to layer. If the ICL tokens vary depending on the input prompts, then a conversion satisfying one input prompt may not satisfy another. Most popular LLM models, especially in NLP are already autoregressive by nature so this assumption is easily fulfilled.

Assumption 4.3 (Relative Positional Encoding)**.** We assume that the transformer model we are dealing with utilizes relative positional encoding. This means that a token's positional embedding depends not on its absolute position, but rather on its position relative to the other tokens.

This assumption is not an absolute necessity but makes the conversion process significantly easier. When using absolute positional embedding, there needs to be a linear transformation following the initial input layer to correct for the positional shift. Without such a linear transformation, the inclusion of ICL prompts will change the value of the input prompts and make the conversion more complicated. For simplicity in this study, we will focus on model architectures that include relative positional encodings such as RoFormer (Su et al., 2023) and Llama (lla, 2023).

Assumption 4.4 (Separation property of normalizing factor)**.** Assume that for all token sequences A and B, for the given normalizing factor D_2 in equation(3) the following is true: $D_2([A;B])-D_2([B])=D_2^*([A])$, i.e., it only depends on A.

This may seem like a very strong assumption, but it is fulfilled by almost all popular linearized transformer models. Below are a few examples:

1. Kernelized attention (Katharopoulos et al., 2020): Set $\phi(\cdot)$ as the representation function of the kernel, set $D_1(Q_i)=\phi(Q_i)$ and $D_2(X)=\sum_{j=1}^M\phi(K_i')$. Potential choices for $\phi(\cdot)$ include but are not limited to:
    (a) In the original paper (Katharopoulos et al., 2020) they propose using a simple scheme $\phi(x)=\mathrm{elu}(x)+1$.
    (b) In the Performers architecture by Choromanski et al. (2020), the PRF (positive random features) and ORF (Orthogonal random features) kernels are learned in the training process.
2. RetNet (Sun et al., 2023): Set $\phi(\cdot)=I(\cdot)$ be the identity function, set $D_1(Q_i)=1$ and $D_2(X)=1$.

4.4. ICL Conversion for Linearized Attention

In this section, we extend the bias conversion method to masked linearized attention layers found in Definition 3.3 with rotary positional encodings (RoPE). Based on Definition 3.3 and the work of Su et al. (2023), kernel methods with RoPE take the form:

$$O_i^T(X) = (R^{d_K}_{\Theta,i}\phi(Q_i))^T \left\{ \frac{1}{D_1(Q_i)^T D_2(X)} \sum_{j=1}^{i} R^{d_K}_{\Theta,j}\phi(K_j)V_j^T \right\}, \tag{12}$$

where $R^{d_K}_{\Theta,i}$ is the rotary matrix defined in the appendix B.1. In the context of positional embeddings, it implies a shift by i positions.

Based on our observations in section 3.1, the logical extension is to include a bias term in the Key-Value matrix. Due to the presence of a normalizing term, we include 2 separate bias terms. The first term b_{KV} is appended to the Key-Value matrix $A = \sum_{j=1}^{i} R^{d_K}_{\Theta,j}\phi(K_j)V_j^T$. The second term b_D is appended to the normalizing term D_2.

With the inclusion of bias terms, linearized transformers with RoPE become:

$$O_i^T(X, b_{KV}, b_D) = (R^{d_K}_{\Theta,i}\phi(Q_i))^T \left\{ \frac{1}{D_1(Q_i)^T(D_2(X) + b_D)} \left[\sum_{j=1}^{i} R^{d_K}_{\Theta,j}\phi(K_j)V_j^T + b_{KV}\right] \right\}. \tag{13}$$

Theorem 4.5. *Given model weights W_V, W_K, W_Q, input prompt X and ICL prompt X', by taking:*

$$b'_{KV} = R^{d_K}_{\Theta,-M} b_{KV} + \sum_{j=1}^{M} R^{d_K}_{\Theta,j-M}\phi(K'_j)V_j'^T, \tag{14}$$

$$b'_D = b_D + D_2^*(X'), \tag{15}$$

then for all X:

$$O_i([X';X], b_{KV}, b_D) = O_i(X, b'_{KV}, b'_D). \tag{16}$$

The proof is fulfilled in the Appendix B.

It is worth highlighting why this makes intuitive sense. The updated bias b'_{KV} involves first a shift of the original bias term b_{KV} by the rotational matrix $R^{d_K}_{\Theta,-M}$, which is the same as a shift left by M tokens. It is equivalent to shifting "past ICL prompts" to the left to include new ones when updating the model.

Apart from the practical applications of Theorem 4.5, we can also arrive at some interesting theoretical conclusions. Typically, people tend to study the attention-score matrix, which is the interaction between the Query and Key matrices. However, our study on linearized attention would seem to suggest a profound connection between the Key and Value matrices. It appears that contextual information is captured within the Key-Value matrix itself.

This yields a whole possible field of research into the interactions between the Key and Value matrix. Rather than the inclusion of a bias term, perhaps there is merit to other approaches such as scaling terms or using MLPs to approximate the Key-Value matrix itself. We don't explore this avenue further in this paper, but it is a fascinating direction for future study.

4.5. ICL Conversion Algorithm for Linearized Attention

In this section, we present the algorithm to include ICL prompts $X' = [X'_1, ..., X'_M]^T$ into a model with linearized attention found in Equation (13). We assume the model has L sub-layers $\{O^{(1)}, ..., O^{(L)}\}$. These can include linearized attention, normalization layers, FFN layers, etc. What's important is except for linearized attention, all the other sub-layers are element-wise operators.

If $O^{(l)}$ is a linearized attention layer, then we define the weights as $W_V^{(l)}, W_K^{(l)}, W_Q^{(l)}, b_{KV}^{(l)}, b_D^{(l)}$. Algorithm 1 takes the form:

Algorithm 1 ICL conversion algorithm (ICLCA)

```
Input: $X'^{(0)} = [X_1'^{(0)}, ..., X_M'^{(0)}]^T$
for $l = 1$ to $L$ do
    if $O^{(l)}$ is not a linearized attention layer then
        return $X'^{(l)} = O^{(l)}(X'^{(l-1)})$
    else if $O^{(l)}$ is a linearized attention layer with weights $W_V^{(l)}, W_K^{(l)}, W_Q^{(l)}, b_{KV}^{(l)}, b_D^{(l)}$ then
        (a) Set $Q'^{(l-1)} = X'^{(l-1)} W_Q^{(l)}$, $K'^{(l-1)} = X'^{(l-1)} W_K^{(l)}$, $V'^{(l-1)} = X'^{(l-1)} W_V^{(l)}$
        (b) Set: $b_{KV}'^{(l)} = R^{d_K}_{\Theta,-M} b_{KV}^{(l)} + \sum_{j=1}^{M} R^{d_K}_{\Theta,j-M}\phi(K_j'^{(l-1)})(V_j'^{(l-1)})^T$
        (c) Set: $b_D'^{(l)} = b_D^{(l)} + D_2^*(X'^{(l-1)})$
        (d) Set:
            $X_i'^{(l)} = \left\{(R^{d_K}_{\Theta,i}\phi(Q_i'^{(l-1)}))^T \left[\sum_{j=1}^{i} R^{d_K}_{\Theta,j}\phi(K_j'^{(l-1)})(V_j'^{(l-1)})^T + b_{KV}'^{(l)}\right]\right\} / \left\{\phi(Q_i'^{(l-1)})^T \left[\sum_{j=1}^{i}\phi(K_j'^{(l-1)} + b_D^{(l)}\right]\right\}$
    end if
end for
Save new bias terms $b_{KV}^{(l)} = b_{KV}'^{(l)}$ and $b_D^{(l)} = b_D'^{(l)}$ for all $l$.
```

5. Approximate Conversion in Regular Attention

While our assumptions cover a large number of efficient transformer architectures, a vast body of work is based on the vanilla attention mechanism without linearization found in Equation (2). We cannot achieve exact conversion of the ICL prompts on such models, but our methods do offer a direct approximation of ICL prompts.

This is done by first approximating the similarity function $\mathrm{sim}(\cdot,\cdot)$ by a kernel and the associated representation function $\phi(\cdot)$. The exact conversion is then completed with this approximation using Theorem 4.5 and the bias is reintroduced to the original model. With the inclusion of RoPE, the resulting output takes the form:

$$O_i^T = \frac{[\sum_{j=1}^{N}\mathrm{sim}(Q_i,K_j)V_j^T] + R_{\Theta,i}^{d_K}\phi(Q_i)^T b_{KV}}{[\sum_{j=1}^{N}\mathrm{sim}(Q_i,K_j)] + R_{\Theta,i}^{d_K}\phi(Q_i)^T b_D}, \quad (17)$$

where

$$b_{KV} = \sum_{j=1}^{M} R_{\Theta,j-M}^{d_K}\phi(K_j')V_j'^T, \quad b_D = \sum_{j=1}^{M}\phi(K_j'). \quad (18)$$

The accuracy of this approximation depends directly on the accuracy of the kernel when approximating $sim(\cdot,\cdot)$. Even now multiple studies are exploring this subject (Choromanski et al., 2020; Zhang et al., 2024) and one can only expect this to get better as time passes.

We include the corresponding algorithm (ICLAA) in Appendix D.1.

6. Experiments

We first demonstrate that our ICL conversion algorithm (ICLCA) is numerically exact for linearized attention models. Next we conduct experiments on a synthetic induction head task and show that ICLCA is able to capture the ICL prompt in the bias terms. Furthermore, we perform experiments on a pretrained GPT-2 model and show that, although the conversion process is approximate, our method reduces the relative error of the logits and the model is aware of the context when generating the text. In this section, we denote the model before conversion with ICL prompt by $M_{\text{old,ICL}}$, the model before conversion without ICL prompt by $M_{\text{old,no ICL}}$ and the model after conversion without ICL prompt by $M_{\text{new,no ICL}}$.

6.1. Effectiveness of Conversion

We demonstrate the effectiveness of our method in section 4.4 on a linear attention transformer with RoPE (Su et al., 2023). We compute the logits of $M_{\text{old,ICL}}$ and $M_{\text{new,no ICL}}$. In Table 1, we show the relative error of the output logits before and after the conversion process for different model sizes. It demonstrates that the conversions are indeed exact up to the numerical rounding errors.

Table 1. Relative error of logits. The relative error is computed on the average of 100 randomly generated ICL prompts and input prompts.

model size	205K	1.99M	19.8M	198M	1.98B
relative error	2.9e-7	4.4e-7	8.3e-7	1.7e-6	4.3e-6

6.2. Induction Head Task

Task We show the application of our method on the following induction head task proposed by Bietti et al. (2023) which demonstrates the in-context learning ability. Consider the bigram data model with *trigger tokens* where the sequences follow a mixed Markov distribution. To be more specific, the vocabulary is all the letters with both upper and lower cases. Out of the alphabet, the trigger tokens are set to be $\{a,b,c,d,e\}$. In each sequence, the tokens form a Markov chain where the transition from non-trigger tokens will be uniformly distributed across the vocabulary while the transition from trigger tokens will be deterministic. For instance, in one sequence, when 'a' first appears, what follows is purely random, but once chosen (say 'x'), the next time we have 'a', we must have the same token ('x') following(we say that the trigger token 'a' has committed to token 'x'). This case is closely related to induction (Elhage et al., 2021). In-context learning on this toy data is very straightforward, i.e., if the context contains committed trigger tokens, then such commitment should be carried over to the rest of the sequence.

Result We use a 12-layer linear attention transformer with embedding dimension=128 and RoPE. We evaluate the in-context learning accuracy which is calculated on every committed trigger token. In Table 2, we show the in context accuracy under three settings and demonstrate that the ICL prompt is exactly captured by ICLCA.

Table 2. In context accuracy of three models.

model	In context accuracy
$M_{\text{old,ICL}}$	99.95%
$M_{\text{old,no ICL}}$	2.23%
$M_{\text{new,no ICL}}$	99.95%

6.3. Effect of the Added Bias Term

In section 4.4, we introduce a new model architecture where we add a bias term in the linearized attention module. To show how this bias term affects the model performance, we train the model of the new architecture from scratch. In Figure 1 we plot the training loss curves and in-context accuracy

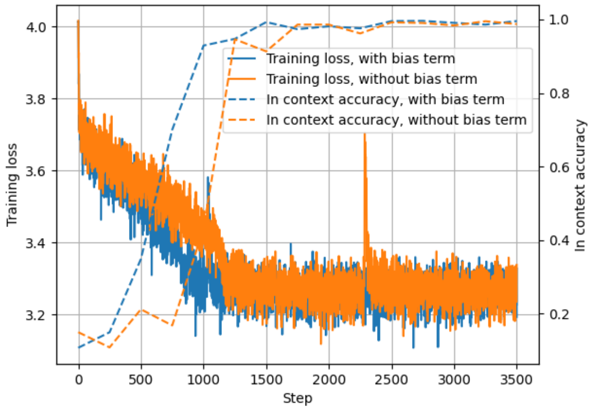

Figure 1. The training loss curves and in context accuracy curve during training.

Table 3. Comparison of generated texts.

ICL prompt: Bill Gates was born at Seattle, Washington.

Model before conversion, with ICL prompt

Input Prompt: Bill Gates was born at Seattle, Washington. He is the CEO of

Generated Text: Bill Gates was born at Seattle, Washington. He is the CEO of Microsoft, the world's largest software company. He is also the co-founder and CEO of Microsoft.com, a leading online marketplace for developers and publishers. He is also the co-founder and CEO of Microsoft's Bing search engine.

Model after conversion, without ICL prompt

Input Prompt: He is the CEO of

Generated Text: He is the CEO of Microsoft, the world's largest company. He is a member of the Microsoft Foundation. He is a member of the Council of Economic Advisers.

Model before conversion, without ICL prompt

Input Prompt: He is the CEO of

Generated Text: He is the CEO of European Cryptocurrency Exchange, where he rapidly explores the various urban applications for cryptocurrencies and cryptocurrencies in development. The Crypto Central Service crew is looking for cash and other cryptocurrencies based on ethics or finance.

curves of models with and without this modification. We see that adding these bias terms makes training slightly faster, which means that our new architecture not only facilitates the ICLCA, but also accomplishes the induction head task better by itself.

6.4. Experiment on Pretrained GPT-2

We have shown the effectiveness of ICLCA on the linearized attention models. However, in practice, most existing models are still based on softmax attention. So we also conduct experiments on the pretrained GPT-2 model. Since ICLCA is only valid for the linear attention model, the conversion process contains two steps: 1. First use the method in Performers (Choromanski et al., 2020) to approximate the softmax attention by linearized attention; 2. Second use ICLAA (details in Appendix D.1) to convert the ICL prompt into the bias terms and include these in the original softmax attention. Since the approximation of the softmax attention is not exact, the conversion process introduces some errors. Before the conversion process, the relative error of the logits between $M_{\text{old,ICL}}$ and $M_{\text{old,no ICL}}$ is 16.56%. After the conversion process, the relative error of the logits between $M_{\text{old,ICL}}$ and $M_{\text{new,no ICL}}$ is 9.17%. The relative error of the logits is improved from 16.56% to 9.17%. In Table 3 we present an example of the generated text from the three models. This example shows that the conversion method can capture the information in the ICL prompt. We present more examples in Appendix E.

7. Transformers as RNNs and Associated Extensions

Based on the work of Katharopoulos et al. (2020), we find that linearized transformers can be expressed in the form of an RNN. This holds true in our setting where Equation (12) becomes:

$$\begin{aligned} s_0 &= 0, \quad s_i = s_{i-1} + R^{d_K}_{\Theta,i}\phi(K_i)V_i \\ z_0 &= 0, \quad z_i = z_{i-1} + D_2^*(X_i) \\ y_i &= \frac{R^{d_K}_{\Theta,i}\phi(Q_i)^T s_i}{D_1(Q_i)^T z_i}. \end{aligned} \tag{19}$$

Within this setting, the addition of our bias terms b_{KV} and b_D are equivalent to changing the initializations of s_0 and z_0 to become: $s_0 = b'_{KV} - b_{KV} = \sum_{i=j}^{M} R^{d_K}_{\Theta,j-M}\phi(K'_j)V'^T_j$ and $z_0 = b'_D - b_D = D_2(X')$. In RNNs, future tokens can only observe the hidden states of previous tokens, hence capturing the hidden states of the ICL prompts is sufficient.

Viewing ICL conversion from the perspective of RNNs further allows us to generalize to modern state-of-the-art architectures which may not be based on the transformer. Recently, RNN-based architectures such as H3 (Fu et al., 2023) and MAMBA (Gu & Dao, 2023) have garnered much popularity with many believing they are the next step in LLMs. While it is yet to be seen whether these models will display emergent abilities such as In-context Learning, if they do, we could extend our ICL conversion method to those autoregressive RNN architectures and capture the ICL prompts in the initialization of the RNN.

8. Conclusion and Discussion

In this work, we leverage the linearity of the attention module in linearized attention architectures to permanently incorporate ICL prompts into the model. This is done through the addition of a bias term to the Key-Value matrix, which can be seen as an architectural modification. Compared to current alternatives, ours is an exact conversion, which is interpretable and very computationally inexpensive to implement.

Exact conversion from context data to model weights not only contributes to the theoretical understanding of ICL but also unlocks new possibilities for LLMs in practice. For instance, LLMs offered by companies are constantly being fine-tuned to incorporate new knowledge, which requires a large amount of data and delicate optimization strategies to avoid catastrophic forgetting. With our method, we can conduct ICL-guided fine-tuning, which can effectively make localized modifications. Furthermore, our method can potentially offer significant computation savings for LLM inference, by converting long context prompts such as system prompts, personalized data, and personal history that are repeatedly added as context for every inference.

When it comes to approximate conversion with standard transformers, being able to add context information in such a cheap way presents new opportunities. While the approximate conversion itself may not be sufficient for real-world applications, it can be used as a potential initialization for existing fine-tuning methods to reduce the number of training steps required.

A legitimate concern is the scalability of our method with model size. In the case of linear attention models, the conversion is exact so the methodology scales without any issues to large models. In Table 1, we show that rounding errors scale well with the number of model parameters. As the model size increases by a factor of 10, the rounding errors increase by approximately a factor of 2. Regarding the approximate softmax conversion method, scalability is an issue. That being said, the scalability depends primarily on whether the kernel function can properly approximate the softmax in pre-trained transformers. If such kernels can scale to larger models our approximate conversion methodology will scale to the same models using that kernel. In fact, in the original works the approximation is done for all tokens. In this work we only approximate softmax for the ICL tokens. Hence the precision of our method should always be better than approximating the full softmax with all the tokens. Recent work such as Hedgehog and Porcupine (Zhang et al., 2024) has shown it is possible to design kernels that scale fairly well to large models such as LlaMa-2. Furthermore, we think it is possible to design kernels specifically for ICL approximation which will perform even better for this specific task. We believe with the correct choice of kernel, our conversion method can scale to large models even for the approximate case.

A natural extension to these findings is to test our approximate conversion method with more sophisticated kernels (Katharopoulos et al., 2020; Zhang et al., 2024). Further experiments can also be conducted to verify how the approximate conversion works in conjunction with existing fine-tuning methods. Another possible line of research is exploring possible methods to achieve exact conversion for transformers with softmax attention.

Impact Statement

This paper presents work whose goal is to advance the field of large language models. By enabling permanent and interpretable incorporation of context into large language models without expensive parameter updates, this approach can make advanced AI more accessible and cost-effective. This democratization can enhance various applications, including education, customer service, and content creation. However, it also raises concerns about potential misuse, such as the reinforcement of biases or misinformation. Therefore, it is crucial to establish ethical guidelines and safeguards to ensure that the benefits of this technology are realized responsibly and equitably.

Acknowledgement

We would like to thank Shuchen Xue, Tianxun Zhou, Tianyi Zhang, Liu Wei, for the helpful discussions and suggestions.

This research is partially supported by the National Research Foundation Singapore under the AI Singapore Programme (AISG Award No: AISG2-TC-2023-010-SGIL) and the Singapore Ministry of Education Academic Research Fund Tier 1 (Award No: T1 251RES2207).

References

Instruct-tune llama on consumer hardware, 2023. URL `https://github.com/tloen/alpaca-lora`.

Abbe, E., Bengio, S., Lotfi, A., and Rizk, K. Generalization on the unseen, logic reasoning and degree curriculum. In *International Conference on Machine Learning*, pp. 31–60. PMLR, 2023.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*, 2022.

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B., DasSarma, N., et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate, 2016.

Bai, Y., Chen, F., Wang, H., Xiong, C., and Mei, S. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *arXiv preprint arXiv:2306.04637*, 2023.

Bietti, A., Cabannes, V., Bouchacourt, D., Jegou, H., and Bottou, L. Birth of a transformer: A memory viewpoint. *arXiv preprint arXiv:2306.00802*, 2023.

Blanc, G. and Rendle, S. Adaptive sampled softmax with kernel based sampling. In *International Conference on Machine Learning*, pp. 590–599. PMLR, 2018.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., et al. Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv preprint arXiv:2303.12712*, 2023.

Choi, E., Jo, Y., Jang, J., and Seo, M. Prompt injection: Parameterization of fixed inputs. *arXiv preprint arXiv:2206.11349*, 2022.

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Chowdhury, S. P., Solomou, A., Dubey, A., and Sachan, M. On learning the transformer kernel. *arXiv preprint arXiv:2110.08323*, 2021.

Dai, D., Dong, L., Hao, Y., Sui, Z., Chang, B., and Wei, F. Knowledge neurons in pretrained transformers. *arXiv preprint arXiv:2104.08696*, 2021.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q. V., and Salakhutdinov, R. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dong, Q., Li, L., Dai, D., Zheng, C., Wu, Z., Chang, B., Sun, X., Xu, J., and Sui, Z. A survey for in-context learning. *arXiv preprint arXiv:2301.00234*, 2022.

Dziri, N., Lu, X., Sclar, M., Li, X. L., Jiang, L., Lin, B. Y., Welleck, S., West, P., Bhagavatula, C., Le Bras, R., et al. Faith and fate: Limits of transformers on compositionality. *Advances in Neural Information Processing Systems*, 36, 2024.

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.

Fu, D. Y., Dao, T., Saab, K. K., Thomas, A. W., Rudra, A., and Ré, C. Hungry hungry hippos: Towards language modeling with state space models, 2023.

Gao, Y., Zheng, C., Xie, E., Shi, H., Hu, T., Li, Y., Ng, M. K., Li, Z., and Liu, Z. On the expressive power of a variant of the looped transformer. *arXiv preprint arXiv:2402.13572*, 2024.

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

Giannou, A., Rajput, S., Sohn, J.-y., Lee, K., Lee, J. D., and Papailiopoulos, D. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pp. 11398–11442. PMLR, 2023.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL `https://doi.org/10.1162/neco.1997.9.8.1735`.

Irie, K., Csordás, R., and Schmidhuber, J. The dual form of neural networks revisited: Connecting test time predictions to training patterns via spotlights of attention. In *International Conference on Machine Learning*, pp. 9639–9659. PMLR, 2022.

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Luong, M.-T., Pham, H., and Manning, C. D. Effective approaches to attention-based neural machine translation, 2015.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022a.

Meng, K., Sharma, A. S., Andonian, A., Belinkov, Y., and Bau, D. Mass-editing memory in a transformer. *arXiv preprint arXiv:2210.07229*, 2022b.

Namazifar, M., Hazarika, D., and Hakkani-Tür, D. Role of bias terms in dot-product attention. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5. IEEE, 2023.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training. 2018.

Rumelhart, D. E., Hinton, G. E., Williams, R. J., et al. Learning internal representations by error propagation, 1985.

Snell, C., Klein, D., and Zhong, R. Learning by distilling context. *arXiv preprint arXiv:2209.15189*, 2022.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, pp. 127063, 2023.

Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., and Wei, F. Retentive network: A successor to transformer for large language models, 2023.

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.

Wang, B. and Komatsuzaki, A. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

Wang, Z., Wang, Y., Zhang, Z., Zhou, Z., Jin, H., Hu, T., Sun, J., Li, Z., Zhang, Y., and Xu, Z.-Q. J. Towards understanding how transformer perform multi-step reasoning with matching operation. *arXiv preprint arXiv:2405.15302*, 2024.

Wei, J., Bosma, M., Zhao, V. Y., Guu, K., Yu, A. W., Lester, B., Du, N., Dai, A. M., and Le, Q. V. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.

Wei, J., Wei, J., Tay, Y., Tran, D., Webson, A., Lu, Y., Chen, X., Liu, H., Huang, D., Zhou, D., et al. Larger language models do in-context learning differently. *arXiv preprint arXiv:2303.03846*, 2023.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.

Zhang, M., Bhatia, K., Kumbong, H., and Ré, C. The hedgehog & the porcupine: Expressive linear attentions with softmax mimicry. *arXiv preprint arXiv:2402.04347*, 2024.

Zheng, C., Li, L., Dong, Q., Fan, Y., Wu, Z., Xu, J., and Chang, B. Can we edit factual knowledge by in-context learning? *arXiv preprint arXiv:2305.12740*, 2023.

Zhou, D., Schärli, N., Hou, L., Wei, J., Scales, N., Wang, X., Schuurmans, D., Cui, C., Bousquet, O., Le, Q., et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

Appendix

A. Naive Attempt at Direct Conversion

To construct the equivalence in equation (7) additional assumptions and properties have to be included. Firstly, the term X^TX does not include the entire matrix space. All matrices of the form X^TX are positive semi-definite. Hence, we can limit the matrix space to only matrices of the form $U^T\Sigma U$ where U^T,U are orthogonal matrices and Σ is a diagonal matrix where each term on the diagonal is the eigenvalue.

Such a space still contains the zero vector. Equation (8) shows that the equivalence does not accommodate for the zero vector. Another possible assumption is for the token set to belong to a shifted subspace. i.e. For all inputs X, we have:

$$X^TX \in c+D. \tag{20}$$

Here c is a shift vector that satisfies equation (8), i.e.

$$(W_1^T \otimes W_2^T)vec(c) = (W_V^T \otimes W_K^T)vec(c) + (W_V^T \otimes W_K^T)vec(X'^TX'). \tag{21}$$

At the same time we need $D \subset ker(W_1^T \otimes W_2^T - W_V^T \otimes W_K^T)$.

There are 2 key problems with such a setting. First of all, if W_1,W_2 exists to satisfy such a scenario, there is no guarantee for the uniqueness. In fact, it is likely not to be unique and there is no clear intuitive choice. Furthermore, c depends not on X' but rather on X. That would mean that the matrix space assumption would vary depending on X. This is very difficult because the equality must be true for all X. Hence we choose to consider the addition of free parameters rather than attempt a direct solution.

B. Proof of Theorem 4.5

Proof. From the original paper introducing RoPE (Su et al., 2023), the rotational matrix is defined as:

Definition B.1 (Rotary matrix)**.** A rotary matrix with pre-defined parameters $\Theta = \{\theta_i = 10000^{-2(i-1)/d}, i \in [1,...,d/2]\}$ and shift m is defined as:

$$R^d_{\Theta,m} = \begin{pmatrix} \cos m\theta_1 & -\sin m\theta_1 & 0 & \cdots & 0 & 0 & 0 \\ \sin m\theta_1 & \cos m\theta_1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \cos m\theta_2 & -\sin m\theta_2 & \cdots & 0 & 0 \\ 0 & 0 & \sin m\theta_2 & \cos m\theta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & \cos m\theta_{d/2} & -\sin m\theta_{d/2} \\ 0 & 0 & 0 & 0 & \cdots & \sin m\theta_{d/2} & \cos m\theta_{d/2}. \end{pmatrix} \tag{22}$$

As one would expect with rotational matrices, we have:

$$R^d_{\Theta,a+b} = R^d_{\Theta,a} + R^d_{\Theta,b}, \tag{23}$$

and

$$R^{dT}_{\Theta,m} = R^d_{\Theta,-m}. \tag{24}$$

$$\begin{aligned} O_i^T([X',X],b_1,b_2) &= \frac{(R^{d_K}_{\Theta,i+M}\phi(Q_i))^T[\sum_{j=1}^i (R^{d_K}_{\Theta,j+M}\phi(K_j))V_j^T + \sum_{j=1}^M (R^{d_K}_{\Theta,j}\phi(K'_j))V_j'^T + b_1]}{D_i([X';X]) + b_2} \\ &= \frac{\phi(Q_i)^T[\sum_{j=1}^i R^{d_K}_{\Theta,i}{}^T(R^{d_K}_{\Theta,j}\phi(K_j)V_j^T + \sum_{j=1}^M R^{d_K}_{\Theta,i}{}^T(R^{d_K}_{\Theta,j-M}\phi(K'_j))V_j'^T + R^{d_K}_{\Theta,i}{}^T R^{d_K}_{\Theta,-M}b_1]}{D_i(X) + D_i^*(X') + b_2} \\ &= \frac{(R^{d_K}_{\Theta,i}\phi(Q_i))^T[\sum_{j=1}^i (R^{d_K}_{\Theta,j}\phi(K_j)V_j^T + \sum_{j=1}^M (R^{d_K}_{\Theta,j-M}\phi(K'_j))V_j'^T + R^{d_K}_{\Theta,-M}b_1]}{D_i(X) + b'_2} \\ &= O_i^T(X,b'_1,b'_2). \end{aligned} \tag{25}$$

□

C. Analysis of Other Choices of Free Parameters

In this section, we consider other options for free parameters within a standard transformer with stacked decoders. Free parameters can be included in 3 main sections. The input, attention module, and feed-forward network. To begin with, we can modify the input to the Attention Module. Since GPT networks and many other LLM architectures are stacked layers of decoders, all of the information included by ICL tokens could technically be captured in an additional encoding layer before the input and transformer layers. There are interesting approaches we considered, such as a modification of the current compressor architecture in popular models such as transformers-XL(Dai et al., 2019), effectively compressing the ICL inputs and using it as an initialization. However, these seem like fanciful ways to append the ICL prompts to the start of all future inputs

An alternative is to directly modify the FFN. This could potentially be very interesting because past work on ROME (Meng et al., 2022a) has demonstrated that factual associations within GPT networks can be located in the MLP/FFN module of the attention process. Instead of adding bias terms to the numerator and denominator in theorem 4.5, we could attempt to induce a similar change within the FFN. While this would be architecturally clean, an exact conversion could still be a challenge.

In this study, we opted to focus on the addition of free parameters to the attention module. Apart from a bias term, there were a few other methods considered. First of all, is a scaling term. However, since the input tokens can be similarly scaled, that shouldn't help with conversion. Another attempt was to include starting/initialization tokens within the key value matrix, however this would just give the same result as the bias term but with more computational cost.

A possible option to look at in future studies is to include the full FFN within the Key-Value matrix itself. This marks a clear departure from the current transformer architecture but would be consistent with our finding that the Key-Value matrix bears special significance. However, such claims will require detailed experiments for verification.

D. Extension to Other Architectures

In this paper, we choose to consider different modifications of the typical stacked transformer architecture. A future alternative to this would be GPT-J (Wang & Komatsuzaki, 2021), which places the attention module and feed-forward network in tandem rather than one after the other.

GPT-J hasn't seen widespread implementation because, while it matches the typical stacked architecture in performance, it fails to exceed it. However, it provides a very natural structure for ICL conversion. Rather than include a whole new bias term to the attention module, one can add that term to the FFN itself. i.e. for each token **q**, rather than just taking $FFN(\mathbf{q})$, we would take $FFN(\mathbf{q})+\mathbf{q}\sum_{j=1}^{M}R^{d_K}_{\Theta,j-M}\phi(K'_j)^T V'^T_j$.

There are a few problems to address before such an approach becomes viable. First of all, GPT-J uses the traditional non-linearized attention with softmax. There has yet to be a detailed study into how it would behave in a linearized setting which is a prerequisite to our conversion method. More importantly, any linear setting introduced would have to have a normalizing term $D(\cdot)=1$ like in RetNet. While RetNet has shown great promise so far, it may not necessarily be the future of transformer networks. Hence being limited to cases with $D(\cdot)=1$ is undesirable.

D.1. ICL Approximation Algorithm for Regular Attention

In this section we present the algorithm to approximately include ICL prompts $X'=[X'_1,...,X'_M]^T$ into a model with regular attention found in equation (1). We assume the model has L sub-layers $\{O^{(1)},...,O^{(L)}\}$. These can include regular attention, normalization layers, FFN layers, etc. Except for regular attention, all other sub-layers are element-wise operators.

If $O^{(l)}$ is a regular attention layer, then we define its weights as $W_V^{(l)},W_K^{(l)},W_Q^{(l)}$. Algorithm 2 takes the form:

Algorithm 2 ICL approximation algorithm (ICLAA)

```
Input: $X'^{(0)} = [X_1'^{(0)}, \dots, X_M'^{(0)}]^T$.
for $l = 1$ to $L$ do
    if $O^{(l)}$ is not a regular attention layer then
        return $X'^{(l)} = O^{(l)}(X'^{(l-1)})$
    else if $O^{(l)}$ is a regular attention layer with weights $W_V^{(l)}, W_K^{(l)}, W_Q^{(l)}$ then
        (a) Set $Q'^{(l-1)} = X'^{(l-1)} W_Q^{(l)}$, $K'^{(l-1)} = X'^{(l-1)} W_K^{(l)}$, $V'^{(l-1)} = X'^{(l-1)} W_V^{(l)}$;
        (b) Set $b_{KV}^{(l)} = \sum_{j=1}^{M} R_{\Theta,j-M}^{d_K} \phi(K_j'^{(l-1)})(V_j'^{(l-1)})^T$;
        (c) Set $b_D^{(l)} = \sum_{j=1}^{M} R_{\Theta,j-M}^{d_K} \phi(K_j'^{(l-1)})$;
        (d) Set $x_i'^{(l)} = \left\{ \sum_{j=1}^{i} \mathrm{sim}(R_{\Theta,i}^{d_K} Q_i'^{(l-1)}, R_{\Theta,j}^{d_K} K_j'^{(l-1)}) V_j'^{(l-1)\,T} \right\} / \left\{ \sum_{j=1}^{i} \mathrm{sim}(Q_i'^{(l-1)}, K_j'^{(l-1)}) \right\}$.
    end if
end for
Save new bias terms $b_{KV}^{(l)}$ and $b_D^{(l)}$ for all $l$.
```

E. More Examples of GPT-2 Generated Text

In this section, we present more examples of GPT-2 generated text before and after conversion.

Table 4. Comparison of generated texts before and after conversion

ICL prompt: Los Angeles is the second most populous city in the United States.

Model before conversion, with ICL prompt

Input Prompt: Los Angeles is the second-most populous city in the United States. It is located in the U.S. state of

Generated Text: Los Angeles is the second-most populous city in the United States. It is located in the U.S. state of California, which is home to the largest number of residents of any U.S. state.

Model after conversion, without ICL prompt

Input Prompt: It is located in the U.S. state of

Generated Text: It is located in the U.S. state of California. The city of Los Angeles is the second most populous city in the United States, with a population of 1.5 million people.

Model before conversion, without ICL prompt

Input Prompt: It is located in the U.S. state of

Generated Text: It is located in the U.S. state of Texas. The State of Texas has a population of about 3,000,000.

Table 5. Comparison of generated texts before and after conversion

ICL prompt: Steve Jobs founded Apple.

Model before conversion, with ICL prompt

Input Prompt: Steve Jobs founded Apple. He is the founder of

Generated Text: Steve Jobs founded Apple. He is the founder of Apple, Inc., a leading technology company that has been around for over 40 years. He is also the founder of the Apple logo.

Model after conversion, without ICL prompt

Input Prompt: He is the founder of

Generated Text: He is the founder of the company that created the iPhone. Apple has been a major player in the tech industry for over a decade. Jobs was a pioneer in the Apple Watch. Jobs was a former CEO of Apple.

Model before conversion, without ICL prompt

Input Prompt: He is the founder of

Generated Text: He is the founder of the New York-based group, which has been a vocal critic of the Obama administration's handling of the Syrian conflict. "This is not a war of words," he said. "This is a war of words."