IN-CONTEXT ALGORITHM EMULATION IN FIXED-WEIGHT TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We prove that a minimal Transformer with frozen weights emulates a broad class of algorithms by in-context prompting. We formalize two modes of in-context algorithm emulation. In the *task-specific mode*, for any continuous function $f : \mathbb{R} \to \mathbb{R}$, we show the existence of a single-head softmax attention layer whose forward pass reproduces functions of the form $f(w^\top x - y)$ to arbitrary precision. This general template subsumes many popular machine learning algorithms (e.g., gradient descent, linear regression, ridge regression). In the *prompt-programmable mode*, we prove universality: a single fixed-weight two-layer softmax attention module emulates all algorithms from the task-specific class (i.e., each implementable by a single softmax attention) via only prompting. Our key idea is to construct prompts that encode an algorithm's parameters into token representations, creating sharp dot-product gaps that force the softmax attention to follow the intended computation. This construction requires no feed-forward layers and no parameter updates. All adaptation happens through the prompt alone. Numerical results corroborate our theory. These findings forge a direct link between in-context learning and algorithmic emulation, and offer a simple mechanism for large Transformers to serve as prompt-programmable interpreters of algorithms. They illuminate how GPT-style foundation models may swap algorithms via prompts alone, and establish a form of algorithmic universality in modern Transformer models.

1 INTRODUCTION

We show that a minimal Transformer architecture with frozen weights is capable of emulating a broad class of algorithms through prompt design alone. This stylized problem setting isolates the core of in-context computation and provides an analytic lens on fundamental questions in Transformer models: How do fixed-weight models execute diverse tasks from context alone? How does a prompt turn into an algorithmic procedure? How do prompt-encoded parameters and query-key routing realize task identification and stepwise execution? What minimal architectural ingredients suffice for general in-context capability? As foundation models rise to prominence in modern AI (Bommasani, 2021), these questions are central, since much of their practical utility comes from in-context learning (prompting) rather than explicit retraining (Brown et al., 2020; Liu et al., 2023). Against this backdrop, this work offers a rigorous basis for in-context *task* learning[1], supplies a simple mechanism for Transformers to act as prompt-programmable algorithm libraries, and shows how GPT-style models may swap algorithms via prompts alone, shedding light on their general-purpose capabilities.

Large Transformer models exhibit ability to adapt to a new task by conditioning on examples or instructions provided in the prompt without any gradient updates. This capability is known as In-Context Learning (ICL) (Min et al., 2022; Brown et al., 2020). Prior work on Transformer in-context learning falls into two strands. One trains models that learn in context for a specific function class (Garg et al., 2022; Akyürek et al., 2023; Li et al., 2023; Ahn et al., 2023; Zhang et al., 2024). The other hand-engineers Transformers to enact particular algorithms with fixed weights (Bai et al., 2023; Von Oswald et al., 2023; Wu et al., 2025). In particular, Bai et al. (2023) demonstrate that *task-specific* attention layers — attention mechanisms with weights designed for a given task — implement a

[1] We use "task" to highlight algorithm-level adaptation (to diverse tasks), not mere pattern completion.

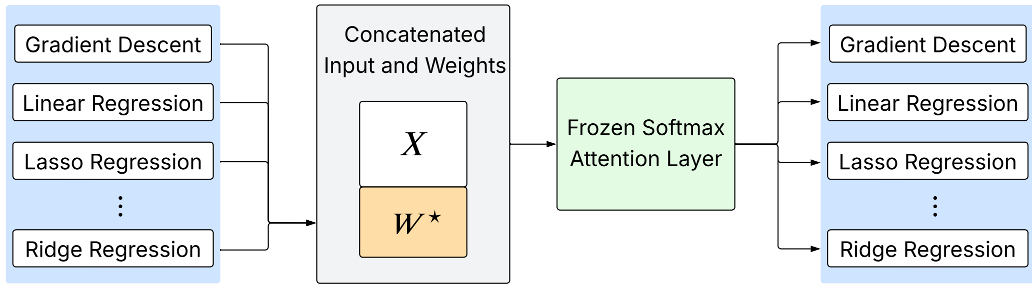

Figure 1: **Prompt-Programmable In-Context Algorithm Emulation Overview.** X denotes the data input, and $W^\star$ encodes the instructions of the algorithm we aim to emulate. We show that even a 2-layer softmax attention module suffices to emulate a broad class of algorithms by changing prompt (Theorem 4.1), i.e., the $W^\star$ in the prompt. This separates *algorithm information* (in the prompt) from "model parameters" (frozen). By sending the algorithm-specific information (e.g., instructions + data) to a fixed-weight model, the *prompt* acts as the *program* and the *frozen transformer* as the *interpreter*. This makes the "weights-as-data" mechanism explicit and is the core mechanism of prompt-programmability: a minimal frozen Transformer serves as a modular interface in which swapping the prompt swaps the algorithm with no retraining.

variety of algorithms without gradient updates. For example, a single Transformer with fixed, task-tailored attention weights achieves near-optimal performance on algorithms such as least-squares regression, ridge regression, lasso, and gradient descent (Bai et al., 2023; Wu et al., 2025). These results suggest that Transformers are capable of *in-context algorithm emulation*. Yet these approaches retrain per task or hard-wire per algorithm. They do not give a single fixed architecture that is prompt-programmable across many algorithms with explicit guarantees and minimal components.

To combat this, we advance this line of research by omitting the need for designing a new Transformer block for every algorithm. We propose a frozen Transformer architecture to emulate a library of attention-based algorithms in context without weight updates. We achieve this by embedding algorithm-specific information into input prompts. Specifically, we formalize two emulation modes, and establish explicit guarantees and constructive minimal designs for both. In the *task-specific* mode (Section 3), a dedicated attention module with fixed weights (single- or multi-head) executes one algorithm in context. In the *prompt-programmable* mode (Section 4), by contrast, a single Transformer module with fixed weights re-programs itself through different prompts to execute multiple algorithms on the fly. These constructions yield universality and minimality results for in-context algorithm emulation. Specifically, we demonstrate a minimalist model of internal algorithm swapping, where prompts serve as the context carrying algorithmic instructions.

Contributions. We establish a new form of in-context learning universality *for algorithm emulation*, limited to *attention-implementable algorithms*. Our contributions are four-fold:

- **Task-Specific Emulation of $f(w^\top x - y)x$.** A single-head, single-layer softmax attention with a linear map universally approximates functions of the form $f(w^\top x - y)x$ for any continuous f, with frozen weights and a suitable prompt. This general result subsumes, for example, computing per-sample gradients and performing gradient descent updates (by choosing f as a loss derivative), as well as solving linear and ridge regression in one forward pass.
- **Constructive, Interpretable Prompt Design for Algorithm Emulation.** We give an explicit prompt design strategy that encodes the target task's parameters and induces large query-key margins so softmax follows the intended pattern, furnishing an interpretable, verifiable recipe for prompt-programming a fixed attention-only module.
- **A Simple Mechanism for Internal Algorithm Swapping of Transformer Models.** Changing only the prompt-encoded algorithm weights swaps the algorithm executed by the fixed attention-only module, without retraining. Theory (finite libraries) and experiments (e.g., Lasso, ridge, linear regression) confirm high-fidelity swapping. Altogether, these results shed light on the general-purpose capability of GPT-style Transformer models to select and swap internal routines via prompts (our formal proofs concern attention-only modules).

In conclusion, we show a minimalist transformer architecture serve as a general-purpose algorithm emulator in context through prompt design. Our findings contribute to a sharp theoretical foundation for viewing in-context learning as in-context algorithm emulation. They suggest that large pretrained softmax attention models (such as GPT-style Transformers) encode a library of algorithms, and swap among them based on prompts. This is achieved within a unified attention architecture and without any parameter updates. We believe this perspective opens new opportunities for understanding the emulation ability of Transformer models.

Organization. Section 2 presents ideas we build on. Section 3 presents illustrative examples of learning statistical models in-context with *task-specific* attention heads. Section 4 presents our main results. Appendix A presents our proof strategies. Section 5 presents numerical validations.

Related Work. Due to page limits, we defer related work discussions to Appendix B.

Notations. We denote the index set $\{1, \ldots, I\}$ by $[I]$. We use lowercase letters for vectors and uppercase letters for matrices. The vector $e_j^{(n)} \in \mathbb{R}^n$ denotes the one-hot vector with 1 in the j-th position and 0 elsewhere. We write $X \in \mathbb{R}^{d \times n}$ for the input sequence, where d is the token dimension and n is the sequence length. We denote the number of attention heads by H. We use $\|\cdot\|_\infty$ and $\|\cdot\|_2$ for the vector ∞-norm and 2-norm, respectively.

2 Preliminaries: Attention, In-Context Learning and Emulation

Softmax Attention. We define a multi-layer self-attention layer with softmax activation as follows.

Definition 2.1 (Softmax Attention Layer). For any input sequence $X \in \mathbb{R}^{d \times n}$, the multi-head attention output (with H heads) is

$$\mathrm{Attn}_m(X) = \sum_{h=1}^{H} \underbrace{W_V^{(h)} X}_{d_o \times n} \mathrm{Softmax}(\underbrace{(W_K^{(h)} X)^\top W_Q^{(h)} X}_{n \times n}) \underbrace{W_O^{(h)}}_{n \times n_o} \in \mathbb{R}^{d_o \times n_o},$$

where $W_K^{(h)}, W_Q^{(h)} \in \mathbb{R}^{d_h \times d}$, $W_V^{(h)} \in \mathbb{R}^{d_o \times d}$, and $W_O^{(h)} \in \mathbb{R}^{n \times n_o}$ for $h \in [H]$. We use Attn_s to denote *single-head* self-attention.

Following the notation of (Hu et al., 2025a), we pick non-identical dimensions for weight matrices W_K, W_Q, W_V for generality of our analysis.

In the common $K := W_K X$, $Q := W_Q X$, $V := W_V X$ notation, a single-layer softmax attention takes a set of key vectors $K = \{k_1, \ldots, k_n\}$, value vectors $V = \{v_1, \ldots, v_n\}$, and a query vector q, to produce an output as a weighted sum of the value vectors. The weights on v_i is $\mathrm{Softmax}(k_i^\top q)$, emphasizing values whose keys are most similar to the query. That is, the softmax attention uses the query as a cue to retrieve the most relevant information from the values (via their keys).

Linear Transformation Layer $\mathrm{Linear}(\cdot)$. Throughout this paper, we sometimes compose attention with an additional linear mapping for flexibility. Such a linear transformation layer uses learned parameters to increase expressivity in attention-based constructions.

Definition 2.2. Let $Z = [z_1, \ldots, z_n] \in \mathbb{R}^{d \times n}$ be the input sequence with columns $z_i \in \mathbb{R}^d$. We use $\mathrm{Linear} : \mathbb{R}^{d \times n} \to \mathbb{R}^{p \times m}$ (for some output length m) to denote *column-wise* linear affine maps. Each output column depends only on one input column, possibly with replication or an additive bias. We write Linear when dimensions are clear (input/output shapes chosen to match attention).

This layer is a generic *column-wise affine* operator. It preprocesses the input to an attention mechanism or post-processes its output. For example, $\mathrm{Attn}_s \circ \mathrm{Linear}(Z)$ applies a per-token affine projection (optionally with replication, so $m \neq n$) before single-head attention. It subsumes the practical per-token linear layer as the special case $m = n$ with shared parameters and optional bias: $\mathrm{Linear}(Z) = AZ + b\mathbb{1}_n^\top \in \mathbb{R}^{p \times n}$ with $A \in \mathbb{R}^{p \times d}$, $b \in \mathbb{R}^p$ and $\mathbb{1}_n$ the all-ones vector. In all cases, columns are processed independently (no cross-column mixing).

In-Context Learning Setup. In in-context learning, a fixed model (e.g., a pretrained Transformer) performs a new task without parameter updates. Formally, the model aims to approximate an unknown function $f : X \to Y$ given a few examples of f in the input prompt. At inference, we provide n exemplar pairs and a query x_q, and concatenate them into a single sequence

$$X := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{(d+1) \times n} \quad \text{and} \quad x_q \in \mathbb{R}^{d \times 1}. \tag{2.1}$$

Namely, the model receives (X, x_q) as the input prompt. The goal of ICL is for the model, given input prompt (X, x_q), to (i) infer f from the exemplars and (ii) apply it to x_q to predict $y_q = f(x_q)$. All the learning happens in the forward pass through the sequence X in an implicit fashion.

Task-Specific Attention. Task-specific attention uses fixed parameters to carry out a particular task when the prompt follows the required structure (see (Bai et al., 2023) for examples.)

Definition 2.3. An attention layer is *task-specific* if there exists a prompt family $\mathcal{P}$ such that, for any prompt $P \in \mathcal{P}$ constructed from task parameters/data, the attention's forward pass implements the task's mapping on the query token(s), with no parameter change.

In particular, we embed the task's defining transformations (e.g. a linear mapping corresponding to f or part of f) into the attention weight matrices. Given a well-formed prompt of exemplar and query tokens, the attention selects and combines these tokens to compute the correct output. Effectively, this allows an attention layer to approximate diverse functions in context without weight updates.

Terminology: Task-Specific vs. Prompt-Programmable In-Context Emulation. In-context algorithm emulation refers to executing an algorithm through a forward pass without weight updates. The core contribution of this work is to formalize two in-context modes and study their scope:

- **Task-Specific In-Context Emulation**: for each algorithm $\mathcal{A}$, there exists an attention module (possibly multi-head) whose forward pass on a well-formed prompt implements $\mathcal{A}$ on the query token(s). Each algorithm therefore requires its own dedicated parameters.
- **Prompt-Programmable In-Context Emulation (via single frozen module)**: there exists a single attention module with fixed weights $\mathrm{Attn}^\star$ such that, for every $\mathcal{A}$ in a target class, a suitable prompt $P_{\mathcal{A}}$ makes $\mathrm{Attn}^\star$ implement $\mathcal{A}$ on the query token(s). All adaptation occurs through the prompt rather than through weight changes. Namely, one $\mathrm{Attn}^\star$ implements a library of algorithms.

These modes are complementary: the first reflects the conventional dedicated-module view (e.g., (Bai et al., 2023)), while the second is stronger — one *fixed-weight attention module* emulates many algorithms via prompts (our contribution). In the remainder of the paper, Section 3 develops the task-specific case. Section 4 establishes the prompt-programmable case by showing how the latter *subsumes* the former via in-context simulation of task-specific modules.

3 TASK-SPECIFIC IN-CONTEXT ALGORITHM EMULATION

We present multiple examples demonstrating how softmax attention modules mimic behaviors of various learning algorithms including gradient descent and linear regression. We begin with a very general result showing that even a single-layer, single-head attention mechanism is a universal approximator for a broad class of functions defined on the prompt.

In-Context Universal Approximation of $f(w^\top x - y)x$. Let $x \in \mathbb{R}^d$, $y \in \mathbb{R}$, $w \in \mathbb{R}^d$, and let $f : \mathbb{R} \to \mathbb{R}$ be continuous. We consider functions of the form $f(w^\top x - y)x$, where f acts on the residual $w^\top x - y$. This template is very general: many learning rules for linear models take this form, including many residual/gradient-style updates[2]. Hence $f(w^\top x - y)x$ subsumes a wide family of residual-driven updates central to machine learning. Thus, their in-context realization explains much of in-context learning. To this end, showing that attention is capable of emulating any continuous $f(w^\top x - y)x$ indicates a powerful and general capability. It means the attention module implements any continuous adjustment or mapping based on the prediction $w^\top x$ and the label y. The next theorem shows how a single-head attention approximates $[f(w^\top x_i - y_i)x_i]_{i=1}^n$ arbitrarily well.

Theorem 3.1 (In-Context Emulation of $f(w^\top x - y)x$ with Single-Head Attention)**.** Let $[L_{\min}, L_{\max}]$ be a bounded interval containing all values of $w^\top x - y$, and let

$$X := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{(d+1)\times n} \quad \text{and} \quad W := [w \quad w \quad \cdots \quad w] \in \mathbb{R}^{d\times n},$$

[2]For example, $f(t) = t$ corresponds to the raw residual $(w^\top x - y)x$, $f(\cdot) = \nabla_w \ell(\cdot)$ corresponds to per-sample gradients $\nabla_w \ell(w^\top x - y)x$ linear regression or classification with loss $\ell(\cdot)$, and nonlinear f (sigmoid, step, etc.) corresponds to perceptron updates or other error-correcting rules.

where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and $w \in \mathbb{R}^d$ is the coefficient vector. Define the input as:

$$Z := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ w & w & \cdots & w \end{bmatrix} = \begin{bmatrix} X \\ W \end{bmatrix} \in \mathbb{R}^{(2d+1)\times n}. \tag{3.1}$$

Assume $\max\{\|X\|_\infty, \|W\|_\infty\} \le B$. For any continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ and any $\epsilon > 0$, there exists a single-head attention Attn_s with a linear layer Linear such that

$$\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z) - \left[f(w^\top x_1 - y_1)x_1 \quad \cdots \quad f(w^\top x_n - y_n)x_n\right]\|_\infty \le \epsilon, \quad \text{for any} \quad \epsilon > 0.$$

Proof. See Appendix D.1 for a detailed proof. □

Theorem 3.1 establishes that even the simplest softmax attention alone suffices to encode any continuous function of the form $f(w^\top x - y)x$ by incorporating weights in the prompt. A direct implication is by replacing f with the derivatives of differentiable loss function as follows.

Example 1: In-Context Emulation of Single-Step GD. Building on Theorem 3.1, we show that a softmax attention layer emulates Gradient Descent (GD) in-context. Fristly, we replace the continuous function $f(\cdot)$ in Theorem 3.1 with $\nabla\ell(\cdot)$, where $\ell : \mathbb{R} \to \mathbb{R}$ is any differentiable loss function. We show that the softmax attention emulates *per-sample gradients* in context.

Corollary 3.1.1 (In-Context Emulation of Per-Sample Gradients)**.** Let $\ell : \mathbb{R} \to \mathbb{R}$ be differentiable and $\ell' : \mathbb{R} \to \mathbb{R}$ for its scalar derivative, $\ell'(t) = \frac{d}{dt}\ell(t)$. For $z := w^\top x - y$ with $x \in \mathbb{R}^d$, $y \in \mathbb{R}$, $w \in \mathbb{R}^d$, denote $\nabla_w \ell(z) := \ell'(z)$. Set $f(\cdot) = \ell'(\cdot)$ in Theorem 3.1. With $Z = [X; W]$ as in (3.1), for any $\epsilon > 0$, there exist a single-head attention $\mathrm{Attn}_s(\cdot)$ and a linear map $\mathrm{Linear}(\cdot)$ such that,

$$\| \underbrace{\mathrm{Attn}_s \circ \mathrm{Linear}(Z)}_{=:\widehat{G} \text{ approximated per-sample gradient matrix}} - \underbrace{[\ell'(w^\top x_1 - y_1)x_1, \cdots, \ell'(w^\top x_n - y_n)x_n]}_{=:G \text{ target per-sample gradient matrix}} \|_\infty \le \epsilon.$$

Corollary 3.1.1 shows that a single-layer single-head softmax attention with a linear map approximates the individual (per-sample) gradient terms $\{\ell'(w^\top x_i - y_i)x_i\}_{i=1}^n$. Moreover, the layer outputs all per-sample gradient terms in parallel. Next, we extend Corollary 3.1.1 to show that a fixed attention layer implements the full gradient-descent update across all samples in-context.

Aggregating the per-sample gradients gives one GD step

$$\widehat{L}_n(w) := \frac{1}{n}\sum_{i=1}^n \ell(w^\top x_i - y_i), \quad \nabla \widehat{L}_n(w) = \frac{1}{n}\sum_{i=1}^n \ell'(w^\top x_i - y_i)x_i =: g.$$

From Corollary 3.1.1, let $\widehat{G}$ be the attention output and choose the readout $u := \frac{1}{n}\mathbf{1}_n \in \mathbb{R}^n$ (equivalently, right-multiply by $W_O = u$ in Definition 2.1). Define the attention estimate of the average gradient as $\widehat{g} := \widehat{G}u$. Then $\widehat{g} \approx g$, and the target update is $w_{\mathrm{GD}}^+ := w - \eta\nabla\widehat{L}_n(w)$. Feeding w in the prompt and applying the same readout produces a single d-dimensional update vector from the layer. The next corollary states the precise approximation guarantee.

Corollary 3.1.2 (In-Context Emulation of a Single GD Step)**.** Let $\ell : \mathbb{R} \to \mathbb{R}$ be differentiable and define $\widehat{L}_n(w) := \frac{1}{n}\sum_{i=1}^n \ell(w^\top x_i - y_i)$. For any step size $\eta > 0$ and any $\epsilon > 0$, there exist a single-head attention Attn_s and a linear map Linear such that, with $Z = [X; W]$ as in (3.1), choosing the readout $u := \frac{1}{n}\mathbf{1}_n$ (equivalently, right-multiply by $W_O = u$ in Definition 2.1), we have

$$\widehat{w}_{\mathrm{GD}} := (\mathrm{Attn}_s \circ \mathrm{Linear}(Z))u \in \mathbb{R}^d \quad \text{and} \quad \|\widehat{w}_{\mathrm{GD}} - \underbrace{(w - \eta\nabla\widehat{L}_n(w))}_{w_{\mathrm{GD}}^+}\|_\infty \le \epsilon.$$

Proof. See Appendix D.2 for a detailed proof. □

Corollary 3.1.2 shows that a single-layer, single-head softmax attention with a linear map aggregates the per-sample gradients via the output projection. It produces a d-vector $\widehat{w}_{\mathrm{GD}}$ that approximates the GD update $w_{\mathrm{GD}}^+ = w - \eta\nabla\widehat{L}_n(w)$. Notably, each output column encodes a copy of w together with a scaled per-sample gradient term. Averaging via the readout $u = \frac{1}{n}\mathbf{1}_n$ then recovers w_{GD}^+ up to ϵ.

Example 2: In-Context Emulation of Multi-Step GD. We extend the single-step construction to show that a multi-layer softmax attention network emulates multi-step gradient descent. In particular, an $(L+1)$-layer transformer approximates L steps of gradient descent.

Stack $(L+1)$ copies of the single-head layer from Corollary 3.1.2. At layer t $(0 \leq t < L)$, use the readout $u^{(t)} = \frac{1}{n}\mathbf{1}_n$ and the prompt $Z^{(t)} = [X; W^{(t)}]$ with $W^{(t)} := [\widehat{w}_{\text{GD}}^{(t)} \cdots \widehat{w}_{\text{GD}}^{(t)}]$. Define

$$\widehat{w}_{\text{GD}}^{(0)} := w, \quad \text{and} \quad \widehat{w}_{\text{GD}}^{(t+1)} := \text{Attn}_s \circ \text{Linear}(Z^{(t)})u^{(t)}.$$

For the target iterates, set $w_{\text{GD}}^{(0)} = w$ and $w_{\text{GD}}^{(t+1)} = w_{\text{GD}}^{(t)} - \eta \nabla \widehat{L}_n(w_{\text{GD}}^{(t)})$. By Corollary 3.1.2, Lemma D.4 and $\|\cdot\|_\infty \leq \|\cdot\|_2$, we arrive

$$\|\widehat{w}_{\text{GD}}^{(t)} - w_{\text{GD}}^{(t)}\|_\infty \leq t\epsilon, \quad t \in [L].$$

Example 3: In-Context Emulation of Linear Regression. We now present the construction for squared loss. We show that a single-layer softmax attention emulates linear regression in-context.

Corollary 3.1.3 (In-Context Emulation of Linear Regression)**.** For any dataset $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$ and any $\epsilon > 0$, there exist a single-head attention Attn_s, a linear map Linear, and a readout $u \in \mathbb{R}^n$ such that, with $Z = [X; W]$ as in (3.1) (for any fixed bounded w),

$$\widehat{w}_{\text{linear}} := (\text{Attn}_s \circ \text{Linear}(Z))u \in \mathbb{R}^d, \quad \text{and} \quad \|\widehat{w}_{\text{linear}} - w_{\text{linear}}\|_\infty \leq \epsilon,$$

where $w_{\text{linear}} := \text{argmin}_{w \in \mathbb{R}^d} \frac{1}{2n}\sum_{i=1}^n (\langle w, x_i\rangle - y_i)^2$.

Proof. See Appendix D.3 for detailed proof. □

Example 4: In-Context Emulation of Ridge Regression. We add regularization term and show that a single-layer softmax attention emulates ridge regression with L_2 penalty.

Corollary 3.1.4 (In-Context Emulation of Ridge Regression)**.** For any dataset $\{(x_i, y_i)\}_{i=1}^n$, any $\lambda \geq 0$, and any $\epsilon > 0$, there exist a single-head attention Attn_s, a linear map Linear, and a readout $u \in \mathbb{R}^n$ such that, with $Z = [X; W]$ as in (3.1) (and the regularization signal included in the prompt),

$$\widehat{w}_{\text{ridge}} := (\text{Attn}_s \circ \text{Linear})(Z)u \in \mathbb{R}^d, \quad \text{and} \quad \|\widehat{w}_{\text{ridge}} - w_{\text{ridge}}\|_\infty \leq \epsilon,$$

where $w_{\text{ridge}} := \text{argmin}_{w \in \mathbb{R}^d} \frac{1}{2n}\sum_{i=1}^n (\langle w, x_i\rangle - y_i)^2 + \frac{\lambda}{2}\|w\|_2^2$ with regularization term $\lambda \geq 0$.

Proof. See Appendix D.4 for detailed proof. □

So far our constructions in Section 3 show that, given freedom to choose parameters per algorithm, attention modules emulate gradient descent, linear regression, ridge regression, and related updates in context. These results establish the expressive power of task-specific in-context emulation, akin to (Bai et al., 2023). In Section 4, we build on this foundation and prove a stronger universality: a single frozen attention module $\text{Attn}^\star$, via prompt programming, simulates all such task-specific modules.

4 Prompt-Programmable In-Context Algorithm Emulation

This section presents our main results: softmax attention is capable of (i) emulating *task-specific attention heads* in-context (Section 4.1), (ii) emulating statistical models in-context (Section 4.2), and (iii) emulating any network (with linear projections) in-context (Section 4.3). Unlike Section 3 requiring a separate task-specific module for each algorithm, here we fix one frozen module $\text{Attn}^\star$ and show that suitable prompts instruct it to emulate every algorithm in the target class. This establishes universality: one set of weights executes a library of algorithms through prompt programming.

4.1 In-Context Emulation of Attention

We first specify the input prompt with weight encoding.

Definition 4.1 (Vectorization)**.** For any matrix $X \in \mathbb{R}^{d_h \times d}$, we define $\underline{X} := \text{vec}(X) \in \mathbb{R}^{dd_h}$ such that $\underline{X}_{(i-1)d+j} = X_{i,j}$ for all $i \in [d_h]$ and $j \in [d]$.

Definition 4.2 (Input Prompt of In-Context Emulation of Attention). Let $X \in \mathbb{R}^{d\times n}$ be the input sequence, and let $W_K, W_Q, W_V \in \mathbb{R}^{d_h\times d}$ be the weight matrices of the target attention head to be emulated. Define the vectorizations

$$\underline{W}_K := \mathrm{vec}(W_K) \in \mathbb{R}^{dd_h}, \ \underline{W}_Q := \mathrm{vec}(W_Q) \in \mathbb{R}^{dd_h}, \ \underline{W}_V := \mathrm{vec}(W_V) \in \mathbb{R}^{dd_h},$$

and

$$w := [\underline{W}_K; \underline{W}_Q; \underline{W}_V] \in \mathbb{R}^{3dd_h},$$

where w is the concatenation of $\underline{W}_K, \underline{W}_Q, \underline{W}_V$. Finally, define the extended input X_p for in-context emulation of the attention head specified by W_K, W_Q, W_V as

$$X_p := \begin{bmatrix} X \\ W_{\text{in}} \\ I_n \end{bmatrix} \quad \text{with} \quad W_{\text{in}} := \begin{bmatrix} 0\cdot w & 1\cdot w & 2\cdot w & \cdots & (n-1)\cdot w \\ w & w & w & \cdots & w \end{bmatrix} \in \mathbb{R}^{6dd_h\times n}.$$

In other words, W_{in} is a $2\times n$ block matrix whose j-th column consists of $j\cdot w \in \mathbb{R}^{dd_h}$ (in the first block-row) and $w \in \mathbb{R}^{dd_h}$ (in the second block-row), for $j = 0, 1, \ldots, n-1$. Appending W_{in} as additional rows to X produces the prompt X_p that encodes the target weights.

Using this weight-encoding prompt, we now design a two-layer attention mechanism to reproduces the effect of the target attention head in-context.

Theorem 4.1 (In-Context Emulation of Attention). Let $X \in \mathbb{R}^{d\times n}$ be an input sequence, and let $W_K, W_Q, W_V \in \mathbb{R}^{d_h\times d}$ be the weight matrices of the target attention head we wish to emulate in-context. Assume $\|W_K X\|_\infty, \|W_Q X\|_\infty, \|W_V X\|_\infty \le B_{KQV}$ with $B_{KQV} > 0$. Then, for any $\epsilon > 0$, there exists a two-layer attention network — a multi-head attention layer Attn_m followed by a single-head attention layer Attn_s — such that

$$\| \underbrace{\mathrm{Attn}_s \circ \mathrm{Attn}_m(X_p)}_{\text{Emulator}} - \underbrace{W_V X \mathrm{Softmax}_\beta((W_K X)^\top W_Q X)}_{\text{Target}} \|_\infty \le \epsilon,$$

where X_p is the prompt defined in Definition 4.2.

Remark 4.1 (Permutation Equivariance). Our construction keeps the permutation equivariance of attention in its approximation. This means changing the order of columns in X results in an identical change in the order of the columns in $\mathrm{Attn}_s \circ \mathrm{Attn}_m(X_p)$.

Proof. See Appendix A.1 for the proof sketch and Appendix D.5 for a detailed proof. □

We now provide an alternative formulation of the above result. In this variant, a single-head attention layer comes first, followed by a multi-head layer with sequence-wise linear projections.

Theorem 4.2 (In-Context Emulation of Attention; Alternative Formulation). Let $X \in \mathbb{R}^{d\times n}$ be the input sequence, and let $W_K, W_Q, W_V \in \mathbb{R}^{n\times d}$ be the weight matrices of the target attention. Assume $B = \max\{\|X\|_\infty, \|W_K\|_\infty, \|W_Q\|_\infty, \|W_V\|_\infty\}$ and $\|W_K X\|_\infty, \|W_Q X\|_\infty, \|W_V X\|_\infty \le B_{KQV}$ for $B_{KQV} \ge 0$. Then, for any $\epsilon > 0$, there exists a single-head attention layer Attn_s followed by a multi-head attention layer with linear projections such that

$$\|\mathrm{Attn}_s \circ (\sum_{j=1}^{3n} \mathrm{Attn}_j \circ \mathrm{Linear}_j(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix})) - \underbrace{W_V X}_{n\times n} \mathrm{Softmax}_\beta \underbrace{((W_K X)^\top W_Q X)}_{n\times n} \|_\infty \le \epsilon.$$

Proof. See Appendix A.2 for the proof sketch and Appendix D.6 for a detailed proof. □

Theorems 4.1 and 4.2 allow us to approximate arbitrary target one-layer attention using another two-layer attention. This construction requires no feed-forward layers and no parameter updates. All approximation happens through the prompt alone (by embedding target attention weights and input X into the prompt).

Discussion: Target Attention Approximation for Algorithm Emulation. Theorems 4.1 and 4.2 present a general algorithm emulation result: a fixed-weight two-layer softmax attention module emulates *all* algorithms implementable by softmax attention via only prompting. For example, if we choose the input sequence $X \in \mathbb{R}^{d\times n}$ in Theorem 4.1 and Theorem 4.2 to be the $\mathsf{Linear}(Z) \in \mathbb{R}^{2(2d+n+2)\times n(P+1)}$ in Theorem 3.1, then we are able to approximate all one-layer attentions implementing target algorithms of the $f(w^\top x - y)x$ class: Corollaries 3.1.1 to 3.1.4. Thus, we achieve in-context emulation of the entire class of algorithms expressible as $f(w^\top x - y)x$.

To the best of our knowledge, this provides the first constructive toy model of fixed-weight transformer exhibiting *general-purpose ability* (i.e., *one* fixed-weight model for *many* tasks). Moreover, the construction is explicit, interpretable, and softmax-native. A few remarks are in order.

Remark 4.2 (Differences between Theorems 4.1 and 4.2)**.** Theorems 4.1 and 4.2 both establish that a fixed multi-head attention network can approximate any given attention head in-context. We present two versions of the construction using different formulations and analytical techniques. In particular, Theorem 4.1 encodes the target algorithm into the token representations (keeping the sequence length fixed), whereas Theorem 4.2 achieves a similar effect by encoding the weights as additional tokens in the input sequence (keeping each token's dimension fixed).

Remark 4.3. Our constructions may contain non-standard choices, including encoding information along the embedding dimension and using $3n$ parallel attention heads. We emphasize that the methods apply to approximate a more realistic attention with far fewer hidden dimensions and number of heads in practice. Section 5 provides further details.

Remark 4.4 (Comparison with Prior Work)**.** We remark that our results differ from prior work in three key aspects. First, we study the practical softmax attention rather than linear or ReLU attention (Bai et al., 2023; Von Oswald et al., 2023; Vladymyrov et al., 2024). Second, our results in Section 4 go beyond task-specific ICL and establish that fixed-weight Transformers are prompt-programmable (Bai et al., 2023; Wu et al., 2025; Li et al., 2025). Third, our results are constructive, providing concrete emulation examples in contrast to prior prompting expressivity (Wang et al., 2023; Furuya et al., 2024) or Turing-completeness results (Pérez et al., 2021; Giannou et al., 2023; Qiu et al., 2024). The closest works to ours are (Giannou et al., 2023; Bai et al., 2023). Bai et al. (2023) show that Transformers can execute several standard algorithms in-context, but each algorithm uses its own tailored attention layer. Our results show that a single fixed attention module can emulate a broad set of algorithms through prompt changes. Giannou et al. (2023) study a fixed looped Transformers that implement arbitrary programs. However, their results are "there exists"-type: their universal Transformer is a conceptual Turing machine, not a fully specified numerical model. While our results are also extensible to looped setting (i.e., Corollary 3.1.2), our focus is different: we use an attention-only, FFN-free model and analyze its algorithmic universality *constructively*. This highlights the power of softmax attention mechanism and offers a clean testbed for developing scientific theories (interpretable, controllable and predictable like physics) of GPT-style foundation models.

Extension: Modern Hopfield Networks. We extend our results to in-context optimization ability of dense associative memory models (Ramsauer et al., 2021) in Appendix E.

4.2 In-Context Emulation of Statistical Methods

Theorem 4.2 shows that a frozen attention module approximates a target attention head by embedding the head's weights into its input prompt. We now leverage this idea to emulate a broader class of algorithms. In essence, we replace the embedded target attention weights with the parameters of an arbitrary statistical method that we aim to emulate. By the same principle, the fixed attention module then mimics the behavior of diverse statistical models within the in-context learning framework.

Corollary 4.2.1 (In-Context Emulation of Statistical Methods)**.** Let $\mathcal{A}$ be the set of all algorithms implementable by a single-layer attention network in-context. For any finite collection of algorithms $\{a_1, a_2, \ldots, a_k\} := \mathcal{A}_0 \subseteq \mathcal{A}$, there exists a two-layer attention network (a single-head layer Attn_s composed with a multi-head layer Attn_m) such that for each $a \in \mathcal{A}_0$ in the collection

$$\|\sum_{j=1}^{3n} \mathrm{Attn}_s \circ \mathrm{Attn}_j \circ \mathrm{Linear}_j \left(\begin{bmatrix} X \\ W^a \end{bmatrix} \right) - a(X)\|_\infty \leq \epsilon,$$

where W^a is the W defined as Definition 4.2 using W_K^a, W_Q^a, W_V^a.

Proof. See Appendix D.7 for detailed proof. □

We show that a fixed attention module emulates an arbitrary finite library of in-context algorithms by varying its prompt. This result highlights the flexibility of softmax attention: unlike prior work that requires re-training or fine-tuning of the model, here we provably achieve task-specific behavior by modifying the input prompt. In effect, a pretrained Transformer internalizes a small set of fundamental procedures and later deploys them, via prompting, across a wide range of data distributions. Since the number of distinct algorithms is far smaller than the number of possible datasets, a model that learns a handful of algorithms can leverage them to handle many different scenarios.

4.3 Attention Makes Every (Linear) Network In-Context

We now extend the above ideas to show that softmax attention emulates *any* network (comprised of linear transformations) in-context. Consider any layer of a neural network that applies a trainable linear map $x \to \Theta x$ with weight matrix Θ. Our results imply that if Θ is provided as part of the input sequence, a fixed attention module is capable of approximating this transformation to arbitrary precision. Hence linear layers in standard architectures are replaceable with attention whose effective weights are encoded in the prompt rather than learned. This substitution turns the network into an in-context learner in place of, or alongside, conventional training.

Remark 4.5 (In-Context Emulation of Linear Layers)**.** For example, suppose a model contains a linear layer $f(x) = \Theta x$ with weight matrix Θ. By including Θ (appropriately encoded) in the input as in our constructions above, a single softmax attention layer emulates $f(x)$ in-context to arbitrary precision. In other words, any trainable linear mapping in the original network is replicable with a prompt-programmable attention layer whose parameters are set by the input sequence. This enables the overall network to adjust that layer's behavior on-the-fly via prompts, rather than having to learn Θ through pre-training.

5 Numerical Studies

This section provides numerical results to back up our theory. We validate two building blocks on synthetic data: (i) approximation of continuous functions (Section 5.1); and (ii) approximation of attention heads (Section 5.2). These studies quantify approximation error and its relation to model size and the number of heads.

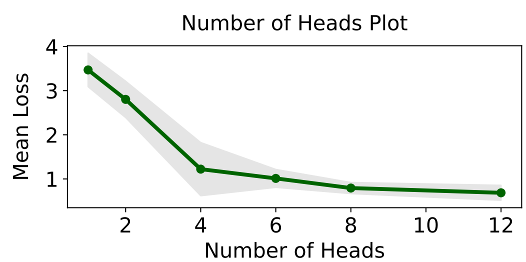

Figure 2: **Sensitivity of Attention Emulation to the Number of Heads.** We report loss (MSE) as the mean and one standard deviation (shaded region) over 10 random seed runs. We use synthetic data of 50000 data points with sequence length being 20 and input dimension being 24. We set batch size to be 32 and hidden dimension to be 48. Each multi-head model and the single-head softmax attention layer is trained for 50 epochs. The optimizer used is Adam with learning rate 0.001. We visualize the performance (MSE $\pm$ Std) for $1, 2, 4, 6, 8, 12$ heads.

5.1 Proof-of-Concept Experiment on Theorem 3.1

Objective: Verifying Attention Approximates $f(w^\top x - y)x$. We investigate accuracy of softmax attention approximating $f(w^\top x - y)x$ by training a single-head softmax attention with linear connection.

Data Generation. We randomly generate $X \in \mathbb{R}^{n \times d}$ drawn from a normal distribution, $X \sim 10 \cdot N(0,1) - 5$. We also generate weight matrix $W \in \mathbb{R}^{n \times d}$ and $y \in \mathbb{R}^n$, both randomly drawn from a standard normal distribution, $N(0,1)$. Here, n represents the sequence length and d represents input dimension. The true label is $f(w^\top x - y)x$, where we choose $f(\cdot) = \tanh(\cdot)$.

Model Architecture. We train a single-head attention network with linear transformation to approximate $\tanh\big(w^\top x - y\big)x$. We first apply linear transformation to both $[X; y]$ and W. We then train the single-head attention model with the linear transformations to approximate our target function as shown in the proof of Theorem 3.1.

Results. As shown in Figure 2, evaluated on Mean Square Error loss, the model approximates the target $\tanh\big(w^\top x - y\big)x$ with minimal error. This experiment proves our theory.

5.2 Proof-of-Concept Experiment on Emulating Attention Heads

Objective: Verifying Approximation Rates. We investigate the affect of the number of attention heads H on the accuracy of softmax attention approximating softmax attention head.

Data Generation. We randomly generate a sequence of tokens $X = [x_1, x_2, \cdots, x_n] \in \mathbb{R}^{d \times n}$, where each entry x_i is drawn independently from a normal distribution,

$$X \sim 2 \cdot N(0, 1) - 1.$$

We also generate weight matrices $K = W_K X^\top \in \mathbb{R}^{h \times n}$, $Q = W_Q X^\top \in \mathbb{R}^{h \times n}$, and $V = W_V X^\top \in \mathbb{R}^{d \times n}$. Each parameter matrix is randomly drawn from a standard normal distribution, $N(0, 1)$. Here, n represents the sequence length, d represents token dimension, and h represents hidden dimension. The true label $Y \in \mathbb{R}^{d \times n}$ results from applying a single-layer softmax attention mechanism on inputs X, K, Q, and V.

Model Architecture. We train a multi-layer attention network to approximate softmax attention function. We first train separate multi-head models with linear transformation to approximate K, Q, and V. Then, we use a single-head softmax attention layer to approximate softmax attention function as in the proof.

Results. As shown in Figure 2 and Table 1, the result validates our claim that a multi-head softmax attention mimics a target softmax attention head to arbitrary precision. Moreover, it demonstrates the convergence of multi-head softmax attention emulating softmax-based attention mapping as the number of heads increases. The approximation rate is in the trend of $O(1/H)$ where H is the number of heads. The small and decreasing MSE error indicates that the simple softmax attention model approximates softmax attention head with stability.

Table 1: **Sensitivity to the Number of Heads.** Emulation MSE (mean $\pm$ std) for multi-head softmax attention with $1, 2, 4, 6, 8$, and 12 heads.

Heads	**1**	**2**	**4**	**6**	**8**	**12**
MSE	3.469	2.802	1.222	1.012	0.793	0.686
Std	0.381	0.413	0.603	0.204	0.127	0.171

Additional Experiments. Due to page limits, we defer several experimental results to Appendix C. These include simulations of statistical algorithms (Appendix C.1) and approximations of statistical models on real-world datasets where the model does not have access to the true algorithm weights (Appendix C.2). They further illustrate the approximation capabilities of Transformer in practice.

6 Discussion and Conclusion

We study *in-context algorithm emulation* in fixed-weight Transformers and formalize two modes: task-specific (Section 3) and prompt-programmable algorithm emulation (Section 4). For the former, we show that even a single-layer, single-head module suffices for emulating core families (of the form $f(w^\top x - y)x$) such as one-step gradient descent and linear/ridge regression, achieving architectural minimality (Theorem 3.1). For the latter, we show that a two-layer multi-head softmax attention module emulates a broad class of algorithms by embedding the algorithm's weights into the input prompt (Theorem 4.1). Altogether, a fixed softmax attention module becomes a prompt-programmable *library of algorithms*: weights remain frozen, and the prompt selects the routine.

Mechanism. The mechanism is constructive. By encoding target weights in the input and creating large dot-product margins, softmax attention routes along the intended computation without weight updates. Numerical studies support the theory: on synthetic data the model accurately approximates continuous maps of the form $f(\langle w, x\rangle - y)\, x$ and emulates attention heads. Approximation error decreases as the number of heads grows. On a real dataset (Ames Housing), the frozen module-driven by prompts rather than true algorithm weights-achieves low error against standard statistical models.

Implications. Our results tighten the link between in-context learning and algorithmic emulation. Viewing prompts as callable subroutines that select and configure algorithms within a frozen model, we draw three takeaways: (i) prompt engineering becomes interface design for algorithm selection, (ii) pretraining objectives *could*, in future work, be designed to encourage learning compact libraries of reusable procedures, and (iii) analyses of internal routing help clarify how foundation models select among algorithms. This lens explains the breadth of in-context generalization, guides prompt design, and motivates new pretraining objectives for more effective algorithm installation and utilization.

Ethic Statement

This paper does not involve human subjects, personally identifiable data, or sensitive applications. We do not foresee direct ethical risks. We follow the ICLR Code of Ethics and affirm that all aspects of this research comply with the principles of fairness, transparency, and integrity.

Reproducibility Statement

We ensure reproducibility on both theoretical and empirical fronts. For theory, we include all formal assumptions, definitions, and complete proofs in the appendix. For experiments, we describe model architectures, datasets, preprocessing steps, hyperparameters, and training details in the main text and appendix. Code and scripts are provided in the supplementary materials to replicate the empirical results.

References

Jacob Abernethy, Alekh Agarwal, Teodor Vanislavov Marinov, and Manfred K Warmuth. A mechanism for sample-efficient in-context learning for sparse retrieval tasks. In *International Conference on Algorithmic Learning Theory*, pages 3–46. PMLR, 2024.

Kwangjun Ahn, Xiang Cheng, Hadi Daneshmand, and Suvrit Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *Advances in Neural Information Processing Systems*, 36:45614–45650, 2023.

Ekin Akyürek, Dale Schuurmans, Jacob Andreas, Tengyu Ma, and Denny Zhou. What learning algorithm is in-context learning? investigations with linear models. In *The Eleventh International Conference on Learning Representations*, 2023.

Yu Bai, Fan Chen, Huan Wang, Caiming Xiong, and Song Mei. Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Advances in neural information processing systems*, 36:57125–57211, 2023.

Rishi Bommasani. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

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

Dean De Cock. Ames, iowa: Alternative to the boston housing data as an end of semester regression project. *Journal of Statistics Education*, 19(3):1–13, 2011.

Takashi Furuya, Maarten V de Hoop, and Gabriel Peyré. Transformers are universal in-context learners. *arXiv preprint arXiv:2408.01367*, 2024.

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

Angeliki Giannou, Shashank Rajput, Jy-yong Sohn, Kangwook Lee, Jason D Lee, and Dimitris Papailiopoulos. Looped transformers as programmable computers. In *International Conference on Machine Learning*, pages 11398–11442. PMLR, 2023.

Jerry Yao-Chieh Hu, Hude Liu, Hong-Yu Chen, Weimin Wu, and Han Liu. Universal approximation with softmax attention. *arXiv preprint arXiv:2504.15956*, 2025a.

Jerry Yao-Chieh Hu, Wei-Po Wang, Ammar Gilani, Chenyang Li, Zhao Song, and Han Liu. Fundamental limits of prompt tuning transformers: Universality, capacity and efficiency. In *The Thirteenth International Conference on Learning Representations*, 2025b.

Tokio Kajitsuka and Issei Sato. Are transformers with one layer self-attention using low-rank weight matrices universal approximators? In *The Twelfth International Conference on Learning Representations*, 2024.

Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

Gen Li, Yuchen Jiao, Yu Huang, Yuting Wei, and Yuxin Chen. Transformers meet in-context learning: A universal approximation theory. *arXiv preprint arXiv:2506.05200*, 2025.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2021.

Yingcong Li, Muhammed Emrullah Ildiz, Dimitris Papailiopoulos, and Samet Oymak. Transformers as algorithms: Generalization and stability in in-context learning. In *International conference on machine learning*, pages 19565–19594. PMLR, 2023.

Hude Liu, Jerry Yao-Chieh Hu, Zhao Song, and Han Liu. Attention mechanism, max-affine partition, and universal approximation. *arXiv preprint arXiv:2504.19901*, 2025.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9):1–35, 2023.

Sheng Liu, Haotian Ye, Lei Xing, and James Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *Forty-first International Conference on Machine Learning*, 2024.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, 2022.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064, 2022.

Jorge Pérez, Pablo Barceló, and Javier Marinkovic. Attention is turing-complete. *Journal of Machine Learning Research*, 22(75):1–35, 2021.

Ruizhong Qiu, Zhe Xu, Wenxuan Bao, and Hanghang Tong. Ask, and it shall be given: On the turing completeness of prompting. *arXiv preprint arXiv:2411.01992*, 2024.

Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Thomas Adler, Lukas Gruber, Markus Holzleitner, Milena Pavlović, Geir Kjetil Sandve, et al. Hopfield networks is all you need. 2021.

Max Vladymyrov, Johannes Von Oswald, Mark Sandler, and Rong Ge. Linear transformers are versatile in-context learners. *Advances in Neural Information Processing Systems*, 37:48784–48809, 2024.

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

Yihan Wang, Jatin Chauhan, Wei Wang, and Cho-Jui Hsieh. Universality and limitations of prompt tuning. *Advances in Neural Information Processing Systems*, 36:75623–75643, 2023.

Weimin Wu, Maojiang Su, Jerry Yao-Chieh Hu, Zhao Song, and Han Liu. In-context deep learning via transformer models. In *International Conference on Machine Learning*. PMLR, 2025.

Chulhee Yun, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank J Reddi, and Sanjiv Kumar. Are transformers universal approximators of sequence-to-sequence functions? In *International Conference on Learning Representations*, 2020.

Ruiqi Zhang, Spencer Frei, and Peter L Bartlett. Trained transformers learn linear models in-context. *Journal of Machine Learning Research*, 25(49):1–55, 2024.

Yufan Zhuang, Chandan Singh, Liyuan Liu, Jingbo Shang, and Jianfeng Gao. Vector-icl: In-context learning with continuous vector representations. In *The Thirteenth International Conference on Learning Representations*, 2025.

Appendix

Impact Statement

We prove that a single frozen softmax attention head emulates a broad library of attention-implementable algorithms via prompt design, establishing pretrained Transformers as universal algorithm stores and reducing the need for task-specific fine-tuning. This sharpens the theoretical basis of in-context learning, offers a principled recipe for prompt engineering, and equips auditors with a clear test for hidden prompt-encoded behaviors, all without releasing new models or data. Therefore, the work advances foundational understanding, lowers compute and energy demands, and introduces minimal societal risk.

Limitations and Future Direction

Prompt length grows linearly with the weight dimension, which limits practicality. The proofs assume exact real-valued softmax and ignore token discretizations or numerical noise. Prompts are hand-crafted. Learning them automatically is open. Language and vision inputs are untested. Weight encoding happens along embedding dimension. The construction is not permutation invariant, but permutation equivariant of the input data. Lastly, we leave tighter constants, shorter prompts, extensions to deeper models and connection with model pretraining to future work.

LLM Usage Disclosure

We used large language models (LLMs) to aid and polish writing, such as improving clarity, grammar, and conciseness. We also used LLMs for retrieval and discovery, for example exhausting literature to identify potential missing related work. All technical content, proofs, experiments, and results are original contributions by the authors.

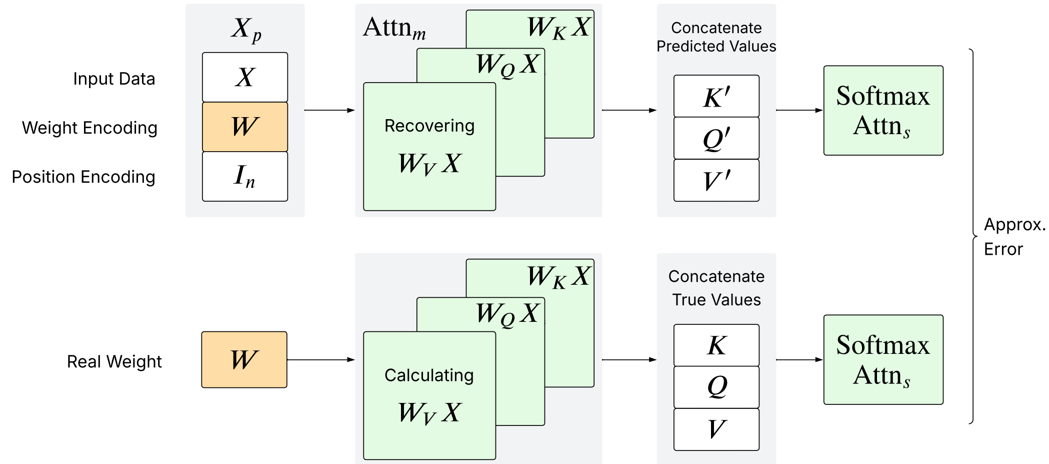

Figure 3: **Visualization of Proof Sketch for Theorem 4.1**. We visualize our proof technique. We combine input data, weight encoding, and position encoding into X_p as input to the multi-head attention Attn_m to recover approximate key, query, and value representations. We then compare the single-head attention Attn_s outputs from approximate values with ground truth values to obtain approximation error.

A PROOF SKETCHES

We present our proof strategies here.

A.1 PROOF SKETCH FOR THEOREM 4.1

We construct a two-layer Transformer (single-head layer Attn_s followed by multi-head layer Attn_m) that replicates the target attention to within any error $\epsilon > 0$. Recall from Theorem 4.1:

$$\| \underbrace{\overbrace{\text{Attn}_s}^{\textbf{Step 3}} \circ \underbrace{\text{Attn}_m(\overbrace{X_p}^{\textbf{Step 1}})}_{\textbf{Step 2}}}_{\text{Emulator}} - \underbrace{W_V X \text{Softmax}((W_K X)^\top W_Q X)}_{\text{Target}} \|_\infty \leq \epsilon.$$

The high-level idea is: (**Step 1**) augment the input with a prompt encoding of the target weights W_K, W_Q, W_V, (**Step 2**) use groups of heads in Attn_m to approximate the matrices $K := W_K X$, $Q := W_Q X$, $V := W_V X$ in-context (up to small error), and (**Step 3**) apply Attn_s with fixed weights to assemble the Attn_m output: the approximators K', Q', V'. We then argue in (**Step 4**) the approximation error can be made $< \epsilon$ via a stability bound on softmax attention.

Step 1: In-Context Weight Encoding. We augment the input $X \in \mathbb{R}^{d \times n}$ by appending special tokens encoding the matrices W_K, W_Q, W_V. We denote the augmented input as X_p. This allows the transformer $\text{Attn}_s \circ \text{Attn}_m$ to "read" the relevant weight parameters in its attention heads.

Explicitly, we embed both the data sequence and the target head into the input (Definition 4.2):

$$X_p = \begin{bmatrix} X \\ W_{\text{in}} \\ I_n \end{bmatrix} \quad \text{with} \quad W_{\text{in}} := \begin{bmatrix} 0 \cdot w & 1 \cdot w & 2 \cdot w & \cdots & (n-1) \cdot w \\ w & w & w & \cdots & w \end{bmatrix},$$

where $w = [\underline{W}_K^\top, \underline{W}_Q^\top, \underline{W}_V^\top]^\top$ concatenates every column of $W_K^\top, W_Q^\top, W_V^\top$ following Definition 4.2. The block I_n provides token-position codes used in our construction.

The idea of this augmentation is to emulate the target computation $W_V X \text{Softmax}((W_K X)^\top W_Q X)$ with the emulator: a two-layer transformer $\text{Attn}_s \circ \text{Attn}_m$. To achieve this, $\text{Attn}_s \circ \text{Attn}_m$ must access the information of X, W_K, W_Q and W_V in-context. The augmentation above encodes these parameters (target algorithm's specifications) in the prompt.

Step 2: Multi-Head Decomposition for In-Context Recovery of K, Q, V. We devote the first attention layer (Attn_m) to recovering the key, query, and value matrices that the target attention would compute. By definition, each row in K, Q, and V takes the form: $k_i^\top X, q_i^\top X$ and $v_i^\top X$. Here $k_i^\top, q_i^\top$ and $v_i^\top$ are rows in W_K, W_Q and W_V. Our goal is, for each data token x_i (the i-th column of X), to approximate $k_j^\top x_i$, $q_j^\top x_i$, and $v_j^\top x_i$. To do this, we design Attn_m to have a fixed number of heads partitioned into *three groups*, corresponding to K, Q, and V respectively. Combining heads' outputs within each group yields approximations of K, Q, and V. Explicitly, in the first multi-head layer Attn_m, we split the heads so that:

- A group of heads jointly approximates $W_K X$. By (Hu et al., 2025a, Theorem 3.2), the heads in this group admit further subdivision into sub-groups. Each sub-group outputs a linear transformations of X, namely $k_i^\top X$ for rows $k_i^\top X$ of K.
- Another group of heads approximates $W_Q X$ in a similar manner.
- A final group approximates $W_V X$.

Concatenate or combine these head outputs so that the final embedding from $\mathrm{Attn}_m(X_p)$ contains (up to small error) the blocks $[K; Q; V]$ for all positions in X.

Explicitly, for each row $k_j^\top$ of W_K (and similarly for $q_j^\top$ and $v_j^\top$), we prepend the corresponding heads with a token-wise linear map $A(\cdot)$. $A(X_p)$ pulls out the target row (i.e., k_j) from w and repeats it n times. The resulting sub-prompt $A(X_p)$ has the form

$$\begin{bmatrix} & & X & \\ 0 \cdot k_j & 1 \cdot k_j & \cdots & (n-1) \cdot k_j \\ k_j & k_j & \cdots & k_j \\ & & I_n & \end{bmatrix},$$

so the corresponding softmax heads return $k_j^\top X$ up to any error ϵ_0 by the truncated-linear interpolation theorem (Theorem D.1). With $H = \lceil 2(b-a)/((n-2)\epsilon_0) \rceil$ heads per sub-group, we cover all d_h rows in K (and similarly for Q and V). Altogether, the $3N = 3d_h H$ heads satisfy

$$\| \underbrace{\sum_{j=1}^{d_h} \mathrm{Attn}_j^K(X_p)}_{:=K'} - K \|_\infty \leq \epsilon_0, \ \| \underbrace{\sum_{j=d_h+1}^{2d_h} \mathrm{Attn}_j^Q(X_p)}_{:=Q'} - Q \|_\infty \leq \epsilon_0, \ \| \underbrace{\sum_{j=2d_h+1}^{3d_h} \mathrm{Attn}_j^V(X_p)}_{:V'} - V \|_\infty \leq \epsilon_0.$$

We collect these outputs column-wise into

$$\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}, \quad \text{and} \quad \| \begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix} - \begin{bmatrix} K \\ Q \\ V \end{bmatrix} \|_\infty \leq \epsilon_0.$$

Step 3: Single-Head Assembly for Emulated Map. We consider the second layer Attn_s as a single-head attention with fixed weights chosen to "read" the K', Q', V' triples from $Z := \mathrm{Attn}_m(X_p)$ and perform the "emulated" attention mechanism. Explicitly, apply a single-head attention layer Attn_s whose parameters are set to read off the K, Q, and V sub-blocks in each token embedding:

$$\mathrm{Attn}_s(Z) := W_V^{(s)} Z \mathrm{Softmax}((W_K^{(s)} Z)^\top (W_Q^{(s)} Z)).$$

For $Z := \mathrm{Attn}_m(X_p)$, we choose fixed weights

$$W_K^{(s)} = [0_{d_h \times 2d_h} \quad I_{d_h}], \quad W_Q^{(s)} = [I_{d_h} \quad 0_{d_h \times 2d_h}], \quad W_V^{(s)} = [0_{d_h \times d_h} \quad I_{d_h} \quad 0_{d_h \times d_h}],$$

so that

$$W_K^{(s)} Z \approx W_K X, \quad W_Q^{(s)} Z \approx W_Q X, \quad W_V^{(s)} Z \approx W_V X.$$

Hence,

$$\begin{aligned} \mathrm{Attn}_s(Z) = \mathrm{Attn}_s \circ \mathrm{Attn}_m(X_p) = \mathrm{Attn}_s(\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}) &= V' \mathrm{Softmax}((K')^\top Q') \\ &\approx W_V X \mathrm{Softmax}((W_K X)^\top W_Q X). \end{aligned}$$

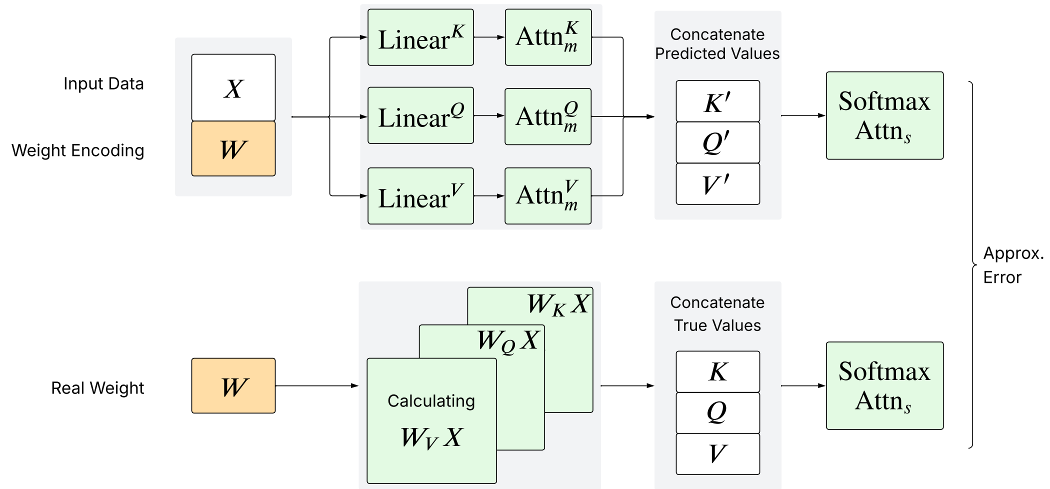

Figure 4: **Visualization of Proof Sketch for Theorem 4.2**. We visualize our proof technique. We combine input data and weight encoding as input. Each key, query, and value has a unique set of linear transformation of input (Linear) and multi-head attention (Attn_m). We feed the input into each set to attain the approximate key, query, and value representations, respectively. We then compare the single-head attention Attn_s outputs from approximate values with ground truth values to obtain approximation error.

To be precise, because K', Q', V' differ from K, Q, V by at most ϵ_0 (in $\|\cdot\|_\infty$), a first-order perturbation argument for softmax (uniform Lipschitz in sup-norm) shows

$$\|\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\| \le \epsilon_0 + nB_{KQV}\epsilon_1,$$

where B_{KQV} bounds X, W_K, W_Q, W_V and $\epsilon_1 = O(\epsilon_0)$.

Step 4: Error Bound. Finally, we make the approximation arbitrarily precise. Because we are capable of making each head's linear approximation arbitrarily close, we ensure

$$\|\mathrm{Attn}_s \circ \mathrm{Attn}_m([X; W]) - W_V X \mathrm{Softmax}((W_K X)^\top W_Q X)\|_\infty \le \epsilon,$$

for any $\epsilon > 0$. This completes the construction, proving in-context emulation of the target attention. Please see Appendix D.5 for a detailed proof and Figure 3 for proof visualization.

A.2 Proof Sketch for Theorem 4.2

We outline how to emulate the desired attention step-by-step with a fixed two-layer transformer. Similar to Theorem 4.1 (and Theorem D.1), our construction ensures each token's representation in the intermediate layer carries an approximate copy of its key, query, and value vectors, which the final layer uses to perform the softmax attention. All necessary components (including the weight matrices W_K, W_Q, W_V themselves) are encoded into the input, so the network's weights remain untrained and generic.

Step 1: Encoding Weights into the Input. Let $X \in \mathbb{R}^{d\times n}$ be the input tokens. Append a "weight encoding" matrix W that contains the rows of W_K, W_Q, W_V (the weight matrices of the target attention head). This forms an extended input $[X; W]$. The entries of X and W remain within a bounded range $[-B, B]$. This bound ensures that all inner products remain finite.

Step 2: Multi-Head Approximation of K, Q, V. The first layer has many heads. Partition them into three groups. One group approximates $K := W_K X$; one approximates $Q := W_Q X$; and one approximates $V := W_V X$. Then

- **Simulating Dot Products on a 1D Grid.** Consider a single entry $k_j^\top x_c$. All entries in K, Q, V lie between $[-dB^2, dB^2]$, since the entries of X and W remain within a bounded range $[-B, B]$. We create grid points $L_0 < \cdots < L_P$ covering $[-dB^2, dB^2]$. We design the head's key and query so that the softmax assigns each grid point L_i a weight based on its distance to $k_j^\top x_c$. We set the value vector to encode L_i. Thus, the head output for token x_c approximates $k_j^\top x_c$. Fine grids reduce the error.

- **Reconstructing Full K, Q, V.** Repeat this idea for every entry in K, Q, V. Each row uses one head to approximate $k_j^\top X$, $q_j^\top X$, or $v_j^\top X$. Combine these approximations to obtain the matrices K', Q', V'. The sup norm

$$\left\| \begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix} - \begin{bmatrix} K \\ Q \\ V \end{bmatrix} \right\|_\infty, \quad \text{can be made arbitrarily small.}$$

Step 3: Single-Head Assembly of the Attention Output. The second layer, Attn_s, has one head. We set its weight matrices $W_K^{(s)}, W_Q^{(s)}, W_V^{(s)}$ to pick out K', Q', V' from each token's embedding. Then, Attn_s computes

$$V'\mathrm{Softmax}((K')^\top Q') \approx W_V X \mathrm{Softmax}((W_K X)^\top W_Q X),$$

since $K' \approx K$, $Q' \approx Q$ and $V' \approx V$.

Step 4: Error Bound. Softmax and matrix multiplication are continuous. Small errors in K', Q', V' cause a small error in the final output. By refining the grid (and using enough heads), we make the sup norm error below any $\epsilon > 0$. Please see Appendix D.6 for a detailed proof and Figure 4 for proof visualization.

B Related Work

Our results diverge from prior findings on Transformer universality and in-context learning.

B.1 Core Related Work

Universal Approximation. Prior studies establish that Transformers approximate arbitrary sequence-to-sequence functions, but they do not address in-context learning and often assume complex architectures. For example, Yun et al. (2020) prove that deep multi-head Transformers with feed-forward layers are universal approximators of continuous sequence-to-sequence functions. Subsequent advances tighten this finding: Kajitsuka and Sato (2024); Hu et al. (2025b) show that even a single-layer Transformer realizes any continuous sequence function. However, these results treat Transformers as parametric function approximators. The model requires re-training and re-prompting to adapt to a new target function instead of handling multiple tasks through context. In contrast, we prove that a minimal Transformer architecture, even a single-layer, single-head attention module with no feed-forward network, emulates a broad class of algorithms without weight updates by varying its prompt. This result achieves a new level of generality through context alone (i.e. prompt-based conditioning) despite a fixed minimalist model.

In-Context Learning and Algorithm Emulation. Another line of recent theory bridges Transformers with in-context learning by designing model components to carry out specific algorithms. For example, Bai et al. (2023) show that Transformers execute a broad range of standard algorithms in-context, but each algorithm requires a distinct, tailored attention head. In comparison, we extend this approach by showing that one fixed attention mechanism emulates any specialized attention head via prompt encoding. Rather than crafting a different attention module for each algorithm, a single frozen softmax-based attention layer takes its instructions from the prompt to perform all tasks in context. This minimal model thus becomes a unified and compact in-context algorithm emulator. It switches behaviors by changing only its input prompt, setting it apart from earlier approaches that required per-task reparameterization.

B.2 Broader Discussion

Universal Approximation and Expressivity of Transformers. Transformers exhibit strong expressive power as sequence models. Recent theory shows even minimal Transformer architectures approximate broad classes of functions. Kajitsuka and Sato (2024); Hu et al. (2025b) prove a single-layer, single-head Transformer can memorize any finite dataset perfectly. Kajitsuka and

Sato (2024) achieve this with low-rank attention matrices, while Hu et al. (2025b) use attention matrices of any rank. Adding two small feed-forward layers makes it a universal approximator for continuous sequence functions under permutation-equivariance. More recently, Hu et al. (2025a) show self-attention layers alone are universal approximators. Specifically, two attention-only layers approximate continuous sequence-to-sequence mappings, and even a single softmax-attention layer suffices for universal approximation. Similarly, Liu et al. (2025) also demonstrate that one single-head attention connected with linear transformations is sufficient to approximate any continuous function in L_∞ norm. These results eliminate the need for feed-forward networks, improving on earlier constructions. Overall, these findings highlight the inherent expressiveness of minimal attention mechanisms.

Transformers as In-Context Learners and Algorithm Emulators. Large Transformers also learn in-context by conditioning on examples in their prompts, without updating weights (Brown et al., 2020). Recent work formally explains this by showing attention-based models implement standard learning algorithms internally. Bai et al. (2023) construct Transformer heads executing algorithms such as linear regression, ridge regression, Lasso, and gradient descent steps, achieving near-optimal predictions. Wu et al. (2025) further build Transformers explicitly simulating multiple gradient descent iterations for training deep neural networks, with provable convergence guarantees. Empirical and theoretical studies confirm Transformers internalize learning algorithms when meta-trained on task families. Garg et al. (2022) show meta-trained Transformers mimic classical algorithms, such as ordinary least squares regression, in-context. Similarly, Akyürek et al. (2023); Von Oswald et al. (2023); Zhang et al. (2024) analyze Transformers trained on linear regression tasks and demonstrate their outputs mimic gradient descent steps precisely. Overall, existing literature shows that sufficiently trained or carefully designed Transformers emulate step-by-step computations of standard algorithms through prompt conditioning.

Prompt Tuning. Prompt-tuning adapts frozen models by learning a short continuous prefix (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2022). It keeps backbone weights fixed and updates only prompt embeddings. Our setting is stricter: prompts are hand-designed, not learned, and we give exact approximation bounds. Thus we expose the theoretical limit of prompt control: a single frozen softmax head can mimic any task-specific head.

Encoding Context Along Embedding Dimension. Recent work in in-context learning explores encoding and manipulating context in the embedding space rather than sequence dimension. For example, Liu et al. (2024) propose In-Context Vectors for steering the model's behavior by adding task-specific vectors along the embedding space. Zhuang et al. (2025) extend this idea by showing that manipulating embedding vectors such as interpolation makes in-context learning more controllable. Abernethy et al. (2024) showcase that appending additional information along the embedding dimension allows the model to perform sample-efficient in-context learning.

Comparison to Our Work. The above results demonstrate the versatility of Transformer networks, but they require task-specific weights, training, or learned prompts. For instance, Bai et al. (2023) design a different task-specific head for each algorithm of interest, raising the question of whether a single fixed attention mechanism could instead serve as a universal emulator for any algorithm given the right prompt. Our work directly addresses this question. In contrast, we prove one fixed softmax head emulates any specialized head through prompt encoding alone. No additional weights or training are required. Even the simplest attention (one layer, one head) acts as a universal algorithm emulator when given the right prompt, shifting focus from architecture to prompt design.

C Additional Numerical Studies

We extend the synthetic validation to statistical algorithms (Appendix C.1) and include a real-world study (Appendix C.2). The frozen attention module emulates linear, ridge, and lasso on synthetic data. On the Ames Housing dataset, the model operates without access to true algorithm weights and achieves low approximation error. In addition, we validate Theorem 4.2 through handcrafted frozen attention weights and parameters as constructed in the proof (Appendix C.3).

C.1 Proof-of-Concept Experiment on Emulating Statistical Models

Objective: Emulation of Statistical Models. We investigate the accuracy of a frozen softmax attention approximating statistical models including Lasso, Ridge and linear regression by only varying the input prompts.

Data Generation. We simulate an in-context dataset by randomly generating a sequence of input tokens $X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{n\times d}$, where each x_i is independently drawn from a scaled standard normal distribution,

$$x_i \sim 2 \cdot N(0, 1) - 1.$$

A task-specific prompt vector $w \in \mathbb{R}^{p\times 1}$ is sampled from $N(0, 1)$. In the case of Lasso, we randomly zero out entries in w with probability 0.5 to induce sparsity. We generate the output sequence $Y \in \mathbb{R}^{n\times 1}$ via a noisy linear projection: $Y = Xw + \epsilon$, where $\epsilon \sim N(0, \sigma^2)$ is Gaussian noise. For Ridge, we calculate weights using $(X^\top X + \lambda I_d)^{-1}X^\top Y$ with $\lambda = 5$.

Model Architecture and Training. We use a mixture of statistical data to train a single-layer attention network with linear transformation. Each input sample consists of $X \in \mathbb{R}^{n\times d}$ and algorithm-specific prompt $w \in \mathbb{R}^p$. We replicate w across the sequence length and concatenate it with X along the feature dimension to obtain an augmented input $[X; w] \in \mathbb{R}^{n\times(d+p)}$. We pass it through a multi-head attention layer. We train the model for 300 epochs using the Adam optimizer with a learning rate of 0.001. We use 6 attention heads, a hidden dimension of 48, an input dimension of 24, a batch size of 32, and 50000 synthetic samples. After training, we freeze the attention weights, resulting in a fixed softmax attention layer. We evaluate the frozen model on its ability to emulate various statistical algorithms using test data.

Baseline Architecture. We train three separate attention models for Lasso, Ridge, and linear regression. That is, each attention model weights are adaptive to its corresponding algorithm. We use these models as baselines for comparison with the frozen attention model we propose. All baseline models use the same hyper-parameters as the frozen model.

Results. As shown in Table 2, we compare mean MSE and standard deviation over 5 random seed runs for the frozen attention model against baseline for Lasso, Ridge, and linear regression on the synthetic data. The frozen attention model performs as well as the baseline models trained individually on each algorithm. It achieves lower MSE on Lasso and linear regression tasks compared to their corresponding baselines. It shows that a frozen attention mechanism generalizes across these tasks given task-specific prompts. Moreover, the frozen model exhibits lower variance across all tasks, suggesting increased stability and robustness. These results support our claim that a frozen softmax attention layer, when conditioned on task-specific prompts, emulates statistical algorithms in context without much performance degradation.

Table 2: **Comparison Between Baseline and Frozen Attention Layer on Synthetic Dataset.** We compare loss (MSE) as the mean and one standard deviation over 5 random seed runs for baseline vs. frozen model on different algorithms. We train on 50000 training data points evaluate on 10000 testing data points for each algorithm.

Model	**Lasso**	**Ridge Regression**	**Linear Regression**
Baseline	0.068±0.015	0.004±0.0003	0.147±0.067
Frozen Attention	0.059±0.001	0.071±0.0002	0.120±0.003

C.2 Real-World Experiment on Emulating Statistical Models

Objective: Real-World Emulation of Statistical Models. Building on Appendix C.1, we use real-world data to investigate the accuracy of a frozen softmax attention emulating algorithms.

Data Collection and Processing. We collect data from Ames Housing Dataset (De Cock, 2011). This dataset consists of 2930 observations and 79 features. We process the data by log-transforming the target variable, encoding categorical variables with one-hot vectors, replacing missing entries with median values, and standardizing numerical features. The resulting data consists of 262 features. We fit the processed data to Lasso, Ridge, and linear regression models to obtain algorithm weights as part of the input.

Model Architecture and Training. We use a mixture of statistical data to train a single-layer attention network with linear transformation. The input is passed through a multi-head attention layer with a linear transformation. We train the model for 300 epochs using the Adam optimizer with a learning rate of 0.001. We use 8 attention heads, a hidden dimension of 524, and a batch size of 32. After training, we freeze the attention weights, resulting in a fixed softmax attention layer. The frozen model is then evaluated on its ability to emulate various statistical algorithms using test data. We train the baseline models the same way as the synthetic experiment.

Table 3: **Comparison Between Baseline and Frozen Attention Layer on Ames Housing Dataset.** We compare loss (MSE) as the mean and one standard deviation over 5 random seed runs for baseline vs. frozen model on different algorithms. We train on 80% training data and evaluate on 20% testing data for each algorithm.

Model	Lasso	Ridge Regression	Linear Regression
Baseline	0.0354±0.0000	0.0132±0.0000	0.0288±0.0000
Frozen Attention	0.0322±0.0000	0.0252±0.0000	0.0250±0.0000

Results. As shown in Table 3, we compare mean MSE and standard deviation over 5 random seed runs for the frozen attention model against baseline for Lasso, Ridge, and linear regression on Ames Housing Data. The results shows the frozen attention model performs as well as the baseline models trained individually. We use an auxiliary network to approximate the required weight encoding. Our experiment validates that the mechanism works even when the exact weights are not supplied in real world scenarios.

C.3 Proof-of-Concept Experiment on Theorem 4.2

Objective: Verifying Handcrafted Frozen Attention Approximates Attention. We validate that the frozen attention prescribed in Theorem 4.2 approximates softmax attention with low error. In particular, we handcraft the weights as in the proof of Theorem 4.2.

Data Generation. We create a synthetic dataset. We randomly generate $X \in \mathbb{R}^{n\times d}$ drawn from a uniform distribution over $[-1, 1]$, $X \sim U(-1, 1)$. For each sample, we generate three weight matrices $W_K, W_Q, W_V \in \mathbb{R}^{n\times d}$ drawn from standard normal distribution $N(0, 1)$. We then compute $K = W_K X^\top, Q = W_Q X^\top, V = W_V X^\top \in \mathbb{R}^{n\times n}$. The true target attention output is therefore given by $Y = V\mathrm{Softmax}(K^\top Q) \in \mathbb{R}^{n\times n}$.

Model Architecture. Following the proof in Appendix D.6, we hard-wire the linear layer weights, attention weights, and interpolation points for the two-layer softmax attention module as our emulator. The model operates in a zero-shot, one-pass setting with no training or parameter updates.

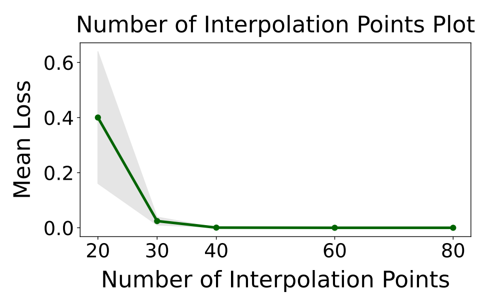

Figure 5: **Sensitivity of Handcrafted Attention Emulation to the Number of Interpolation Points.** We report loss (MSE) as the mean and one standard deviation (shaded region) over 4 sample data points. Each data point has sequence length $n = 12$ and input dimension $d = 24$. We set softmax temperature $\beta = 2$. We visualize the performance (MSE ± Std) for $P = 20, 30, 40, 60, 80$.

Results. We report the results in Table 4 and Figure 5. We compare the MSE loss between the emulator output and the target attention output. Specifically, we fix the number of data points n, input dimension d, softmax temperature β, and number of samples for testing. We vary the number of interpolation points P. The result validates our claim that the handcrafted frozen attention approximates the target attention. Moreover, we show that as P increases, the approximation error and standard deviation both further decrease.

Table 4: **Sensitivity to the Number of Interpolation Points.** We report MSE loss (mean± std) between outputs of handcrafted frozen attention and target attention varying number of interpolation points P over 4 samples. We choose $n = 12, d = 4, \beta = 2$, samples $= 4$ for evaluation.

P	20	30	40	60	80
Mean MSE	4.002×10^{-1}	2.442×10^{-2}	5.852×10^{-4}	5.770×10^{-9}	5.037×10^{-14}
Std	2.393×10^{-1}	1.451×10^{-2}	8.538×10^{-5}	1.994×10^{-9}	1.620×10^{-14}

D PROOFS OF MAIN TEXT

To prepare our proofs, we state the following axillary definitions and lemmas.

Definition D.1 (Truncated Linear Function)**.** We define the truncated linear function as follows:

$$\mathrm{Range}_{[a,b]}(x) = \begin{cases} a & x \le a, \\ x & a \le x \le b, \\ b & b \le x. \end{cases}$$

Intuitively, $\mathrm{Range}_{[a,b]}(\cdot)$ is the part of a linear function whose value is in $[a, b]$.

We then define the interpolation points in $[a, b]$ that are used in later proofs.

Definition D.2 (Interpolation)**.** Let $[a, b] \subset \mathbb{R}$ be an interval with $a \le b$ and let $p \in \mathbb{N}^*$ be a positive integer. We define

$$\widetilde{L}_0^{[a,b]} := a, \quad \widetilde{L}_p^{[a,b]} := b, \quad \widetilde{L}_i^{[a,b]} := a + \frac{i}{p}(b - a), \quad i = [p - 1].$$

Hence, $\widetilde{L}_0 < \widetilde{L}_1 < \cdots < \widetilde{L}_p$ forms a uniform partition of $[a, b]$. We also write

$$\Delta L := \widetilde{L}_i^{[a,b]} - \widetilde{L}_{i-1}^{[a,b]}, \quad i \in [p].$$

We often omit the superscript $[a, b]$ when the context is clear.

We also propose the following lemma to show Hardmax property that is capable of being approximated by Softmax.

Lemma D.1 (Lemma F.1 in (Hu et al., 2025a): Approximating Hardmax with Finite-Temperature Softmax)**.** Let $x = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^n$, $\epsilon > 0$. Define $\mathrm{Softmax}_\beta(\cdot)$ as

$$\mathrm{Softmax}_\beta(x) := [\frac{\exp(\beta x_1)}{\sum_{j=1}^n \exp(\beta x_j)}, \cdots, \frac{\exp(\beta x_n)}{\sum_{j=1}^n \exp(\beta x_j)}].$$

The following statements hold:

- **Case of a Unique Largest Entry.** Assume $x_1 = \max_{i\in[n]} x_i$ is unique, and $x_2 = \max_{i\in[n]\setminus\{1\}} x_i$. Then, if $\beta \ge (\ln(n-1) - \ln(\epsilon))/(x_1 - x_2)$, we have
$$\left\|\mathrm{Softmax}_\beta(x) - e_1\right\|_\infty \le \epsilon,$$
where $e_1 \in \mathbb{R}^n$ is the one-hot vector corresponding to to the maximal entry of x (i.e., x_1.)
- **Case of Two Largest Entries (Tied or Separated by δ).** Assume x_1 and x_2 are the first and second largest entries, respectively, with $\delta = x_1 - x_2 \ge 0$. Let x_3 be the third largest entry and is smaller than x_1 by a constant $\gamma > 0$ irrelevant to the input. Then, if $\beta \ge (\ln(n-2) - \ln\epsilon)/\gamma$, we have
$$\left\|\mathrm{Softmax}_\beta(x) - \frac{1}{1+e^{-\beta\delta}}e_1 - \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_2\right\|_\infty \le \epsilon.$$

The following technical lemma is used in the proof of Theorem D.1.

Lemma D.2 (Refined Version of Lemma F.2 in (Hu et al., 2025a): Cases of All Heads in Attn^H**).** For $a \in [\widetilde{L}_0, \widetilde{L}_{H(n-2)}]$. For any $h \in [H]$, define three cases of the relationship between a and h

- **Case 1:** $a \in [\widetilde{L}_{(h-1)(n-2)}, \widetilde{L}_{h(n-2)-1}]$,
- **Case 2:** $a \notin [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{h(n-2)}]$.
- **Case 3:** $a \in [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{(h-1)(n-2)}] \cup [\widetilde{L}_{h(n-2)-1}, \widetilde{L}_{h(n-2)}]$.

These cases includes all possible situation. Then for all h, only two cases exists
- a falls in Case 1 for an h and Case 2 for all others.
- a falls in Case 3 for two adjacent h and Case 2 for all others.

Proof. Because $a \in [\widetilde{L}_0, \widetilde{L}_{H(n-2)}]$ and

$$[\widetilde{L}_0, \widetilde{L}_{H(n-2)}] = \cup_{h=1}^{H} [\widetilde{L}_{(h-1)(n-2)}, \widetilde{L}_{h(n-2)}],$$

we have

$$a \in [\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)}] \tag{D.1}$$

for an arbitrary h_a.

This leads to only two possible cases

- Case 1*: $a \in [\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)-1}]$.
- Case 2*: $a \in [\widetilde{L}_{h_a(n-2)-1}, \widetilde{L}_{h_a(n-2)}]$.

Case 1*: $a \in [\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)-1}]$**.** Because $a \in [\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)-1}]$, for $h \neq h_a$, we have

$$\widetilde{L}_{h(n-2)-2}, \widetilde{L}_{h(n-2)} < \widetilde{L}_{(h_a-1)(n-2)}, \quad h < h_a$$
$$\widetilde{L}_{h(n-2)+1}, \widetilde{L}_{(h-1)(n-2)-1} \geq \widetilde{L}_{h_a(n-2)-1}, \quad h > h_a.$$

Thus

$$[\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)-1}] \cap [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{h(n-2)}] = \emptyset$$
$$[\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{h_a(n-2)-1}] \cap [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{h(n-2)}] = \emptyset$$

for all $h \neq h_a$.

This means that a does not fall into Case 1 nor Case 3 for other $h \in [H]$. Thus a has to fall into Case 2 for other h.

Case 2*: $a \in [\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{(h_a-1)(n-2)+1}] \cup [\widetilde{L}_{h_a(n-2)-1}, \widetilde{L}_{h_a(n-2)}]$**.** Without loss of generality, assume a to be in the left half $[\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{(h_a-1)(n-2)+1}]$. Because

$$[\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{(h_a-1)(n-2)+1}] = [\widetilde{L}_{(h_a-1)(n-2)-1}, \widetilde{L}_{(h_a-1)(n-2)}], \qquad \text{(Case 3 of } h_a - 1)$$
$$[\widetilde{L}_{(h_a-1)(n-2)}, \widetilde{L}_{(h_a-1)(n-2)+1}] = [\widetilde{L}_{(h_a-1)(n-2)-1}, \widetilde{L}_{(h_a-1)(n-2)}], \qquad \text{(Case 3 of } h_a)$$

this means a falls into Case 3 for h_a and $h_a - 1$.

This completes the proof. □

We are now ready to prove a refined version of (Hu et al., 2025a, Theorem 3.2).

Theorem D.1 (Multi-Head Attention Approximate Truncated Linear Models In-Context)**.** Let $X \in \mathbb{R}^{d \times n}$ be the input. Fix real numbers $a < b$, and let the truncation operator $\mathrm{Range}_{[a,b]}(\cdot)$ follow Definition D.1. Let w_s denote the linear coefficient of the in-context truncated linear model. Define W_s as

$$W_s := \begin{bmatrix} 0 \cdot w_s & 1 \cdot w_s & \cdots & (n-1) \cdot w_s \\ w_s & w_s & \cdots & w_s \end{bmatrix} \in \mathbb{R}^{2d \times n}.$$

For a precision parameter $p > n$ with $\epsilon = O(1/p)$, number of head $H = p/(n-2)$ there exists a single-layer, H-head self-attention Attn^H with a linear transformation $A : \mathbb{R}^{d \times n} \to \mathbb{R}^{(3d+n) \times n}$, such that $\mathrm{Attn}^H \circ A : \mathbb{R}^{d\times n} \to \mathbb{R}^{d_o\times n}$ satisfies, for any $i \in [n]$,

$$\|\mathrm{Attn}^H \circ A(X)_{:,i} - \mathrm{Range}_{[a,b]}(w_s^\top x_i) e_{\widetilde{k}_i}\|_\infty \le \underbrace{\max\{|a|,|b|\}\cdot\epsilon_0}_{\text{finite-}\beta\text{ softmax error}} + \underbrace{\frac{b-a}{(n-2)H}}_{\text{interpolation error}}.$$

Here $e_{\widetilde{k}_i}$ is the one-hot vector with a 1 at position $\widetilde{k}_i$-th index and 0 elsewhere, and

$$\widetilde{k}_i = G(k_i) \in [d_o], \quad \text{with} \quad k_i = \underset{k\in\{0,1,\cdots,p-1\}}{\mathrm{argmin}} \ (-2x_i^\top w_i - 2t_i + \widetilde{L}_0 + \widetilde{L}_k)\cdot k,$$

where $G : [p] \to [d_o]$ denotes any set-to-constant function sending each selected interpolation index k_i into an appropriate integer $\widetilde{k}_i \in [d_o]$ for $i \in [n]$.

Proof. Define $A : \mathbb{R}^{d\times n} \to \mathbb{R}^{(3d+n)\times n}$ for the input sequence X as

$$A(X) := \underbrace{\begin{bmatrix} I_{3d} \\ 0_{n\times 3d}\end{bmatrix}}_{\text{token-wise linear}} \begin{bmatrix} X \\ W_s \end{bmatrix} + \underbrace{\begin{bmatrix} 0_{3d\times n} \\ I_n \end{bmatrix}}_{\text{positional encoding}} = \begin{bmatrix} X \\ W_s \\ I_n \end{bmatrix} \in \mathbb{R}^{(3d+n)\times n}.$$

Thus, A is a token-wise linear layer augmented with positional encoding, as it applies a linear projection to each token and then adds a unique per-token bias.

Let p be a precision parameter, without loss of generality, let it be divisible by $n-2$ and denote $p/(n-2)$ as H.

Now we define the multi-head attention Attn of H heads. Denote $\ell_k := k(\widetilde{L}_k + \widetilde{L}_0)$ for $k \in [p]$ following Definition D.2. We denote the h-th head as Attn_h, and define the weight matrices as

$$W_K^{(h)} = -\beta \begin{bmatrix} 0_{d\times d} & -2I_d & -2[(h-1)(n-2)-1]I_d & 0 & 0 & \cdots & 0 \\ 0_{1\times d} & 0_{1\times d} & 0_{1\times d} & \ell_{(h-1)(n-2)-1} & \ell_{(h-1)(n-2)} & \cdots & \ell_{h(n-2))} \end{bmatrix},$$

$$W_Q^{(h)} = \begin{bmatrix} I_d & 0_{d\times 2d} & 0_{d\times n} \\ 0_{1\times d} & 0_{1\times 2d} & 1_{1\times n}\end{bmatrix},$$

$$W_V^{(h)} = \begin{bmatrix} 0_{d_o\times(3d+1)} & \widetilde{L}_{(h-1)(n-2)} e_{\widetilde{k}_{(h-1)(n-2)}} & \widetilde{L}_{(h-1)(n-2)+1} e_{\widetilde{k}_{(h-1)(n-2)+1}} & \cdots & \widetilde{L}_{h(n-2)-1} e_{\widetilde{k}_{h(n-2)-1}} & 0_{d_o}\end{bmatrix},$$

for every $h \in [H]$.

Here $\beta > 0$ is a coefficient we use to control the precision of our approximation. The attention reaches higher precision as β gets larger.

With the construction of weights, we are also able to calculate the $K,\ Q,\ V$ matrices in Attn

$$\begin{aligned} K^{(h)} &:= W_K^{(h)} A(X) \\ &= W_K^{(h)} \cdot \begin{bmatrix} X \\ W_s \\ I_n \end{bmatrix} \\ &= W_K^{(h)} \cdot \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ 0\cdot w_s & 1\cdot w_s & \cdots & (n-1)\cdot w_s \\ w_s & w_s & \cdots & w_s \\ e_1^{(n)} & e_2^{(n)} & \cdots & e_n^{(n)} \end{bmatrix} \\ &= -\beta \begin{bmatrix} 0_{d\times d} & -2I_d & -2[(h-1)(n-2)-1]I_d & 0 & 0 & \cdots & 0 \\ 0_{1\times d} & 0_{1\times d} & 0_{1\times d} & \ell_{(h-1)(n-2)-1} & \ell_{(h-1)(n-2)} & \cdots & \ell_{h(n-2))} \end{bmatrix} \\ &\quad \cdot \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ 0\cdot w_s & 1\cdot w_s & \cdots & (n-1)\cdot w_s \\ w_s & w_s & \cdots & w_s \\ e_1^{(n)} & e_2^{(n)} & \cdots & e_n^{(n)} \end{bmatrix} \end{aligned}$$

$$= -\beta \begin{bmatrix} -2[(h-1)(n-2)-1]w_s & -2(h-1)(n-2)w_s & \cdots & -2h(n-2)w_s \\ \ell_{(h-1)(n-2)-1} & \ell_{(h-1)(n-2)} & \cdots & \ell_{h(n-2)} \end{bmatrix} \in \mathbb{R}^{(d+1)\times n}, \tag{D.2}$$

where the last equality comes from multiplying X with 0, thus this is a extraction of non-zero entries in W_K.

For Q, we have

$$\begin{aligned} Q^{(h)} &:= W_Q^h A(X) \\ &= \begin{bmatrix} I_d & 0_{d\times 2d} & 0_{d\times n} \\ 0_{1\times d} & 0_{1\times 2d} & 1_{1\times n} \end{bmatrix} \cdot \begin{bmatrix} X \\ W_s \\ I_n \end{bmatrix} \\ &= \begin{bmatrix} I_d \cdot X + 0_{d\times 2d}\cdot W_s + 0_{d\times n}\cdot I_n \\ 0_{1\times d}\cdot X + 0_{1\times 2d}\cdot W_s + 1_{1\times n}\cdot I_n \end{bmatrix} \\ &= \begin{bmatrix} X \\ 1_{1\times n} \end{bmatrix} \in \mathbb{R}^{(d+1)\times n}. \end{aligned} \tag{D.3}$$

For V, we have

$$\begin{aligned} V^{(h)} &:= W_V^{(h)} A(X) \\ &= \begin{bmatrix} 0_{d_o\times(3d+1)} & \widetilde{L}_{(h-1)(n-2)} e_{\widetilde{k}_{(h-1)(n-2)}} & \cdots & \widetilde{L}_{h(n-2)-1} e_{\widetilde{k}_{h(n-2)-1}} & 0_{d_o} \end{bmatrix} \cdot \begin{bmatrix} X \\ W_s \\ I_n \end{bmatrix} \\ &= \underbrace{0}_{d_o\times 3d} \cdot \begin{bmatrix} X \\ W_s \end{bmatrix} + \underbrace{\begin{bmatrix} 0_{d_o} & \widetilde{L}_{(h-1)(n-2)} e_{\widetilde{k}_{(h-1)(n-2)}} & \cdots & \widetilde{L}_{h(n-2)-1} e_{\widetilde{k}_{h(n-2)-1}} & 0_{d_o} \end{bmatrix}}_{d_o\times n} \cdot I_n \\ &= \begin{bmatrix} 0_{d_o} & \widetilde{L}_{(h-1)(n-2)} e_{\widetilde{k}_{(h-1)(n-2)}} & \widetilde{L}_{(h-1)(n-2)+1} e_{\widetilde{k}_{(h-1)(n-2)+1}} & \cdots & \widetilde{L}_{h(n-2)-1} e_{\widetilde{k}_{h(n-2)-1}} & 0_{d_o} \end{bmatrix} \\ &\in \mathbb{R}^{d_o\times n}. \end{aligned} \tag{D.4}$$

Given that all $\widetilde{k}_j$, where $j \in [p]$, share the same number in $[d_o]$, we denote this number by k_G.

Hence we rewrite $V^{(h)}$ as

$$V^{(h)} = \begin{bmatrix} 0_{d_o} & \widetilde{L}_{(h-1)(n-2)} e_{k_G} & \widetilde{L}_{(h-1)(n-2)+1} e_{k_G} & \cdots & \widetilde{L}_{h(n-2)-1} e_{k_G} & 0_{d_o} \end{bmatrix}.$$

We define m_v as

$$m_v := \max\{|a|, |b|\}.$$

By the definition of $V^{(h)}$, we have

$$\|V\|_\infty \le \max_{i\in[P]}\{\widetilde{L}_i\} \le m_v. \tag{D.5}$$

Remark D.1 (Intuition of the Construction of $V^{(h)}$). As previously mentioned, $\widetilde{L}_i$, for $i \in [p]$, are all the interpolation points. In this context, $V^{(h)}$ encompasses the $(n-2)$ elements of these interpolations (i.e., $(h-1)(n-2)$ to $h(n-2)-1$). Meanwhile, the value on the two ends of V^h are both set to 0_{d_o}, because we suppress the head and let it output 0 when the input X is not close enough to the interpolations of the head.

Now we are ready to calculate the output of each Attn_h

$$\begin{aligned} &\mathrm{Attn}_h(A(X)) \\ &= V^{(h)} \mathrm{Softmax}((K^{(h)})^\top Q^{(h)}) \\ &= V^{(h)} \mathrm{Softmax}\left(-\beta \begin{bmatrix} -2[(h-1)(n-2)-1]w & -2(h-1)(n-2)w & \cdots & -2h(n-2)w \\ \ell_{(h-1)(n-2)-1} & \ell_{(h-1)(n-2)} & \cdots & \ell_{h(n-2)} \end{bmatrix}^\top \begin{bmatrix} X \\ 1_{1\times n} \end{bmatrix} \right), \end{aligned}$$

where last line is by plug in (D.2) and (D.3). Note the i-th column of the attention score matrix (the $\mathrm{Softmax}$ nested expression) is equivalent to the following expressions

$$
\begin{aligned}
&\mathrm{Softmax}((K^{(h)})^\top Q^{(h)})_{:,i} \\
=& \mathrm{Softmax}\left(-\beta \begin{bmatrix} -2[(h-1)(n-2)-1]w & -2(h-1)(n-2)w & \cdots & -2h(n-2)w \\ \ell_{(h-1)(n-2)-1} & \ell_{(h-1)(n-2)} & \cdots & \ell_{h(n-2)} \end{bmatrix}^\top \begin{bmatrix} X \\ 1_{1\times n} \end{bmatrix} \right)_{:,i} \\
=& \mathrm{Softmax}\left(-\beta \begin{bmatrix} -2[(h-1)(n-2)-1]w^\top x_i + \ell_{(h-1)(n-2)-1} \\ -2(h-1)(n-2)w^\top x_i + \ell_{(h-1)(n-2)} \\ \vdots \\ -2h(n-2)w^\top x_i + \ell_{h(n-2)} \end{bmatrix} \right) \quad \text{(pick column } i\text{)} \\
=& \mathrm{Softmax}\left(-\beta \begin{bmatrix} [(h-1)(n-2)-1](-2w^\top x_i + \widetilde{L}_{(h-1)(n-2)-1} + \widetilde{L}_0) - 2[(h-1)(n-2)-1]t \\ (h-1)(n-2)(-2w^\top x_i + \widetilde{L}_{(h-1)(n-2)} + \widetilde{L}_0) - 2(h-1)(n-2)t \\ \vdots \\ h(n-2)(-2w^\top x_i + \widetilde{L}_{h(n-2)} + \widetilde{L}_0) - 2h(n-2)t \end{bmatrix} \right) \\
& \qquad \text{(By } \ell_k = k(\widetilde{L}_k + \widetilde{L}_0) - 2kt\text{)} \\
=& \mathrm{Softmax}\left(-\frac{\beta}{\Delta L} \begin{bmatrix} (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{(h-1)(n-2)-1}) \cdot [(h-1)(n-2)-1]\Delta L \\ (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{(h-1)(n-2)}) \cdot (h-1)(n-2)\Delta L \\ \vdots \\ (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{h(n-2)}) \cdot h(n-2)\Delta L \end{bmatrix} \right) \\
& \qquad \text{(By mutiplying and dividing by } \Delta L\text{)} \\
=& \mathrm{Softmax}\left(-\frac{\beta}{\Delta L} \begin{bmatrix} (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{(h-1)(n-2)-1}) \cdot (\widetilde{L}_{(h-1)(n-2)-1} - \widetilde{L}_0) \\ (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{(h-1)(n-2)}) \cdot (\widetilde{L}_{(h-1)(n-2)} - \widetilde{L}_0) \\ \vdots \\ (-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_{h(n-2)}) \cdot (\widetilde{L}_{h(n-2)} - \widetilde{L}_0) \end{bmatrix} \right) \\
& \qquad \text{(By } k\Delta L = \widetilde{L}_k - \widetilde{L}_0\text{)} \\
=& \mathrm{Softmax}\left(-\frac{\beta}{\Delta L} \begin{bmatrix} (-2x_i^\top w - 2t) \cdot \widetilde{L}_{(h-1)(n-2)-1} + (\widetilde{L}_{(h-1)(n-2)-1})^2 + (x_i^\top w + t)^2 \\ (-2x_i^\top w - 2t) \cdot \widetilde{L}_{(h-1)(n-2)} + (\widetilde{L}_{(h-1)(n-2)})^2 + (x_i^\top w + t)^2 \\ \vdots \\ (-2x_i^\top w - 2t) \cdot \widetilde{L}_{h(n-2)} + (\widetilde{L}_{h(n-2)})^2 + (x_i^\top w + t)^2 \end{bmatrix} \right) \\
=& \mathrm{Softmax}\left(-\frac{\beta}{\Delta L} \begin{bmatrix} (x_i^\top w + t - \widetilde{L}_{(h-1)(n-2)-1})^2 \\ (x_i^\top w + t - \widetilde{L}_{(h-1)(n-2)})^2 \\ \vdots \\ (x_i^\top w + t - \widetilde{L}_{h(n-2)})^2 \end{bmatrix} \right). \qquad \text{(D.6)}
\end{aligned}
$$

Here, the second-last equality arises from the fact that the softmax function is shift-invariant, allowing us to subtract and add a constant across all coordinates. To be more precise, we first expand the product for k-th coordinate of the column vector

$$
\begin{aligned}
&(-2x_i^\top w - 2t + \widetilde{L}_0 + \widetilde{L}_k)(\widetilde{L}_k - \widetilde{L}_0) \\
=& (-2x_i^\top w - 2t)L_k + L_0 L_k + L_k^2 - (-2x_i^\top w - 2t)L_0 - L_0^2 - L_0 L_k \\
=& (-2x_i^\top w - 2t)L_k + L_k^2 - \underbrace{(-2x_i^\top w - 2t)L_0 - L_0^2}_{\text{constant across the column vector}}.
\end{aligned}
$$

Then, dropping the constant and adding another constant $(x_i^\top w + t)^2$ across all coordinates, the above equation becomes

$$
(-2x_i^\top w - 2t)L_k + L_k^2 + (x_i^\top w + t)^2 = (x_i^\top w + t - L_k)^2.
$$

Hence we finish the derivation of (D.6). Thus we have

$$\mathrm{Attn}_h(A(X))_{:,i} = V^{(h)}\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(x_i^\top w + t - \widetilde{L}_{(h-1)(n-2)-1})^2\\(x_i^\top w + t - \widetilde{L}_{(h-1)(n-2)})^2\\\vdots\\(x_i^\top w + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right). \tag{D.7}$$

For a specific h, we calculate the result of (D.7) column by column. Let X_i denote any column (token) of the matrix X. We partition the situation at each column (token) into three distinct cases:

- **Case 1:** $w^\top X_i + t$ is strictly within the interpolation range of Attn_h ($X \in [\widetilde{L}_{(h-1)(n-2)}, \widetilde{L}_{h(n-2)-1}]$). This excludes the following range at the edge of the interpolation range of
$$[\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{(h-1)(n-2)}] \cup [\widetilde{L}_{h(n-2)-1}, \widetilde{L}_{h(n-2)}].$$
- **Case 2:** $w^\top X_i + t$ is not within the interpolation range of Attn_h:
$$w^\top X_i + t \notin [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{h(n-2)}].$$
- **Case 3:** $w^\top X_i + t$ is on the edge (region) of the interpolation range of Attn_h:
$$w^\top X_i + t \in [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{(h-1)(n-2)}] \cup [\widetilde{L}_{h(n-2)-1}, \widetilde{L}_{h(n-2)}].$$

Two remarks are in order.

Remark D.2 (Cases of a Single Head Attention)**.** The H heads split the approximation of the truncated linear map across disjoint intervals. For head h,

$$\|\mathrm{Attn}_h(X) - \mathrm{Range}_{[a+\frac{b-a}{p}((h-1)(n-2)-1), a+\frac{b-a}{p}h(n-2)]}(X)\|_\infty \le \epsilon_1,$$

where $\epsilon > 0$ is arbitrarily small.
With this understanding, $w^\top X_i + t$:
- **Case 1:** falls into the interior of the interpolation range of the h-th head Attn_h, denoted as $\mathrm{Range}_{[a+(b-a)((h-1)(n-2)-1)/p, a+(b-a)h(n-2)/p]}$.
- **Case 2:** remains outside of the interpolation range of the h-th head Attn_h.
- **Case 3:** falls on the boundary of the interpolation range of the h-th head Attn_h.

Remark D.3 (Cases of All Attention Heads)**.** According to Lemma D.2, for all heads in Attn^H, there are two possible cases:
- **Case 1*:** x falls into Case 1 for a head, and Case 2 for all other heads.
- **Case 2*:** x falls into Case 3 for two heads with adjacent interpolation ranges, and Case 2 for other heads.

This also means that when Case 1 appears in Attn^H, the situation of all head in Attn^H falls into Case 1*. And when Case 3 appears in Attn^H, the situation of all head in Attn^H falls into Case 2*. Thus, We discuss Case 2* in the discussion of Case 3.

Case 1: $X_i \in [\widetilde{L}_{(h-1)(n-2)}, \widetilde{L}_{h(n-2)-1}]$**.** In this case, our goal is to demonstrate this attention head outputs a value close to $\mathrm{Range}_{[a,b]}(w^\top X_i + t)$.

Let $\widetilde{L}_s$ and $\widetilde{L}_{s+1}$ be the two interpolants such that

$$w^\top X_i + t \in [\widetilde{L}_s, \widetilde{L}_{s+1}]. \tag{D.8}$$

Then, s and $s+1$ are also the labels of the two largest entries in

$$-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2\\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2\\ \vdots\\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix},$$

since

$$\begin{aligned}&\underset{k\in\{(h-1)(n-2)-1, h(n-2)\}}{\operatorname{argmax}} -\frac{\beta}{\Delta L}(w^\top X_i + t - \widetilde{L}_k)^2\\ =&\underset{k\in\{(h-1)(n-2)-1, h(n-2)\}}{\operatorname{argmin}} (w^\top X_i + t - \widetilde{L}_k)^2\\ =&\underset{k\in\{(h-1)(n-2)-1, h(n-2)\}}{\operatorname{argmin}} |w^\top X_i + t - \widetilde{L}_k|.\end{aligned}$$

We also note that the distance of $w^\top X_i + t$ to interpolants beside $\widetilde{L}_s$ and $\widetilde{L}_{s+1}$ differs from $w^\top X_i + t$ for at least $\widetilde{L}_s - \widetilde{L}_{s-1} = (b-a)/p$ or $\widetilde{L}_{s+1} - \widetilde{L}_s = (b-a)/p$.

This is equivalent to the occasion when $x_1 - x_3$ in Lemma D.1 is larger than

$$\begin{aligned}&\max\left\{\frac{\beta}{\Delta L}(w^\top X_i + t - \widetilde{L}_{s-1})^2 - (w^\top X_i + t - \widetilde{L}_s)^2, \frac{\beta}{\Delta L}(w^\top X_i + t - \widetilde{L}_{s+2})^2 - (w^\top X_i + t - \widetilde{L}_{s+1})^2\right\}\\ &\geq \frac{\beta}{\Delta L}\cdot(\frac{b-a}{p})^2,\end{aligned}$$

which is invariant to X_i.

Thus according to Lemma D.1 and the fact that the s and $s+1$ are the two largest entries in the i-th column of the attention score matrix, we have

$$\left\|\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2\\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2\\ \vdots\\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right) - \frac{1}{1+e^{-\beta\delta}}\underbrace{e_s}_{n\times 1} - \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}\underbrace{e_{s+1}}_{n\times 1}\right\|_\infty \leq \epsilon_2,$$

for any $\epsilon_2 > 0$.

This yields that

$$\begin{aligned}&\left\|V\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2\\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2\\ \vdots\\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right) - V\frac{1}{1+e^{-\beta\delta}}e_s - V\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{s+1}\right\|_\infty\\ &\leq \left\|\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2\\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2\\ \vdots\\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right) - \frac{1}{1+e^{-\beta\delta}}e_s - \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{s+1}\right\|_\infty\cdot\|V\|_\infty\\ &\leq \|V\|_\infty\epsilon_2.\end{aligned}$$

This is equivalent to

$$\begin{aligned}&\|V\mathrm{Softmax}(K^\top Q)_{:,i} - \frac{1}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s-1}e_{k_G} - \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s}e_{k_G}\|_\infty\\ &\leq \|V\|_\infty\cdot\epsilon_2 && (\text{By } \|AB\|\leq\|A\|\cdot\|B\|)\\ &\leq m_v\epsilon_2, && \text{(D.9)}\end{aligned}$$

where the last line is by (D.5).

From (D.8), we derive that

$$\begin{aligned}
&\|\frac{1}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s-1} + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s} - (w^\top X_i + t)e_{k_G}\|_\infty \\
&\le \|\frac{1}{1+e^{-\beta\delta}}(\widetilde{L}_{(h-1)(n-2)+s-1} - (w^\top X_i + t)e_{k_G})\|_\infty + \|\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}(\widetilde{L}_{(h-1)(n-2)+s} - (w^\top X_i + t))\|_\infty \\
&\qquad \text{(By convex combination of } (w^\top X_i + t) \text{ and triangle inequality)} \\
&\le \frac{1}{1+e^{-\beta\delta}} \cdot \frac{b-a}{p} + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}} \cdot \frac{b-a}{p} \qquad \text{(By (D.8))} \\
&= \frac{b-a}{p}. \qquad \text{(D.10)}
\end{aligned}$$

Combing (D.9) and (D.10) yields

$$\begin{aligned}
&\|V\mathrm{Softmax}(K^\top Q)_{:,i} - (w^\top X_i + t)\|_\infty \\
&\le \|V\mathrm{Softmax}(K^\top Q)_{:,i} - \frac{1}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s-1} - \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s}\|_\infty \\
&\quad + \|\frac{1}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s-1} + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}\widetilde{L}_{(h-1)(n-2)+s} - (w^\top X_i + t)e_{k_G}\|_\infty \\
&\qquad \text{(By triangle inequality)} \\
&\le m_v\epsilon_2 + \frac{b-a}{p}, \qquad \text{(D.11)}
\end{aligned}$$

where the first inequality comes from adding and subtracting the interpolation points' convex combination and then applying triangle inequality.

Case 2: $X \notin [\widetilde{L}_{(h-1)(n-2)-1}, \widetilde{L}_{h(n-2)}]$**.** In this case, X_i falls out of the range of interpolation covered by Attn_h.

Without loss of generality, suppose $w^\top X_i + t$ to lie left to the range of interpolation of Attn_h.

This yields that $\widetilde{L}_{(h-1)(n-2)-1}$ is the closest interpolant within Attn_h to $w^\top X_i + t$. Furthermore, the second closest interpolant $\widetilde{L}_{(h-1)(n-2)}$ is at least further for at least $(b-a)/p$, which is a constant irrelevant to X_i

Then by Lemma D.1, we have

$$\Big\|\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2 \\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2 \\ \vdots \\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right) - \underbrace{e_1}_{n\times 1}\Big\|_\infty \le \epsilon_3,$$

for any $\epsilon_3 > 0$.

This yields that

$$\begin{aligned}
&\|V\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)-1})^2 \\ (w^\top X_i + t - \widetilde{L}_{(h-1)(n-2)})^2 \\ \vdots \\ (w^\top X_i + t - \widetilde{L}_{h(n-2)})^2\end{bmatrix}\right) - V\underbrace{e_1}_{n\times 1}\|_\infty \\
&\le \|V\|_\infty \cdot \epsilon_3 \qquad \text{(By } \|AB\| \le \|A\|\cdot\|B\|\text{)} \\
&\le m_v\epsilon_3,
\end{aligned}$$

where the last line is by (D.5).

This is equivalent to

$$\left\|V\mathrm{Softmax}\left(-\frac{\beta}{\Delta L}\begin{bmatrix}(w^\top X_i+t-\widetilde{L}_{(h-1)(n-2)-1})^2\\(w^\top X_i+t-\widetilde{L}_{(h-1)(n-2)})^2\\\vdots\\(w^\top X_i+t-\widetilde{L}_{h(n-2)})^2\end{bmatrix}\right)-0_{d_o}\right\|_\infty\leq m_v\epsilon_3. \tag{D.12}$$

Case 1*. According to Lemma D.2, when Case 1 occurs for one head in the H heads of Attn^H, all other head will be in Case 2.

Combining with the result in Case 2, we have the output of all heads as

$$\begin{aligned}&\|\mathrm{Attn}^H(A(X))_{:,i}-(w^\top X_i+t)e_{k_G}\|_\infty\\&=\|\sum_{h_0\in[H]/\{h\}}\mathrm{Attn}_{h_0}\circ A(X)_{:,i}\|_\infty+\|\mathrm{Attn}_h\circ A(X)_{:,i}-(w^\top X_i+t)e_{k_G}\|_\infty\\&=(H-1)m_v\epsilon_3+m_v\epsilon_2+\frac{b-a}{p} && \text{(By (D.11) and (D.12))}\\&=(H-1)m_v\epsilon_3+m_v\epsilon_2+\frac{b-a}{H(n-2)}.\end{aligned}$$

Setting ϵ_2,ϵ_3 to be

$$\epsilon_2=\frac{\epsilon_0}{2}$$
$$\epsilon_3=\frac{\epsilon_0}{2(H-1)m}$$

yields the final result.

Case 3 (and Case 2*): $X\in[\widetilde{L}_{(h-1)(n-2)-1},\widetilde{L}_{(h-1)(n-2)}]\cup[\widetilde{L}_{h(n-2)-1},\widetilde{L}_{h(n-2)}]$. In this case, $w^\top X_i+t$ is the boundary of the interpolation range of Attn_{h_0}. By Lemma D.2, it should also fall on the boundary of a head with neighboring interpolation range. Without loss of generality, we set it to be Attn_{h_0-1}. Furthermore, Lemma D.2 indicates that $w^\top X_i+t$ should fall on no other interpolation range of any heads beside Attn_{h_0} and Attn_{h_0-1}.

Combining this with case 2, we have

$$\begin{aligned}\mathrm{Attn}^H(A(X))_{:,i}&=\sum_{h=1}^{H}\mathrm{Attn}_h\circ A(X)_{:,i}\\&\in[(-(H-2)m_v\epsilon_3+\mathrm{Attn}_{h_0}\circ A(X)_{:,i}+\mathrm{Attn}_{h_0-1}\circ A(X)_{:,i}),\\&\quad((H-2)m_v\epsilon_3+\mathrm{Attn}_{h_0}\circ A(X)_{:,i}+\mathrm{Attn}_{h_0-1}\circ A(X)_{:,i})].\\& && \text{(By (D.12))}\end{aligned}$$

By Lemma D.1, let δ denote

$$\delta=\widetilde{L}_{(h-1)(n-2)+s}-(w^\top X_i+t)e_{k_G}-[\widetilde{L}_{(h-1)(n-2)+s}-(w^\top X_i+t)e_{k_G}],$$

we have

$$\|\mathrm{Softmax}((K^{(h)})^\top Q^{(h)})-(\frac{1}{1+e^{-\beta\delta}}e_1+\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_2)\|\leq\epsilon_4,$$

and

$$\|\mathrm{Softmax}((K^{(h-1)})^\top Q^{(h-1)})-(\frac{1}{1+e^{-\beta\delta}}e_{n-1}+\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_n)\|\leq\epsilon_5,$$

for any $\epsilon_4,\epsilon_5>0$.

Thus we have

$$\|V^{(h)}\mathrm{Softmax}((K^{(h)})^\top Q^{(h)})+V^{(h-1)}\mathrm{Softmax}((K^{(h-1)})^\top Q^{(h-1)})$$

$$- V(\frac{1}{1+e^{-\beta\delta}}e_1 + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_2 + \frac{1}{1+e^{-\beta\delta}}e_{n-1} + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_n)\|_\infty$$
$$\leq \|V\|_\infty(\epsilon_4+\epsilon_5).$$

This is equivalent to

$$\|V^{(h)}\mathrm{Softmax}((K^{(h)})^\top Q^{(h)}) + V^{(h-1)}\mathrm{Softmax}((K^{(h-1)})^\top Q^{(h-1)})$$
$$- (\frac{1}{1+e^{-\beta\delta}}\cdot 0 + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1} + \frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G})\cdot 0\|_\infty$$
$$\leq \|V\|_\infty\cdot(\epsilon_4+\epsilon_5).$$

Thus we have

$$\|V^{(h)}\mathrm{Softmax}((K^{(h)})^\top Q^{(h)}) + V^{(h-1)}\mathrm{Softmax}((K^{(h-1)})^\top Q^{(h-1)})$$
$$- (\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1})\|_\infty$$
$$\leq \|V\|_\infty(\epsilon_4+\epsilon_5),$$

which implies

$$\|\sum_{h=1}^{H}\mathrm{Attn}_h(A(X))_{:,i} - (\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1})\|_\infty$$
$$\leq (H-2)m_v\epsilon_3 + \|V\|_\infty(\epsilon_4+\epsilon_5). \tag{D.13}$$

Finally, since

$$\|\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1} - (w^\top X_i + t)e_{k_G}\|_\infty \leq \frac{b-a}{p},$$

(By (D.10))

combining with (D.13), we have

$$\|\sum_{h=1}^{H}\mathrm{Attn}_h(A(X))_{:,i} - (w^\top X_i + t)e_{k_G}\|_\infty$$
$$\leq \|\sum_{h=1}^{H}\mathrm{Attn}_h(A(X))_{:,i} - (\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1})\|_\infty$$
$$+ \|(\frac{e^{-\beta\delta}}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s} + \frac{1}{1+e^{-\beta\delta}}e_{k_G}\widetilde{L}_{(h-1)(n-2)+s-1}) - (w^\top X_i + t)e_{k_G}\|_\infty$$

(By triangle inequality)

$$\leq \frac{b-a}{p} + (H-2)m_v\epsilon_3 + \|V\|_\infty(\epsilon_4+\epsilon_5)$$
$$\leq \frac{b-a}{H(n-2)} + (H-2)\max\{|a|,|b|\}\epsilon_3 + \max\{|a|,|b|\}(\epsilon_4+\epsilon_5).$$

Setting $\epsilon_3, \epsilon_4, \epsilon_5$ to be

$$\epsilon_3 = \frac{\epsilon_0}{3(H-2)}$$
$$\epsilon_4 = \epsilon_5 = \frac{\epsilon_0}{3}$$

yields the final result.

This completes the proof. □

Lemma D.3 (Attention Prepended with Token-Wise Linear Transformation is Still a Transformer)**.** For any attention Attn and any linear transformation A, $\mathrm{Attn}\circ A$ is still an attention.

Proof. We denote the transformation matrix of A also as M_A. Denote the attention Attn as

$$\mathrm{Attn}(Z) := W_V Z \mathrm{Softmax}((W_K Z)^\top W_Q Z).$$

Then we have

$$\mathrm{Attn} \circ A(Z) = W_V M_A Z \mathrm{Softmax}((W_K M_A Z)^\top W_Q M_A Z).$$

It is a new attention with parameters $W_K M_A$,$W_Q M_A$ and $W_V M_A$. □

Lemma D.4 (Lemma 14 in (Bai et al., 2023): Composition of Error for Approximating Convex GD)**.** Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is a convex function. Let $w^\star \in \mathrm{argmin}_{w \in \mathbb{R}^d} f(w)$, $R \geq 2\|w^\star\|_2$, and assume that ∇f is L_f-smooth on $B_2^d(R)$. Let sequences $\{\widehat{w}^\ell\}_{\ell \geq 0} \subset \mathbb{R}^d$ and $\{w_{\mathrm{GD}}^\ell\}_{\ell \geq 0} \subset \mathbb{R}^d$ be given by $\widehat{w}^0 = w_{\mathrm{GD}}^0 = \mathbf{0}$,

$$\begin{cases} \widehat{w}^{\ell+1} = \widehat{w}^\ell - \eta \nabla f(\widehat{w}^\ell) + \epsilon^\ell, & \|\epsilon^\ell\|_2 \leq \epsilon, \\ w_{\mathrm{GD}}^{\ell+1} = w_{\mathrm{GD}}^\ell - \eta \nabla f(w_{\mathrm{GD}}^\ell), \end{cases}$$

for all $\ell \geq 0$. Then as long as $\eta \leq 2/L_f$, for any $0 \leq L \leq R/(2\epsilon)$, it holds that $\|\widehat{w}^L - w_{\mathrm{GD}}^L\|_2 \leq L\epsilon$ and $\|\widehat{w}^L\|_2 \leq \frac{R}{2} + L\epsilon \leq R$.

Corollary D.1.1 (Corollary A.2 in (Bai et al., 2023): Gradient Descent for Smooth and Strongly Convex Function)**.** Suppose $L : \mathbb{R}^d \to \mathbb{R}$ is a α-strongly convex and β-smooth for som $0 < \alpha \leq \beta$. Then, the gradient descent iterates $w_{\mathrm{GD}}^{(t+1)} := w_{\mathrm{GD}}^t - \eta \nabla L(w_{\mathrm{GD}}^L)$ with learning rage $\eta = 1/\beta$ and initialization $w_{\mathrm{GD}}^0 \in \mathbb{R}^d$ satisfies for any $t \geq 1$,

$$\|w_{\mathrm{GD}}^t - w^\star\|_2^2 \leq \exp\left(-\frac{t}{\kappa}\right) \cdot \|w_{\mathrm{GD}}^0 - w^\star\|_2^2,$$

$$L(w_{\mathrm{GD}}^t) - L(w^\star) \leq \frac{\beta}{2} \exp\left(-\frac{t}{\kappa}\right) \cdot \|w_{\mathrm{GD}}^0 - w^\star\|_2^2,$$

where $\kappa := \beta/\alpha$ is the condition number of L, and $w^\star := \mathrm{argmin}_{w \in \mathbb{R}^d} L(w)$.

D.1 Proof of Theorem 3.1

Definition D.3 (Interpolation Points)**.** Define $P+1$ interpolation points of the effective domain of f, i.e., the range of $w^\top x - y$, as

$$L_j := L_{\min} + \frac{j}{P}(L_{\max} - L_{\min}), \quad \text{for} \quad j \in 0, 1, \ldots, P,$$

where $[L_{\min}, L_{\max}]$ is a bounded interval containing all values of $w^\top x - y$.

Theorem D.2 (In-Context Emulation of $f(w^\top x - y)x$ with Single-Head Attention; Theorem 3.1 Restate)**.** Let $[L_{\min}, L_{\max}]$ be a bounded interval containing all values of $w^\top x - y$, and let

$$X := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{(d+1) \times n} \quad \text{and} \quad W := [w \quad w \quad \cdots \quad w] \in \mathbb{R}^{d \times n},$$

where $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$, and $w \in \mathbb{R}^d$ is the coefficient vector. Define the input as:

$$Z := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ w & w & \cdots & w \end{bmatrix} = \begin{bmatrix} X \\ W \end{bmatrix} \in \mathbb{R}^{(2d+1) \times n}. \tag{D.14}$$

Assume $\max\{\|X\|_\infty, \|W\|_\infty\} \leq B$. For any continuously differentiable function $f : \mathbb{R} \to \mathbb{R}$ and any $\epsilon > 0$, there exists a single-head attention Attn_s with a linear layer Linear such that

$$\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z) - \begin{bmatrix} f(w^\top x_1 - y_1)x_1 & \cdots & f(w^\top x_n - y_n)x_n \end{bmatrix}\|_\infty \leq \epsilon, \quad \text{for any} \quad \epsilon > 0.$$

Proof. We define the linear transformation of Z as a concatenation of two functions with some manual padding of zeros:

$$\mathrm{Linear}(Z) := \underbrace{\begin{bmatrix} \multicolumn{2}{c}{\mathrm{Linear}_x(X)} \\ \mathrm{Linear}_w(W) & 0_{(2d+n+2)\times nP} \end{bmatrix}}_{2(2d+n+2)\times n(P+1)},$$

where we define $\mathrm{Linear}_x \in \mathbb{R}^{(2d+n+2)\times n(P+1)}$ and $\mathrm{Linear}_w \in \mathbb{R}^{(2d+n+2)\times n}$ as below.

We define Linear_w as:

$$\mathrm{Linear}_w(W) := \underbrace{\begin{bmatrix} I_d \\ 0_{(d+n+2)\times d} \end{bmatrix}}_{(2d+n+2)\times d} \underbrace{W}_{d\times n} + \underbrace{\begin{bmatrix} 0_{d\times n} \\ -1_{1\times n} \\ 0_{d\times n} \\ 1_{1\times n} \\ I_n \end{bmatrix}}_{(2d+n+2)\times n} = \underbrace{\begin{bmatrix} W \\ -1_{1\times n} \\ 0_{d\times n} \\ 1_{1\times n} \\ I_n \end{bmatrix}}_{(2d+n+2)\times n}.$$

We define Linear_x as:

$$\begin{aligned}
&\mathrm{Linear}_x(X) \\
:= &\sum_{i=1}^{n} \underbrace{\begin{bmatrix} I_{d+1} \\ 0_{(d+1+n)\times(d+1)} \end{bmatrix}}_{(2d+n+2)\times(d+1)} \underbrace{X}_{(d+1)\times n} \underbrace{\begin{bmatrix} 0_{n\times(i-1)(P+1)} & 2L_0 e_i^{(n)} & 2L_1 e_i^{(n)} & \cdots & 2L_P e_i^{(n)} & 0_{n\times(n-i)(P+1)} \end{bmatrix}}_{n\times n(P+1)} \\
&+ \sum_{i=1}^{n} \underbrace{\begin{bmatrix} 0_{(d+1)\times d} & 0_{(d+1)} \\ I_d & 0_d \\ 0_{(n+1)\times d} & 0_{n+1} \end{bmatrix}}_{(2d+n+2)\times(d+1)} \underbrace{X}_{(d+1)\times n} \underbrace{\begin{bmatrix} 0_{n\times(i-1)(P+1)} & f(L_0)e_i^{(i)} & f(L_1)e_i^{(i)} & \cdots & f(L_P)e_i^{(i)} & 0_{n\times(n-i)(P+1)} \end{bmatrix}}_{n\times n(P+1)} \\
&+ \underbrace{\begin{bmatrix} 0_{(2d+1)\times(P+1)} & \cdots & 0_{(2d+1)\times(P+1)} \\ S & \cdots & S \\ Ce_1^{(n)} 1_{1\times(P+1)} & \cdots & Ce_n^{(n)} 1_{1\times(P+1)} \end{bmatrix}}_{(2d+n+2)\times n(P+1)} \\
= &\underbrace{\begin{bmatrix} T_1 & T_2 & \cdots & T_n \end{bmatrix}}_{(2d+n+2)\times n(P+1)},
\end{aligned}$$

where $\{L_j\}_{j=0}^{P}$ are the $P+1$ interpolation points (Definition D.3); $e_i^{(n)} \in \mathbb{R}^n$ is the one-hot vector with 1 at index i and 0 elsewhere; C is a constant to be determined later, and

$$1_{1\times(P+1)} := \underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}}_{1\times(P+1)},$$

$$S := \underbrace{\begin{bmatrix} -L_0^2 & -L_1^2 & \cdots & L_P^2 \end{bmatrix}}_{1\times(P+1)},$$

$$T_i := \underbrace{\begin{bmatrix} 2L_0 x_i & 2L_1 x_i & \cdots & 2L_P x_i \\ 2L_0 y_i & 2L_1 y_i & \cdots & 2L_P y_i \\ f(L_0) x_i & f(L_1) x_i & \cdots & f(L_P) x_i \\ -L_0^2 & -L_1^2 & \cdots & -L_P^2 \\ Ce_i^{(n)} & Ce_i^{(n)} & \cdots & Ce_i^{(n)} \end{bmatrix}}_{(2d+n+2)\times(P+1)}.$$

So our $\text{Linear}(Z)$ is:

$$\text{Linear}(Z) = \underbrace{\begin{bmatrix} T_1 \quad T_2 \quad \cdots \quad T_n \\ W_{d\times n} & 0_{d\times nP} \\ -1_{1\times n} & 0_{1\times nP} \\ 0_{d\times n} & 0_{d\times nP} \\ 1_{1\times n} & 0_{1\times nP} \\ I_n & 0_{n\times nP} \end{bmatrix}}_{2(2d+n+2)\times n(P+1)} = \underbrace{\begin{bmatrix} 2L_0x_1 & \cdots & 2L_Px_1 & \cdots & \cdots & 2L_0x_n & \cdots & 2L_Px_n \\ 2L_0y_1 & \cdots & 2L_Py_1 & \cdots & \cdots & 2L_0y_n & \cdots & 2L_Py_n \\ f(L_0)x_1 & \cdots & f(L_P)x_1 & \cdots & \cdots & f(L_0)x_n & \cdots & f(L_P)x_n \\ -L_0^2 & \cdots & -L_P^2 & \cdots & \cdots & -L_0^2 & \cdots & -L_P^2 \\ Ce_1^{(n)} & \cdots & Ce_1^{(n)} & \cdots & \cdots & Ce_n^{(n)} & \cdots & Ce_n^{(n)} \\ & W_{d\times n} & & & & & 0_{d\times nP} & \\ & -1_{1\times n} & & & & & 0_{1\times nP} & \\ & 0_{d\times n} & & & & & 0_{d\times nP} & \\ & 1_{1\times n} & & & & & 0_{1\times nP} & \\ & I_n & & & & & 0_{n\times nP} & \end{bmatrix}}_{2(2d+n+2)\times n(P+1)}.$$

Now we construct W_K, W_Q, W_V, W_O to be:

$$W_K := \underbrace{\begin{bmatrix} I_{2d+n+2} & 0_{(2d+n+2)\times(2d+n+2)} \end{bmatrix}}_{(2d+n+2)\times 2(2d+n+2)},$$

$$W_Q := \underbrace{\begin{bmatrix} 0_{(2d+n+2)\times(2d+n+2)} & I_{2d+n+2} \end{bmatrix}}_{(2d+n+2)\times 2(2d+n+2)},$$

$$W_V := \underbrace{\begin{bmatrix} 0_{d\times(d+1)} & I_d & 0_{d\times(2d+2n+3)} \end{bmatrix}}_{d\times 2(2d+n+2)},$$

$$W_O := \underbrace{\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix}}_{n(P+1)\times n}.$$

Thus,

$$W_K\text{Linear}(Z) = \underbrace{\begin{bmatrix} T_1 & T_2 & \cdots & T_n \end{bmatrix}}_{(2d+n+2)\times n(P+1)}, \qquad (W_K \text{ selects the } T_i \text{ blocks in } \text{Linear}(Z))$$

$$W_Q\text{Linear}(Z) = \underbrace{\begin{bmatrix} W_{d\times n} & 0_{d\times nP} \\ -1_{1\times n} & 0_{1\times nP} \\ 0_{d\times n} & 0_{d\times nP} \\ 1_{1\times n} & 0_{1\times nP} \\ I_n & 0_{n\times nP} \end{bmatrix}}_{(2d+n+2)\times n(P+1)}, \qquad (W_Q \text{ selects the bottom } (2d+n+2) \text{ rows in } \text{Linear}(Z))$$

$$W_V\text{Linear}(Z) = \underbrace{\begin{bmatrix} F_1 & F_2 & \cdots & F_n \end{bmatrix}}_{d\times n(P+1)}, \qquad (W_V \text{ selects the } (d+2)\text{-th row in } T_i)$$

where we define F_i as:

$$F_i := \underbrace{\begin{bmatrix} f(L_0)x_i & f(L_1)x_i & \cdots & f(L_P)x_i \end{bmatrix}}_{d\times(P+1)}.$$

Therefore,

$$\begin{aligned} &\text{Attn}_s \circ \text{Linear}(Z) \\ &= W_V\text{Linear}(Z)\cdot \text{Softmax}_\beta((W_K\text{Linear}(Z))^\top W_Q\text{Linear}(Z))\cdot W_O \\ &= \underbrace{\begin{bmatrix} F_1 & F_2 & \cdots & F_n \end{bmatrix}}_{d\times n(P+1)} \text{Softmax}_\beta\Big(\overbrace{\underbrace{\begin{bmatrix} T_1 & T_2 & \cdots & T_n \end{bmatrix}^\top}_{n(P+1)\times(2d+n+2)} \underbrace{\begin{bmatrix} W_{d\times n} & 0_{d\times nP} \\ -1_{1\times n} & 0_{1\times nP} \\ 0_{d\times n} & 0_{d\times nP} \\ 1_{1\times n} & 0_{1\times nP} \\ I_n & 0_{n\times nP} \end{bmatrix}}_{(2d+n+2)\times n(P+1)}}^{n(P+1)\times n(P+1)}\Big) \underbrace{\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix}}_{n(P+1)\times n}. \end{aligned} \tag{D.15}$$

For simplicity of presentation, we define

$$\widetilde{T} := \underbrace{[T_1 \quad T_2 \quad \cdots \quad T_n]^\top}_{n(P+1)\times(2d+n+2)}.$$

For the $\mathrm{Softmax}_\beta$ part in (D.15), we have:

$$\begin{aligned}
&\mathrm{Softmax}_\beta((W_K \mathrm{Linear}(Z))^\top W_Q \mathrm{Linear}(Z)) \cdot W_O \\
&= \mathrm{Softmax}_\beta(\underbrace{\widetilde{T}}_{n(P+1)\times(2d+n+2)} \underbrace{\begin{bmatrix} W_{d\times n} & 0_{d\times nP} \\ -1_{1\times n} & 0_{1\times nP} \\ 0_{d\times n} & 0_{d\times nP} \\ 1_{1\times n} & 0_{1\times nP} \\ I_n & 0_{n\times nP} \end{bmatrix}}_{(2d+n+2)\times n(P+1)}) \underbrace{\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix}}_{n(P+1)\times n} && \text{(By the definition of } \widetilde{T}) \\
&= \mathrm{Softmax}_\beta(\begin{bmatrix} \underbrace{\widetilde{T} \begin{bmatrix} W_{d\times n} \\ -1_{1\times n} \\ 0_{d\times n} \\ 1_{1\times n} \\ I_n \end{bmatrix}}_{n(P+1)\times n} & \underbrace{\widetilde{T} \begin{bmatrix} 0_{d\times nP} \\ 0_{1\times nP} \\ 0_{d\times nP} \\ 0_{1\times nP} \\ 0_{n\times nP} \end{bmatrix}}_{n(P+1)\times nP} \end{bmatrix}) \underbrace{\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix}}_{n(P+1)\times n} \\
& && \text{(By distributivity of matrix multiplication over block concatenation)} \\
&= \begin{bmatrix} \underbrace{\mathrm{Softmax}_\beta(\widetilde{T} \begin{bmatrix} W_{d\times n} \\ -1_{1\times n} \\ 0_{d\times n} \\ 1_{1\times n} \\ I_n \end{bmatrix})}_{n(P+1)\times n} & \underbrace{\mathrm{Softmax}_\beta(\widetilde{T} \begin{bmatrix} 0_{d\times nP} \\ 0_{1\times nP} \\ 0_{d\times nP} \\ 0_{1\times nP} \\ 0_{n\times nP} \end{bmatrix})}_{n(P+1)\times nP} \end{bmatrix} \underbrace{\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix}}_{n(P+1)\times n} \\
& && \text{(By the column-wise operation nature of } \mathrm{Softmax}_\beta) \\
&= \mathrm{Softmax}_\beta(\underbrace{[T_1 \quad T_2 \quad \cdots \quad T_n]^\top}_{n(P+1)\times(2d+n+2)} \underbrace{\begin{bmatrix} W_{d\times n} \\ -1_{1\times n} \\ 0_{d\times n} \\ 1_{1\times n} \\ I_n \end{bmatrix}}_{(2d+n+2)\times n}). && (\begin{bmatrix} I_n \\ 0_{nP\times n} \end{bmatrix} \text{ selects the first } \mathrm{Softmax}_\beta \text{ block})
\end{aligned}$$

Since our target is a token-wise approximation, we focus on a single token. We consider the c-th column ($c \in [n]$) in the $\mathrm{Softmax}_\beta$ part, and we have

$$\begin{aligned}
(\mathrm{Softmax}_\beta((W_K \mathrm{Linear}(Z))^\top W_Q \mathrm{Linear}(Z)) \cdot W_O)_{:,c} &= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} T_1^\top \\ T_2^\top \\ \vdots \\ T_n^\top \end{bmatrix}}_{n(P+1)\times(2d+n+2)} \cdot \underbrace{\begin{bmatrix} w \\ -1 \\ 0_d \\ 1 \\ e_c^{(n)} \end{bmatrix}}_{(2d+n+2)\times 1}) \\
&= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} M_{1,c} \\ M_{2,c} \\ \vdots \\ M_{n,c} \end{bmatrix}}_{n(P+1)\times 1}),
\end{aligned}$$

where each sub-block $M_{i,c} \in \mathbb{R}^{(P+1)\times 1}$ for $i \in [n]$ is

$$M_{i,c} := \underbrace{T_i^\top}_{(P+1)\times(2d+n+2)} \cdot \underbrace{\begin{bmatrix} w \\ -1 \\ 0_d \\ 1 \\ e_c^{(n)} \end{bmatrix}}_{(2d+n+2)\times 1}$$

$$= \underbrace{\begin{bmatrix} 2L_0 x_i^\top & 2L_0 y_i & f(L_0) x_i^\top & -L_0^2 & C(e_i^{(n)})^\top \\ 2L_1 x_i^\top & 2L_1 y_i & f(L_1) x_i^\top & -L_1^2 & C(e_i^{(n)})^\top \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 2L_P x_i^\top & 2L_P y_i & f(L_P) x_i^\top & -L_P^2 & C(e_i^{(n)})^\top \end{bmatrix}}_{(P+1)\times(2d+n+2)} \cdot \underbrace{\begin{bmatrix} w \\ -1 \\ 0_d \\ 1 \\ e_c^{(n)} \end{bmatrix}}_{(2d+n+2)\times 1} \quad \text{(By transpose of } T_i\text{)}$$

$$= \underbrace{\begin{bmatrix} 2L_0 x_i^\top w - 2L_0 y_i - L_0^2 + C\mathbb{1}_{i=c} \\ 2L_1 x_i^\top w - 2L_1 y_i - L_1^2 + C\mathbb{1}_{i=c} \\ \vdots \\ 2L_P x_i^\top w - 2L_P y_i - L_P^2 + C\mathbb{1}_{i=c} \end{bmatrix}}_{(P+1)\times 1},$$

where $\mathbb{1}_{i=c}$ denotes the indicator function of $i = c$.

For simplicity, let

$$u_j^{(i)} := 2L_j x_i^\top w - 2L_j y_i - L_j^2, \quad \text{for} \quad j \in \{0, \ldots, P\}, \tag{D.16}$$

such that

$$M_{i,c} = \underbrace{\begin{bmatrix} u_0^{(i)} + C\mathbb{1}_{i=c} \\ u_1^{(i)} + C\mathbb{1}_{i=c} \\ \vdots \\ u_P^{(i)} + C\mathbb{1}_{i=c} \end{bmatrix}}_{(P+1)\times 1}.$$

This means that

$$\begin{aligned} &(\mathrm{Softmax}_\beta((W_K \mathrm{Linear}(Z))^\top W_Q \mathrm{Linear}(Z)) \cdot W_O)_{:,c} \\ &= \mathrm{Softmax}(\beta \underbrace{\begin{bmatrix} M_{1,c} \\ M_{2,c} \\ \vdots \\ M_{n,c} \end{bmatrix}}_{n(P+1)\times 1}) \\ &= \sum_{i=1}^n \sum_{j=0}^P \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} \underbrace{e_{(i-1)(P+1)+(j+1)}^{(n(P+1))}}_{n(P+1)\times 1}. \end{aligned}$$

(By the definition of $\mathrm{Softmax}_\beta$)

Thus we have

$$\begin{aligned} &\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} \\ &= W_V \mathrm{Linear}(Z) \cdot (\mathrm{Softmax}_\beta((W_K \mathrm{Linear}(Z))^\top W_Q \mathrm{Linear}(Z)) \cdot W_O)_{:,c} \\ &= \underbrace{\begin{bmatrix} F_1 & \cdots & F_n \end{bmatrix}}_{d\times n(P+1)} \cdot \sum_{i=1}^n \sum_{j=0}^P \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} \underbrace{e_{(i-1)(P+1)+(j+1)}^{(n(P+1))}}_{n(P+1)\times 1} \end{aligned}$$

$$= \underbrace{\begin{bmatrix} f(L_0)x_1 & \cdots & f(L_P)x_1 & \cdots & f(L_0)x_n & \cdots & f(L_P)x_n \end{bmatrix}}_{d\times n(P+1)} \cdot \qquad \text{(By the definitoin of } F_i\text{)}$$

$$\sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} \underbrace{e_{(i-1)(P+1)+(j+1)}^{(n(P+1))}}_{n(P+1)\times 1}$$

$$= \sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}}$$

$$\underbrace{\begin{bmatrix} f(L_0)x_1 & \cdots & f(L_P)x_1 & \cdots & f(L_0)x_n & \cdots & f(L_P)x_n \end{bmatrix}}_{d\times n(P+1)} \cdot \underbrace{e_{(i-1)(P+1)+(j+1)}^{(n(P+1))}}_{n(P+1)\times 1}$$

(By distributivity of matrix multiplication)

$$= \sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} f(L_j) \underbrace{x_i}_{d\times 1}.$$

(The one-hot vector retrieves the $((i-1)(P+1)+(j+1))$-th column)

Again, our goal is to approximate $f(x_c^\top w - y_c)x_c$ with:

$$\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} = \sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\left\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\right\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} f(L_j)x_i. \tag{D.17}$$

We start to analyze the summation of weights $\sum_{j=0}^{P}(\cdots)$ for $i = c$ and $i \neq c$. We use the result of this analysis to bound our target approximation $\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} - f(x_c^\top w - y_c)x_c\|_\infty$ later.

- For every $i \in [n]$, if $i \neq c$, we have

$$\frac{\sum_{j=0}^{P}\exp\left\{\beta u_j^{(i)}\right\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\right\}} \qquad \text{(By } \mathbb{1}_{i\neq c} = 0 \text{ for } i \neq c\text{)}$$

$$< \frac{\sum_{j=0}^{P}\exp\left\{\beta u_j^{(i)}\right\}}{\sum_{j'=0}^{P}\exp\left\{\beta(u_{j'}^{(i')} + C)\right\}} \tag{D.18}$$

$$\leq \epsilon_0, \tag{D.19}$$

where (D.18) is by taking only the $i' = c$ term, and the last line is by the softmax-shift equality

$$\frac{\sum_{j=0}^{P} e^{u_j}}{\sum_{j'=0}^{P} e^{v_{j'}+C}} = \frac{\sum_{j=0}^{P} e^{u_j}}{e^C \sum_{j'=0}^{P} e^{v_{j'}}},$$

for any constant C and choosing $C := M - \frac{1}{\beta}\ln\epsilon_0 = (\max_j L_j)\cdot(2dB^2 + 2B) - \frac{1}{\beta}\ln\epsilon_0$ with $\epsilon_0 > 0$.[3]

[3]More explicitly, recall (D.16): $u_j^{(i)} := 2L_j x_i^\top w - 2L_j y_i - L_j^2$. Since $\max\{\|X\|_\infty, \|W\|_\infty\} \leq B$, we have

$$\|x_i\|_\infty \leq B, \quad |y_i| \leq B, \quad \|w\|_\infty \leq B,$$

which implies $\|w\|_1 \leq dB$. Let $L_\star := \max_j |L_j|$. For a fixed pair of i, i', we have

$$\begin{aligned}
u_j^{(i)} - u_j^{(i')} &= 2L_j \cdot ((x_i - x_{i'})^\top w - (y_i - y_{i'})) \\
&\leq 2|L_j| \cdot (|(x_i - x_{i'})^\top w| + |(y_i - y_{i'})|) && \text{(By triangle inequality)} \\
&\leq 2L_\star \cdot (\|x_i - x_{i'}\|_\infty \cdot \|w\|_1 + |(y_i - y_{i'})|) && \text{(By } L_\star := \max_j |L_j| \text{ and Hölder's inequality)} \\
&\leq 2L_\star \cdot ((\|x_i\|_\infty + \|x_{i'}\|_\infty) \cdot \|w\|_1 + |y_i| + |y_{i'}|)) && \text{(By triangle inequality)} \\
&\leq 2L_\star \cdot ((2B) \cdot dB + 2B) && \text{(By } \|x_i\|_\infty \leq B, |y| \leq B \text{ and } \|w\|_1 \leq dB\text{)}
\end{aligned}$$

Thus, the weight assigned to $i \neq c$ is tiny.

- For $i = c$, we have

$$
\begin{aligned}
&\frac{\sum_{j=0}^{P} \exp\Big\{\beta(u_j^{(c)} + C\Big\})}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \\
&= 1 - \frac{\sum_{i\neq c}^{n}\sum_{j=0}^{P} \exp\Big\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \quad (\text{By } \textstyle\sum_{i\neq c}^{n}\sum_{j=0}^{P}(\cdots) + \sum_{i=c}^{n}\sum_{j=0}^{P}(\cdots) = 1) \\
&\geq 1 - (n-1)\epsilon_0, \qquad \text{(D.21)}
\end{aligned}
$$

where the last inequality follows from (i.e., (D.19))

$$
\frac{\sum_{i\neq c}^{n}\sum_{j=0}^{P} \exp\Big\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \leq (n-1)\epsilon_0,
$$

and setting $0 < \epsilon_0 < 1/(n-1)$. Therefore, the weight concentrates at $i = c$.

From (D.19), (D.21), and our target approximation

$$
\begin{aligned}
&\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} - f(x_c^\top w - y_c)x_c\|_\infty \\
&= \|\sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i - f(x_c^\top w - y_c)x_c\|_\infty, \qquad \text{(D.22)}
\end{aligned}
$$

we split (D.22) into two terms

$$
\begin{aligned}
&\|\sum_{i=1}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)} + C\mathbb{1}_{i=c})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i - f(x_c^\top w - y_c)x_c\|_\infty \\
&= \|\sum_{i\neq c}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i \\
&\quad + \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i - f(x_c^\top w - y_c)x_c\|_\infty \\
&\qquad\qquad (\text{By splitting the summation over } i \text{ into two parts: } i = c \text{ and } i \neq c) \\
&\leq \underbrace{\|\sum_{i\neq c}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i\|_\infty}_{(I)} \qquad \text{(D.23)}
\end{aligned}
$$

$$
\leq 2L_\star \cdot (2dB^2 + 2B) =: M.
$$

Hence, we have $u_j^{(i)} \leq u_j^{(i')} + M$, which implies

$$
e^{\beta u_j^{(i)}} \leq e^{\beta M} e^{\beta u_j^{(i')}}, \quad \text{for all} \quad j \in \{0, \ldots, P\}. \qquad \text{(D.20)}
$$

Then, (D.18) becomes, for any constant C,

$$
\frac{\sum_{j=0}^{P} \exp\Big\{\beta u_j^{(i)}\Big\}}{\exp\{\beta C\}\sum_{j'=0}^{P} \exp\Big\{\beta u_{j'}^{(i')}\Big\}} \leq \frac{e^{\beta M}\sum_{j=0}^{P} \exp\Big\{\beta u_j^{(i')}\Big\}}{e^{\beta C}\sum_{j'=0}^{P} \exp\Big\{\beta u_{j'}^{(i')}\Big\}} = e^{\beta(M-C)}.
$$

Choosing $C := M - \frac{1}{\beta}\ln\epsilon_0 = (\max_j L_j)\cdot(2dB^2 + 2B) - \frac{1}{\beta}\ln\epsilon_0$, we obtain the desired bound ϵ_0.

$$+ \underbrace{\| \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i - f(x_c^\top w - y_c)x_c\|_\infty}_{(II)},$$

(By triangle inequality)

where we are capable of bounding term (I) with (D.19) and term (II) as follows.

For term (I) in (D.23), we have

$$\begin{aligned}
&(I)\\
&= \| \sum_{i\neq c}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i\|_\infty\\
&\leq \sum_{i\neq c}^{n}\sum_{j=0}^{P} \| \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j)x_i\|_\infty && \text{(By triangle inequality)}\\
&= \sum_{i\neq c}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \|f(L_j)x_i\|_\infty && \text{(By non-negativity of exponential)}\\
&= \sum_{i\neq c}^{n}\sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} |f(L_j)| \cdot \|x_i\|_\infty\\
& && \text{(By } \|f(L_j)x_i\|_\infty = |f(L_j)|\cdot\|x_i\|_\infty\text{)}\\
&\leq \frac{\sum_{i\neq c}^{n}\sum_{j=0}^{P} \exp\Big\{\beta(u_j^{(i)})\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} B_f \cdot B\\
& && \text{(By } B_f \coloneqq \max|f| \text{ and } \max\{\|X\|_\infty, \|W\|_\infty\} \leq B\text{)}\\
&\leq (n-1)\epsilon_0 B_f B, && \text{(By (D.19))}
\end{aligned}$$

where we define $B_f \coloneqq \max |f|$ as the bound for f.

For term (II) in (D.23), we have

$$\begin{aligned}
&(II)\\
&= \| \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} f(L_j) \underbrace{x_c}_{d\times 1} - f(x_c^\top w - y_c)\underbrace{x_c}_{d\times 1}\|_\infty\\
&= \| \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_{k}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \cdot (f(L_j)x_c - f(x_c^\top w - y_c)x_c)\\
&\quad - (1 - \frac{\sum_{j'=0}^{P}\exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_{k}^{(i')} + C\mathbb{1}_{i'=c})\Big\}}) \cdot f(x_c^\top w - y_c)x_c\|_\infty\\
& && \text{(By } \textstyle\sum_j \frac{A_j}{C}B_j - D = \sum_j \frac{A_j}{C}(B_j - D) - (1 - \frac{\sum_{j'} A_{j'}}{C})D\text{)}\\
&= \| \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}} \cdot \frac{\sum_{j'=0}^{P}\exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_{k}^{(i')} + C\mathbb{1}_{i'=c})\Big\}} \cdot (f(L_j)x_c - f(x_c^\top w - y_c)x_c)
\end{aligned}$$

$$- (1 - \frac{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_k^{(i')} + C\mathbb{1}_{i'=c})\Big\}}) \cdot f(x_c^\top w - y_c) x_c\|_\infty$$

$$\Big(\text{By } \sum_j \tfrac{A_j}{C}(B_j - D) - (1 - \tfrac{\sum_{j'} A_{j'}}{C})D = \sum_j \tfrac{A_j}{E} \cdot \tfrac{E}{C}(B_j - D) - (1 - \tfrac{\sum_{j'} A_{j'}}{C})D\Big)$$

$$\leq \| \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}} \cdot (f(L_j) - f(x_c^\top w - y_c)) x_c$$

$$- (1 - \frac{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_k^{(i')} + C\mathbb{1}_{i'=c})\Big\}}) \cdot f(x_c^\top w - y_c) x_c\|_\infty \qquad \Big(\text{By } \tfrac{E}{C} < 1\Big)$$

$$\leq \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty$$

$$- (1 - \frac{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}}{\sum_{i'=1}^{n}\sum_{k=0}^{P} \exp\Big\{\beta(u_k^{(i')} + C\mathbb{1}_{i'=c})\Big\}}) \cdot |f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty$$

$$\Big(\text{By triangle inequality, and } \|av\|_\infty \leq |a| \cdot \|v\|_\infty \text{ where } a \in \mathbb{R} \text{ and } v \in \mathbb{R}^d\Big)$$

$$\leq \sum_{j=0}^{P} \frac{\exp\Big\{\beta(u_j^{(c)} + C)\Big\}}{\sum_{j'=0}^{P} \exp\Big\{\beta(u_{j'}^{(c)} + C)\Big\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty + (n-1)\epsilon_0 B_f \|x_c\|_\infty$$

$$\Big(\text{By (D.21) and } B_f \coloneqq \max |f|\Big)$$

$$= \sum_{j=0}^{P} \frac{\exp\Big\{\beta u_j^{(c)}\Big\}}{\sum_{j'=0}^{P} \exp\Big\{\beta u_{j'}^{(c)}\Big\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty + (n-1)\epsilon_0 B_f B$$

$$\Big(\text{By } \exp\Big\{\beta(u_j^{(c)} + C)\Big\} = \exp\Big\{\beta u_j^{(c)}\Big\} \exp\{\beta C\} \text{ and } \max\{\|X\|_\infty, \|W\|_\infty\} \leq B\Big)$$

$$= \underbrace{\sum_{j=0}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty}_{\coloneqq (II\text{-}1)} + \underbrace{(n-1)\epsilon_0 B_f B}_{\coloneqq (II\text{-}2)}, \tag{D.24}$$

where the last equality follows from completing the square

$$u_j^{(c)} = 2L_j x_c^\top w - 2L_j y_c - L_j^2 = -(L_j - (x_c^\top w - y_c))^2 + (x_c^\top w - y_c)^2.$$

For term $(II\text{-}1)$ in (D.24), we have

$$(II\text{-}1)$$

$$= \sum_{j=0}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty$$

$$= \sum_{j:|L_j - (x_c^\top w - y_c)| \leq \Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty$$

$$+ \sum_{j:|L_j - (x_c^\top w - y_c)| > \Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty, \tag{D.25}$$

where we define $\Delta L \coloneqq (L_{\max} - L_{\min})/P$ and divide the interpolation points into two groups with one group at least ΔL away from $x_c^\top w - y_c$, and the other within ΔL.

For the first term in (D.25), we set ΔL to be sufficiently small (P large enough) such that,

$$|f(t) - f(t')| \leq \epsilon_1, \quad \forall\, \epsilon_1 > 0,$$

when $|t - t'| \leq \Delta L$.

For the second term in (D.25), we set β to be sufficiently large such that

$$\sum_{j:|L_j-(x_c^\top w-y_c)|>\Delta L} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \leq \epsilon_2, \tag{D.26}$$

for any $0 < \epsilon_2 < 1$.

Thus, for term $(II\text{-}1)$, we have

$$\begin{aligned}
&(II\text{-}1)\\
=\ & \sum_{j:|L_j-(x_c^\top w-y_c)|\leq\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty\\
&+ \sum_{j:|L_j-(x_c^\top w-y_c)|>\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot |f(L_j) - f(x_c^\top w - y_c)| \cdot \|x_c\|_\infty\\
\leq\ & \sum_{j:|L_j-(x_c^\top w-y_c)|\leq\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot \epsilon_1 \cdot B\\
&+ \sum_{j:|L_j-(x_c^\top w-y_c)|>\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot (|f(L_j)| + |f(x_c^\top w - y_c)|) \cdot B\\
&\qquad (\text{By } |f(L_j) - f(x_c^\top w - y_c)| < \epsilon_1, \max\{\|X\|_\infty, \|W\|_\infty\} \leq B, \text{ and triangle inequality})\\
\leq\ & \sum_{j:|L_j-(x_c^\top w-y_c)|\leq\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot \epsilon_1 \cdot B\\
&+ \sum_{j:|L_j-(x_c^\top w-y_c)|>\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \cdot 2B_f \cdot B \qquad (\text{By } B_f := \max|f|)\\
\leq\ & \underbrace{\sum_{j:|L_j-(x_c^\top w-y_c)|\leq\Delta L}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P}\exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}}}_{:=\kappa} \cdot \epsilon_1 \cdot B + \epsilon_2 \cdot 2B_f \cdot B \qquad (\text{By (D.26)})\\
\leq\ & \epsilon_1 \cdot B + \epsilon_2 \cdot 2B_f \cdot B. \qquad (\text{By } \kappa < 1)
\end{aligned}$$

Combining (I), $(II\text{-}1)$, and $(II\text{-}2)$, we have:

$$\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} - f(x_c^\top w - y_c)x_c\|_\infty \leq \underbrace{(n-1)\epsilon_0 B_f B}_{\text{from } (I)} + \underbrace{\epsilon_1 B + 2\epsilon_2 B_f B}_{\text{from } (II\text{-}1)} + \underbrace{(n-1)\epsilon_0 B_f B}_{\text{from } (II\text{-}2)}.$$

Since ϵ_0, ϵ_1 and ϵ_2 are arbitrarily small, we have

$$\|\mathrm{Attn}_s \circ \mathrm{Linear}(Z)_{:,c} - f(x_c^\top w - y_c)x_c\|_\infty \leq \epsilon,$$

for any $\epsilon > 0$.

This completes the proof. □

D.2 Proof of Corollary 3.1.2

Corollary D.2.1 (In-Context Emulation of a Single GD Step; Corollary 3.1.2 Restate). Let $\ell : \mathbb{R} \to \mathbb{R}$ be differentiable and define $\widehat{L}_n(w) := \frac{1}{n}\sum_{i=1}^{n} \ell(w^\top x_i - y_i)$. For any step size $\eta > 0$ and any $\epsilon > 0$, there exist a single-head attention Attn_s and a linear map Linear such that, with $Z = [X; W]$ as in (3.1), choosing the readout $u := \frac{1}{n}\mathbf{1}_n$ (equivalently, right-multiply by $W_O = u$ in Definition 2.1), we have

$$\widehat{w}_{\text{GD}} := (\text{Attn}_s \circ \text{Linear}(Z))u \in \mathbb{R}^d \quad \text{and} \quad \|\widehat{w}_{\text{GD}} - \underbrace{(w - \eta \nabla \widehat{L}_n(w))}_{w_{\text{GD}}^+}\|_\infty \leq \epsilon.$$

Proof. From Corollary 3.1.1, we derive that $\|(\text{Attn}_s \circ \text{Linear})_i - \nabla \ell(w^\top x_i - y_i)x_i\|_\infty \leq \epsilon$ for all $i \in [n]$.

Therefore,

$$\begin{aligned}
\widehat{w} &= w + \frac{1}{n}\sum_{j=1}^{n}(\text{Attn}_s \circ \text{Linear})_j \\
&= w - \frac{\eta}{n}\sum_{j=1}^{n}\nabla \ell(w^\top x_j - y_j)x_j + \epsilon' \\
&= w - \eta \nabla \widehat{L_n}(w) + \epsilon' \\
&= w_{\text{GD}} + \epsilon'
\end{aligned}$$

This completes the proof. ☐

D.3 Proof of Corollary 3.1.3

Theorem D.3 (In-Context Emulation of Linear Regression; Corollary 3.1.3 Restate)**.** For any dataset $\{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^d$, $y_i \in \mathbb{R}$ and any $\epsilon > 0$, there exist a single-head attention Attn_s, a linear map Linear, and a readout $u \in \mathbb{R}^n$ such that, with $Z = [X; W]$ as in (3.1) (for any fixed bounded w),

$$\widehat{w}_{\text{linear}} := (\text{Attn}_s \circ \text{Linear}(Z))u \in \mathbb{R}^d, \quad \text{and} \quad \|\widehat{w}_{\text{linear}} - w_{\text{linear}}\|_\infty \leq \epsilon,$$

where $w_{\text{linear}} := \text{argmin}_{w \in \mathbb{R}^d} \frac{1}{2n}\sum_{i=1}^n(\langle w, x_i\rangle - y_i)^2$.

Proof. From Corollary 3.1.2, we know that $\|\widehat{w}^l - w_{\text{GD}}^l\|_\infty \leq \epsilon/2$ for all $l \in [L]$. Note that $\frac{1}{2n}\sum_{i=1}^n(\langle w, x_i\rangle - y_i)^2$ is convex and smooth which satisfies the precondition for Corollary D.1.1. Therefore, from Corollary D.1.1, using $\|\cdot\|_\infty \leq \|\cdot\|_2$, we derive that $\|w_{\text{GD}}^l - w_{\text{linear}}^l\|_\infty \leq \epsilon/2$. Thus, $\|\text{Attn} - w_{\text{linear}}\|_\infty \leq \|\widehat{w}^l - w_{\text{GD}}^l\|_\infty + \|w_{\text{GD}}^l - w_{\text{linear}}^l\|_\infty \leq \epsilon$ by triangle inequality. This completes the proof. ☐

D.4 Proof of Corollary 3.1.4

Theorem D.4 (Restate of Corollary 3.1.4: In-Context Emulation of Ridge Regression)**.** For any input-output pair (x_i, y_i), where $x_i \in \mathbb{R}^d, y_i \in \mathbb{R}, i \in [n]$, and any $\epsilon > 0$, there exists a single-layer Attention network with linear connection Attn such that

$$\|\text{Attn} - w_{\text{ridge}}\|_\infty \leq \epsilon,$$

where $w_{\text{ridge}} := \text{argmin}_{w \in \mathbb{R}^d} \frac{1}{2n}\sum_{i=1}^n(\langle w, x_i\rangle - y_i)^2 + \frac{\lambda}{2}\|w\|_2^2$ with regularization term $\lambda \geq 0$.

Proof. From Corollary 3.1.2, we know that $\|\widehat{w}^l - w_{\text{GD}}^l\|_\infty \leq \epsilon/2$ for all $l \in [L]$. Note that $\frac{1}{2n}\sum_{i=1}^n(\langle w, x_i\rangle - y_i)^2 + \frac{\lambda}{2}\|w\|_2^2$ is convex and smooth which satisfies the precondition for Corollary D.1.1. Therefore, from Corollary D.1.1, using $\|\cdot\|_\infty \leq \|\cdot\|_2$, we derive that $\|w_{\text{GD}}^l - w_{\text{ridge}}^l\|_\infty \leq \epsilon/2$. Thus, $\|\text{Attn} - w_{\text{ridge}}\|_\infty \leq \|\widehat{w}^l - w_{\text{GD}}^l\|_\infty + \|w_{\text{GD}}^l - w_{\text{ridge}}^l\|_\infty \leq \epsilon$ by triangle inequality. This completes the proof. ☐

D.5 Proof of Theorem 4.1

Theorem D.5 (In-Context Emulation of Attention; Theorem 4.1 Restate). Let $X \in \mathbb{R}^{d \times n}$ be an input sequence, and let $W_K, W_Q, W_V \in \mathbb{R}^{d_h \times d}$ be the weight matrices of the target attention head we wish to emulate in-context. Assume $\|W_K X\|_\infty, \|W_Q X\|_\infty, \|W_V X\|_\infty \le B_{KQV}$ with $B_{KQV} > 0$. Then, for any $\epsilon > 0$, there exists a two-layer attention network — a multi-head attention layer Attn_m followed by a single-head attention layer Attn_s — such that

$$\| \underbrace{\mathrm{Attn}_s \circ \mathrm{Attn}_m(X_p)}_{\text{Emulator}} - \underbrace{W_V X \mathrm{Softmax}_\beta((W_K X)^\top W_Q X)}_{\text{Target}} \|_\infty \le \epsilon,$$

where X_p is the prompt defined in Definition 4.2.

Proof. We state our high-level proof sketch first.

Step 1: In-Context Weight Encoding. We define

$$\underbrace{K}_{d_h \times n} := \underbrace{W_K}_{d_h \times d} \cdot \underbrace{X}_{d \times n}, \quad \underbrace{Q}_{d_h \times n} := \underbrace{W_Q}_{d_h \times d} \cdot \underbrace{X}_{d \times n}, \quad \underbrace{V}_{d_h \times n} := \underbrace{W_V}_{d_h \times d} \cdot \underbrace{X}_{d \times n}.$$

We aim to approximate the attention mechanism $V \mathrm{Softmax}_\beta(K^\top Q)$ using a two-layer transformer $\mathrm{Attn}_s \circ \mathrm{Attn}_m$. Therefore, the transformer $\mathrm{Attn}_s \circ \mathrm{Attn}_m$ must have in-context access to the information about W_K, W_Q and W_V. This is equivalent to exposing the transformer $\mathrm{Attn}_s \circ \mathrm{Attn}_m$ to the target algorithm's specification.

To that end, we augment the input sequence X with two auxiliary blocks:

1. The weight encoding W_{in} of the target algorithm. W_{in} contains the vectorization of the target weights W_K, W_Q and W_V.
2. A positional encoding I_n. I_n exposes token indices.

Concretely, following Definition 4.2, we form

$$X_p = \underbrace{\begin{bmatrix} X \\ W_{\text{in}} \\ I_n \end{bmatrix}}_{(d+6dd_h+n)\times n} \quad \text{with} \quad W_{\text{in}} = \underbrace{\begin{bmatrix} 0 \cdot w & 1 \cdot w & 2 \cdot w & \cdots & (n-1)\cdot w \\ w & w & w & \cdots & w \end{bmatrix}}_{6dd_h \times n},$$

and

$$w = \underbrace{\begin{bmatrix} \underline{W_K} \\ \underline{W_Q} \\ \underline{W_V} \end{bmatrix}}_{3dd_h \times 1}.$$

Step 2: Multi-Head Decomposition for In-Context Recovery of K, Q, V. In this step, we use the multi-head layer Attn_m to build approximators of K, Q, and V from the prompt X_p. We denote these approximators by K', Q', and V', corresponding to K, Q, and V.

Explicitly, we have

$$\| \overbrace{\underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3d_h \times n}}^{\mathrm{Attn}_m(X_p)} - \underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3d_h \times n} \|_\infty \le \epsilon_0.$$

Intuitively, this works as: X_p contains the raw input X and the weight encodings of W_K, W_Q and W_V. Then Attn_m "reads" X and the target weight parameters from X_p within its attention heads to form the desired approximation.

Step 3: Single-Head Assembly for Emulated Map. We use the single-head layer Attn_s to perform the attention computation. From K', Q', V', Attn_s produces

$$V'\mathrm{Softmax}_\beta((K')^\top Q').$$

For reference, the target computation is $V\mathrm{Softmax}_\beta((K)^\top Q)$. This step aligns the output of Attn_s with the target attention, using the approximated K', Q', V' as inputs.

Step 4: Error Bound. Finally, we bound the gap between the computed and target attention:

$$\|V'\mathrm{Softmax}_\beta((K')^\top Q') - V\mathrm{Softmax}_\beta((K)^\top Q)\|_\infty \le \epsilon_0 + nB_{KQV}\epsilon_1.$$

Our proof starts here.

Step 1: In-Context Weight Encoding. For clarity and simplicity, we define

$$k_i := (W_K^\top)_{:,i} \in \mathbb{R}^d, \tag{D.27}$$

$$q_i := (W_Q^\top)_{:,i} \in \mathbb{R}^d, \tag{D.28}$$

$$v_i := (W_V^\top)_{:,i} \in \mathbb{R}^d, \tag{D.29}$$

such that the vectorized weight matrices $\underline{W}_K, \underline{W}_Q, \underline{W}_V$ in Definition 4.2 become

$$\underline{W}_K = \begin{bmatrix} k_1 \\ k_2 \\ \vdots \\ k_{d_h} \end{bmatrix} \in \mathbb{R}^{dd_h},\ \underline{W}_Q = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{d_h} \end{bmatrix} \in \mathbb{R}^{dd_h},\ \underline{W}_V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{d_h} \end{bmatrix} \in \mathbb{R}^{dd_h},$$

and w becomes

$$w = \begin{bmatrix} \underline{W}_K \\ \underline{W}_Q \\ \underline{W}_V \end{bmatrix} = \begin{bmatrix} k_1 \\ \vdots \\ k_{d_h} \\ q_1 \\ \vdots \\ q_{d_h} \\ v_1 \\ \vdots \\ v_{d_h} \end{bmatrix}.$$

W_{in} remains as

$$W_{\text{in}} := \begin{bmatrix} 0 \cdot w & 1 \cdot w & 2 \cdot w & \cdots & (n-1) \cdot w \\ w & w & w & \cdots & w \end{bmatrix} \in \mathbb{R}^{6dd_h \times n}.$$

Then, for the input X, we append it with the target weights W_{in} and the positional encoding I_n as in Definition 4.2. We denote this result with X_p and write it out as

$$X_p := \underbrace{\begin{bmatrix} X \\ W_{\text{in}} \\ I_n \end{bmatrix}}_{(d+6dd_h+n)\times n}. \tag{D.30}$$

Step 2: Multi-Head Decomposition for In-Context Recovery of K, Q, V. In this part, we construct approximators for K, Q and V via Attn_m. We construct the approximators by approximating each row of K, Q, V and then aggregating the results across rows. Each row in K, Q and V has the form: $k_i^\top X, q_i^\top X$, and $v_i^\top X$. To approximate these rows in K, Q, V, we apply Theorem D.1 to each row separately. Namely, we allocate an H-head attention to each row of K, Q and V to carry out the row-wise approximations. Since $K, Q, V \in \mathbb{R}^{d_h \times n}$, each of K, Q, V uses $H \cdot d_h$ heads. We interpret H as the number of heads per row dimension, since each K, Q, and V has d_h rows. Finally, we define a multi-head attention Attn_m as the union of these three groups of Hd_h heads. Therefore, Attn_m has $3Hd_h$ heads in total.

We label the $3Hd_h$ heads in Attn_m as:

$$\mathrm{Attn}^K_{j,\widetilde{h}}, \quad j \in [d_h], \quad \widetilde{h} \in \{J+1, \ldots, J+H\}; \qquad \text{(Approximates } K\text{)}$$

$$\mathrm{Attn}^Q_{j,\widetilde{h}}, \quad j \in [2d_h] \setminus [d_h], \quad \widetilde{h} \in \{J+1, \ldots, J+H\}; \qquad \text{(Approximates } Q\text{)}$$

$$\mathrm{Attn}^V_{j,\widetilde{h}}, \quad j \in [3d_h] \setminus [2d_h], \quad \widetilde{h} \in \{J+1, \ldots, J+H\}, \qquad \text{(Approximates } V\text{)}$$

where we define $J := (j-1)H$ to simplify our notation. Each $\mathrm{Attn}^K_{j,\widetilde{h}}$, $\mathrm{Attn}^Q_{j,\widetilde{h}}$, and $\mathrm{Attn}^V_{j,\widetilde{h}}$ is a single-head attention. Index j identifies the target row, and index $\widetilde{h}$ identifies the head allocated to that row. Here $j \in [2d_h] \setminus [d_h]$ denotes the set difference. That is, $j \in [2d_h] \setminus [d_h]$ means $j \in \{d_h+1, \ldots, 2d_h\}$.

Thus, Attn_m consists of three groups of attention heads:

$$\mathrm{Attn}_m := \underbrace{\sum_{j=1}^{d_h} \sum_{\widetilde{h}=J+1}^{J+H} \mathrm{Attn}^K_{j,\widetilde{h}}}_{\text{Approximates } K} + \underbrace{\sum_{j=d_h+1}^{2d_h} \sum_{\widetilde{h}=J+1}^{J+H} \mathrm{Attn}^Q_{j,\widetilde{h}}}_{\text{Approximates } Q} + \underbrace{\sum_{j=2d_h+1}^{3d_h} \sum_{\widetilde{h}=J+1}^{J+H} \mathrm{Attn}^V_{j,\widetilde{h}}}_{\text{Approximates } V},$$

In the subsequent proof, we provide the constructions of $\mathrm{Attn}^K_{j,\widetilde{h}}$, $\mathrm{Attn}^Q_{j,\widetilde{h}}$, $\mathrm{Attn}^V_{j,\widetilde{h}}$ from Theorem D.1.

To apply Theorem D.1 to construct heads in Attn_m, let a and b denote the minimum and maximum of the inner products $k_i^\top x_m$, $q_i^\top x_m$, and $v_i^\top x_m$, over all $i \in [d_h]$ and $m \in [n]$:

$$a \le \min\{k_i^\top x_m, q_i^\top x_m, v_i^\top x_m\} \quad \text{and} \quad \max\{k_i^\top x_m, q_i^\top x_m, v_i^\top x_m\} \le b.$$

Next, we choose

$$H := \lceil \frac{2(b-a)}{(n-2)\epsilon_0} \rceil,$$

such that the interpolation error in Theorem D.1 is at most $\frac{\epsilon_0}{2}$ for any $\epsilon_0 > 0$.

Third, Theorem D.1 requires a single map $A : \mathbb{R}^{d\times n} \to \mathbb{R}^{(3d+n)\times n}$ shared across all H heads. In our construction, we realize this augmentation by prepending each head of Attn^H in Theorem D.1 with a head-specific linear map $A_{\widetilde{h}} : \mathbb{R}^{(d+6dd_h+n)\times n} \to \mathbb{R}^{(3d+n)\times n}$. The map $A_{\widetilde{h}}$ maps the input X_p to the desired dimension and picks out the target k_i, q_i or v_i (this is equivalent to the w_s in Theorem D.1.) to let Attn^H perform the desired linear transformation $k_i^\top X, v_i^\top X$ or $q_i^\top X$. Here $\widetilde{h}$ still identifies the single head assigned to a specific row. By Lemma D.3, $\mathrm{Attn}^H \circ A_{\widetilde{h}}$ remains an H-head attention. Therefore, we use $\mathrm{Attn}^H \circ A_{\widetilde{h}}$ to build the heads in Attn_m.

We construct $A_{\widetilde{h}}$ as

$$A_{\widetilde{h}} := \underbrace{\begin{bmatrix} I_d & 0_{d\times 3dd_h} & 0_{d\times 3dd_h} & 0_{d\times n} \\ 0_{d\times d} & E_{\widetilde{h}} & 0_{d\times 3dd_h} & 0_{d\times n} \\ 0_{d\times d} & E_{\widetilde{h}} & K_{\widetilde{h}}E_{\widetilde{h}} & 0_{d\times n} \\ 0_{n\times d} & 0_{n\times 3dd_h} & 0_{n\times 3dd_h} & I_n \end{bmatrix}}_{(3d+n)\times(d+6dd_h+n)},$$

where

$$E_{\widetilde{h}} := \underbrace{\begin{bmatrix} 0_{d\times(d\lceil \widetilde{h}/H \rceil)} & I_d & 0_{d\times(3dd_h - d\lceil \widetilde{h}/H\rceil - d)} \end{bmatrix}}_{d\times 3dd_h},$$

$$K_{\widetilde{h}} := [(\widetilde{h}\%H - 1)(n-2) - 1].$$

Here $\widetilde{h}\%H$ denotes the remainder of dividing $\widetilde{h}$ by H. We define $\%$ such that instead of the common $(kH)\%H = 0$,

$$kH\%H = H, \quad \text{for all} \quad k \in \mathbb{N}^+.$$

Applying $A_{\widetilde{h}}$ to X_p yields

$$A_{\widetilde{h}} \cdot X_p \coloneqq \underbrace{\begin{bmatrix} I_d & 0_{d\times 3dd_h} & 0_{d\times 3dd_h} & 0_{d\times n} \\ 0_{d\times d} & E_{\widetilde{h}} & 0_{d\times 3dd_h} & 0_{d\times n} \\ 0_{d\times d} & E_{\widetilde{h}} & K_{\widetilde{h}}E_{\widetilde{h}} & 0_{d\times n} \\ 0_{n\times d} & 0_{n\times 3dd_h} & 0_{n\times 3dd_h} & I_n \end{bmatrix}}_{(3d+n)\times(d+6dd_h+n)} \cdot \underbrace{\begin{bmatrix} X \\ W_{\text{in}} \\ I_n \end{bmatrix}}_{(d+6dd_h+n)\times n} \quad \text{(By the definition of } A_{\widetilde{h}} \text{ and } X_p\text{)}$$

$$= \underbrace{\begin{bmatrix} X \\ \begin{bmatrix} E_{\widetilde{h}} & 0_{d\times 3dd_h} \end{bmatrix} \cdot W_{\text{in}} \\ \begin{bmatrix} E_{\widetilde{h}} & K_{\widetilde{h}}E_{\widetilde{h}} \end{bmatrix} \cdot W_{\text{in}} \\ I_n \end{bmatrix}}_{(3d+n)\times n},$$

where $\begin{bmatrix} E_{\widetilde{h}} & 0_{d\times 3dd_h} \end{bmatrix} W_{\text{in}}$ expands as

$$\begin{aligned} & \begin{bmatrix} E_{\widetilde{h}} & 0_{d\times 3dd_h} \end{bmatrix} W_{\text{in}} \\ = & \underbrace{\begin{bmatrix} E_{\widetilde{h}} & 0_{d\times 3dd_h} \end{bmatrix}}_{d\times 6dd_h} \underbrace{\begin{bmatrix} 0\cdot w & 1\cdot w & 2\cdot w & \cdots & (n-1)\cdot w \\ w & w & w & \cdots & w \end{bmatrix}}_{6dd_h\times n} \\ = & \underbrace{E_{\widetilde{h}}}_{d\times 3dd_h} \cdot \underbrace{\begin{bmatrix} 0\cdot w & 1\cdot w & 2\cdot w & \cdots & (n-1)\cdot w \end{bmatrix}}_{3dd_h\times n} + \begin{bmatrix} 0_{d\times 3dd_h} \end{bmatrix} \underbrace{\begin{bmatrix} w & w & w & \cdots & w \end{bmatrix}}_{3dd_h\times n} \\ = & \underbrace{\begin{bmatrix} 0\cdot E_{\widetilde{h}}w & 1\cdot E_{\widetilde{h}}w & 2\cdot E_{\widetilde{h}}w & \cdots & (n-1)\cdot E_{\widetilde{h}}w \end{bmatrix}}_{d\times n}, \end{aligned}$$

and

$$E_{\widetilde{h}}w = \underbrace{\begin{bmatrix} 0_{d\times(d\lceil \widetilde{h}/H\rceil - d)} & I_d & 0_{d\times(3dd_h-(d\lceil\widetilde{h}/H\rceil-d)-d)} \end{bmatrix}}_{d\times 3dd_h} \underbrace{\begin{bmatrix} k_1 \\ \vdots \\ k_{d_h} \\ q_1 \\ \vdots \\ q_{d_h} \\ v_1 \\ \vdots \\ v_{d_h} \end{bmatrix}}_{3dd_h\times 1} \quad \text{(By the definition of } E_{\widetilde{h}} \text{ and } w\text{)}$$

$$= \begin{cases} k_{\lceil \widetilde{h}/H\rceil}, & 1 \le \widetilde{h} \le Hd_h \\ q_{\lceil \widetilde{h}/H\rceil - d_h}, & Hd_h < \widetilde{h} \le 2Hd_h \\ v_{\lceil \widetilde{h}/H\rceil - 2d_h}, & 2Hd_h < \widetilde{h} \le 3Hd_h \end{cases}. \tag{D.31}$$

The equality (D.31) holds since $E_{\widetilde{h}}$ selects the $\lceil \widetilde{h}/H\rceil$-th block in w.

Similarly, $[E_{\widetilde{h}} \quad K_{\widetilde{h}}E_{\widetilde{h}}] \cdot W_{\text{in}}$ expands as

$$\begin{aligned} & \begin{bmatrix} E_{\widetilde{h}} & K_{\widetilde{h}}E_{\widetilde{h}} \end{bmatrix} W_{\text{in}} \\ = & \underbrace{\begin{bmatrix} E_{\widetilde{h}} & K_{\widetilde{h}}E_{\widetilde{h}} \end{bmatrix}}_{d\times 6dd_h} \underbrace{\begin{bmatrix} 0\cdot w & 1\cdot w & 2\cdot w & \cdots & (n-1)\cdot w \\ w & w & w & \cdots & w \end{bmatrix}}_{6dd_h\times n} \quad \text{(By the definition of } W_{\text{in}}\text{)} \\ = & \underbrace{E_{\widetilde{h}}}_{d\times 3dd_h} \cdot \underbrace{\begin{bmatrix} 0\cdot w & 1\cdot w & 2\cdot w & \cdots & (n-1)\cdot w \end{bmatrix}}_{3dd_h\times n} + \underbrace{K_{\widetilde{h}}E_{\widetilde{h}}}_{d\times 3dd_h} \underbrace{\begin{bmatrix} w & w & w & \cdots & w \end{bmatrix}}_{3dd_h\times n} \\ & \text{(By } \begin{bmatrix} A & B \end{bmatrix} \begin{bmatrix} C \\ D \end{bmatrix} = AB + CD\text{)} \end{aligned}$$

$$= \underbrace{\begin{bmatrix} K_{\widetilde{h}} E_{\widetilde{h}} w & (K_{\widetilde{h}}+1) E_{\widetilde{h}} w & \cdots & (K_{\widetilde{h}}+n-1) E_{\widetilde{h}} w \end{bmatrix}}_{d \times n},$$

Up to here, we are capable of selecting a target k_i, q_i or v_i, and we start to build our heads in Attn_m.

When $1 \leq \widetilde{h} \leq H d_h$, we compute $A_{\widetilde{h}} \cdot X_p$ as

$$A_{\widetilde{h}} \cdot X_p = \underbrace{\begin{bmatrix} & & X & \\ 0 \cdot k_{\lceil \widetilde{h}/H \rceil} & 1 \cdot k_{\lceil \widetilde{h}/H \rceil} & \cdots & (n-1) \cdot k_{\lceil \widetilde{h}/H \rceil} \\ k_{\lceil \widetilde{h}/H \rceil} & k_{\lceil \widetilde{h}/H \rceil} & \cdots & k_{\lceil \widetilde{h}/H \rceil} \\ & & I_n & \end{bmatrix}}_{(3d+n) \times n}.$$

This means every $\widetilde{h}$ in $\{J+1, \ldots, J+H\}$ with $j \in [d_h]$ has the same $A_{\widetilde{h}} \cdot X_p$:

$$\underbrace{\begin{bmatrix} & & X & \\ 0 \cdot k_j & 1 \cdot k_j & \cdots & (n-1) \cdot k_j \\ k_j & k_j & \cdots & k_j \\ & & I_n & \end{bmatrix}}_{(3d+n) \times n}$$

For each $j \in [d_h]$, by Theorem D.1, there exists an H-head attention $\mathrm{Attn}'_j : \mathbb{R}^{(3d+n) \times n} \to \mathbb{R}^{3d_h \times n}$, such that the output satisfies

$$\| \underbrace{\mathrm{Attn}'_j(\overbrace{A_{\widetilde{h}} \cdot X_p}^{(3d+n) \times n})_{:,i}}_{3d_h \times 1} - (k_j^\top x_i) \underbrace{e_j^{(3d_h)}}_{3d_h \times 1} \|_\infty \leq \epsilon_0, \tag{D.32}$$

for every $i \in [n]$ and any $\epsilon_0 > 0$.

From (D.32), we have

$$\| \underbrace{\mathrm{Attn}'_j(\overbrace{A_{\widetilde{h}} \cdot X_p}^{(3d+n) \times n})}_{3d_h \times n} - \underbrace{e_j^{(3d_h)} k_j^\top X}_{3d_h \times n} \|_\infty \leq \epsilon_0,$$

where

$$e_j^{(3d_h)} k_j^\top X = \underbrace{\begin{bmatrix} (k_j^\top x_1) e_j^{(3d_h)} & (k_j^\top x_2) e_j^{(3d_h)} & \cdots & (k_j^\top x_n) e_j^{(3d_h)} \end{bmatrix}}_{3d_h \times n}.$$

We use $\mathrm{Attn}_j'^{(s)}$ to label the heads in Attn'_j, and we define $\mathrm{Attn}^K_{j,\widetilde{h}}(Z)$ to be

$$\mathrm{Attn}^K_{j,\widetilde{h}}(Z) := \mathrm{Attn}_j'^{(\widetilde{h})}(A_{\widetilde{h}} \cdot Z), \qquad (Z \in \mathbb{R}^{(d+6dd_h+n) \times n} \text{ denotes any input})$$

where $j \in [d_h]$ and $\widetilde{h} \in \{J+1, \ldots, J+H\}$.

By Lemma D.3, $\mathrm{Attn}^K_{j,\widetilde{h}}(Z)$ is still an attention.

Thus

$$\mathrm{Attn}^K_j(Z) := \sum_{\widetilde{h}=J+1}^{J+H} \mathrm{Attn}^K_{j,\widetilde{h}}(Z),$$

is also an attention.

Thus, we have

$$\| \underbrace{\mathrm{Attn}^K_j(X_p)}_{3d_h \times n} - \underbrace{e_j^{(3d_h)} k_j^\top X}_{3d_h \times n} \|_\infty \leq \epsilon_0.$$

This means that

$$\| \underbrace{\sum_{j=1}^{d_h} \mathrm{Attn}_j^K(X_p)}_{3d_h \times n} - \underbrace{\begin{bmatrix} K \\ 0_{d_h \times n} \\ 0_{d_h \times n} \end{bmatrix}}_{3d_h \times n} \|_\infty \leq \epsilon_0. \tag{D.33}$$

Similarly for Q, by (D.31), when $Hd_h < \widetilde{h} \leq 2Hd_h$, we have

$$E_{\widetilde{h}} w = \underbrace{q_{\lceil \widetilde{h}/H \rceil - d_h}}_{d \times 1},$$

and

$$A_{\widetilde{h}} \cdot X_p = \underbrace{\begin{bmatrix} & & X & \\ 0 \cdot q_{j-d_h} & 1 \cdot q_{j-d_h} & \cdots & (n-1) \cdot q_{j-d_h} \\ q_{j-d_h} & q_{j-d_h} & \cdots & q_{j-d_h} \\ & & I_n & \end{bmatrix}}_{(3d+n) \times n},$$

where $j \in [2d_h] \setminus [d_h]$.

For each $j \in [2d_h] \setminus [d_h]$, by Theorem D.1, there exists an H-head attention $\mathrm{Attn}_j'' : \mathbb{R}^{(3d+n) \times n} \to \mathbb{R}^{3d_h \times n}$, such that

$$\| \underbrace{\mathrm{Attn}_j''(A_{\widetilde{h}} \cdot X_p)}_{3d_h \times n} - \underbrace{e_j^{(3d_h)} q_{j-d_h}^\top X}_{3d_h \times n} \|_\infty \leq \epsilon_0,$$

for any $\epsilon_0 > 0$.

Then we construct $\mathrm{Attn}_{j,\widetilde{h}}^Q$ in a way similar to $\mathrm{Attn}_{j,\widetilde{h}}^K$

$$\mathrm{Attn}_{j,\widetilde{h}}^Q(Z) \coloneqq \mathrm{Attn}_j''^{(\widetilde{h})}(A_{\widetilde{h}} \cdot Z), \qquad (Z \in \mathbb{R}^{(d+6dd_h+n) \times n} \text{ denotes any input})$$

where $j \in [2d_h] \setminus [d_h]$ and $\widetilde{h} \in \{J+1, \ldots, J+H\}$.

By Lemma D.3, $\mathrm{Attn}_{j,\widetilde{h}}^Q(Z)$ is an attention.

Thus

$$\mathrm{Attn}_j^Q(Z) \coloneqq \sum_{\widetilde{h}=J+1}^{J+H} \mathrm{Attn}_{j,\widetilde{h}}^Q(Z),$$

is also an attention.

Thus, we have

$$\| \underbrace{\mathrm{Attn}_j^Q(X_p)}_{3d_h \times n} - \underbrace{e_j^{(3d_h)} q_{j-d_h}^\top X}_{3d_h \times n} \|_\infty \leq \epsilon_0.$$

This means that

$$\| \underbrace{\sum_{j=d_h+1}^{2d_h} \mathrm{Attn}_j^Q(X_p)}_{3d_h \times n} - \underbrace{\begin{bmatrix} 0_{d_h \times n} \\ Q \\ 0_{d_h \times n} \end{bmatrix}}_{3d_h \times n} \|_\infty \leq \epsilon_0. \tag{D.34}$$

For V, with analogous construction to that of K and Q, there exists an H-head attention $\mathrm{Attn}_j^V : \mathbb{R}^{(3d+n) \times n} \to \mathbb{R}^{3d_h \times n}$ such that

$$\| \underbrace{\mathrm{Attn}_j^V(X_p)}_{3d_h \times n} - \underbrace{e_j^{(3d_h)} v_{j-2d_h}^\top X}_{3d_h \times n} \|_\infty \leq \epsilon_0,$$

for each $j \in [3d_h] \setminus [2d_h]$ and any $\epsilon_0 > 0$.

This means that

$$\| \underbrace{\sum_{j=2d_h+1}^{3d_h} \mathrm{Attn}_j^V(X_p)}_{3d_h \times n} - \underbrace{\begin{bmatrix} 0_{d_h \times n} \\ 0_{d_h \times n} \\ V \end{bmatrix}}_{3d_h \times n} \|_\infty \le \epsilon_0. \tag{D.35}$$

Combining (D.33), (D.34) and (D.35), we have

$$\begin{aligned} & \|\mathrm{Attn}_m(X_p) - \begin{bmatrix} K \\ Q \\ V \end{bmatrix} \|_\infty \\ = & \| \sum_{j=1}^{d_h} \mathrm{Attn}_j^K(X_p) - \begin{bmatrix} K \\ 0_{d_h \times n} \\ 0_{d_h \times n} \end{bmatrix} + \sum_{j=d_h+1}^{2d_h} \mathrm{Attn}_j^Q(X_p) - \begin{bmatrix} 0_{d_h \times n} \\ Q \\ 0_{d_h \times n} \end{bmatrix} + \sum_{j=2d_h+1}^{3d_h} \mathrm{Attn}_j^V(X_p) - \begin{bmatrix} 0_{d_h \times n} \\ 0_{d_h \times n} \\ V \end{bmatrix} \|_\infty \\ \le & \epsilon_0. \end{aligned} \tag{D.36}$$

We define

$$\underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3d_h \times n} := \mathrm{Attn}_m(X_p).$$

Thus, (D.36) becomes

$$\| \underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3d_h \times n} - \underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3d_h \times n} \|_\infty \le \epsilon_0, \tag{D.37}$$

Step 3: Single-Head Assembly for Emulated Map. Our goal in this part is to reconstruct the attention mechanism

$$V'\mathrm{Softmax}_\beta((K')^\top Q'), \quad \text{and} \quad V\mathrm{Softmax}_\beta((K)^\top Q),$$

from K', Q', V' and K, Q, V via Attn_s.

In order to achieve this, we construct Attn_s to be

$$\mathrm{Attn}_s(Z) := \underbrace{\begin{bmatrix} 0_{d_h \times 2d_h} & I_{d_h} \end{bmatrix}}_{:=W_{V,s}} Z \cdot \mathrm{Softmax}_\beta((\underbrace{\begin{bmatrix} I_{d_h} & 0_{d_h \times 2d_h} \end{bmatrix}}_{:=W_{K,s}} Z)^\top \underbrace{\begin{bmatrix} 0_{d_h \times d_h} & I_{d_h} & 0_{d_h \times d_h} \end{bmatrix}}_{:=W_{Q,s}} Z),$$

where $Z \in \mathbb{R}^{3d_h \times n}$ denotes any input.

Thus, we have

$$\overbrace{\mathrm{Attn}_s(\underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3d_h \times n})}^{d_h \times n} = \underbrace{V}_{d_h \times n} \mathrm{Softmax}_\beta(\underbrace{(K)^\top Q}_{n \times n}), \tag{D.38}$$

and

$$\overbrace{\mathrm{Attn}_s(\underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3d_h \times n})}^{d_h \times n} = \underbrace{V'}_{d_h \times n} \mathrm{Softmax}_\beta(\underbrace{(K')^\top Q'}_{n \times n}). \tag{D.39}$$

Step 4: Error Bound. From (D.38) and (D.39), we have

$$\begin{aligned}
&\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix}) \\
&= \underbrace{V'}_{d_h\times n}\cdot\underbrace{\mathrm{Softmax}_\beta(K'^\top Q')}_{n\times n} - \underbrace{V}_{d_h\times n}\cdot\underbrace{\mathrm{Softmax}_\beta(K^\top Q)}_{n\times n} && \text{(By (D.39) and (D.38))} \\
&= V'\mathrm{Softmax}_\beta(K'^\top Q') - V\mathrm{Softmax}_\beta(K'^\top Q') + V\mathrm{Softmax}_\beta(K'^\top Q') - V\mathrm{Softmax}_\beta(K^\top Q) \\
&= (V'-V)\mathrm{Softmax}_\beta(K'^\top Q') + V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q)), && \text{(D.40)}
\end{aligned}$$

and the last equality follows from the distributivity of matrix multiplication.

Then, (D.40) yields

$$\begin{aligned}
&\|\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\|_\infty \\
&= \|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q') + V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty \\
&\leq \underbrace{\|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q')\|_\infty}_{:=(I)} + \underbrace{\|V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty}_{:=(II)}, && \text{(D.41)}
\end{aligned}$$

and the last inequality follows from the triangle inequality.

For term (I) in (D.41), since each column in $\mathrm{Softmax}_\beta(K'^\top Q')$ sums up to 1, then

$$\underbrace{(V'-V)}_{d_h\times n}\underbrace{\mathrm{Softmax}_\beta(K'^\top Q')_{:,j}}_{n\times 1},$$

is a weighted sum of the columns in $(V'-V)$.

Thus we have

$$\|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q')_{:,j}\|_\infty \leq \|V'-V\|_\infty \leq \epsilon_0,$$

and the first inequality holds since the column average of $(V'-V)$ has a maximum entry no greater than the maximum entry among the original columns in $(V'-V)$. The second inequality holds since (D.37).

Then we get

$$(I) \leq \epsilon_0. \tag{D.42}$$

Term (II) in (D.41) is

$$(II) = \|V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty.$$

For simplicity of presentation, we define

$$\Delta S := \mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q),$$

such that for each entry in (II), we have

$$\begin{aligned}
|(V\Delta S)_{ij}| &= |\sum_{k=1}^n V_{ik}(\Delta S)_{kj}| && \text{(By the definition of matrix multiplication)} \\
&\leq \sum_{k=1}^n |V_{ik}|\cdot|(\Delta S)_{kj}| && \text{(By triangle inequality and } |ab| = |a|\cdot|b| \text{ for all } a,b\in\mathbb{R}) \\
&\leq \sum_{k=1}^n \|V\|_\infty\cdot\|\Delta S\|_\infty && \text{(By } |V_{ik}|\leq\|V\|_\infty \text{ and } |(\Delta S)_{kj}|\leq\|\Delta S\|_\infty) \\
&= n\|V\|_\infty\cdot\|\Delta S\|_\infty,
\end{aligned}$$

and this leads to

$$(II) \leq n\|V\|_\infty\cdot\|\Delta S\|_\infty. \tag{D.43}$$

For each entry in ΔS, we have

$$
\begin{aligned}
& |(\Delta S)_{i,j}| \quad \text{(D.44)} \\
= & |(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))_{i,j}| \\
= & |\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}| \quad (K_i', Q_i', K_i, Q_i \text{ denote the } i\text{-th column in } K', Q', K, Q) \\
= & |\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} + \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}| \\
\leq & |\frac{e^{\beta K_i'\cdot Q_j'} - e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}| + |e^{\beta K_i\cdot Q_j}(\frac{1}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{1}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}})| \quad (\text{By triangle inequality}) \\
= & \frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}|1 - e^{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')}| + \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}|\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - 1| \\
& \quad (\text{By non-negativity of exponential}) \\
< & \underbrace{|1 - e^{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')}|}_{:=(II\text{-}1)} + \underbrace{|1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}|}_{:=(II\text{-}2)}, \quad \text{(D.45)}
\end{aligned}
$$

and the last inequality holds since

$$
\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} < 1, \quad \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}} < 1.
$$

To bound term $(II\text{-}1)$ in (D.45), we recall

$$
\| \underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3d_h\times n} - \underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3d_h\times n} \|_\infty \leq \epsilon_0,
$$

so

$$
\begin{aligned}
\| \underbrace{K' - K}_{d_h\times n} \|_\infty \leq \epsilon_0, \\
\| \underbrace{Q' - Q}_{d_h\times n} \|_\infty \leq \epsilon_0.
\end{aligned}
$$

Let K_i', Q_i', K_i, Q_i denote the i-th column in K', Q', K, Q, then we have

$$
\begin{aligned}
\underbrace{\Delta K_i}_{d_h\times 1} := \underbrace{K_i' - K_i}_{d_h\times 1}, \quad \|\Delta K_i\|_\infty \leq \epsilon_0, \\
\underbrace{\Delta Q_i}_{d_h\times 1} := \underbrace{Q_i' - Q_i}_{d_h\times 1}, \quad \|\Delta Q_i\|_\infty \leq \epsilon_0.
\end{aligned}
$$

Thus, for term $(II\text{-}1)$ in (D.45), we have

$$
\begin{aligned}
& (II\text{-}1) \\
= & |1 - \exp\{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')\}| \\
= & |1 - \exp\{\beta(K_i\cdot Q_j - (K_i + \Delta K_i)\cdot(Q_j + \Delta Q_j))\}| \\
& \quad (\text{By } K_i' = K_i + \Delta K_i \text{ and } Q_i' = Q_i + \Delta Q_i) \\
= & |1 - \exp\{-\beta(K_i\cdot \Delta Q_j + Q_j\cdot \Delta K_i + \Delta K_i\cdot \Delta Q_j)\}|, \\
& \quad (\text{By } K_i\cdot Q_j - (K_i + \Delta K_i)\cdot(Q_i + \Delta Q_i) = -(K_i\cdot \Delta Q_j + Q_j\cdot \Delta K_i + \Delta K_i\cdot \Delta Q_j))
\end{aligned}
$$

and we know

$$
\begin{aligned}
& K_i\cdot \Delta Q_j + Q_j\cdot \Delta K_i + \Delta K_i\cdot \Delta Q_j \\
\leq & d_h\cdot \|K_i\|_\infty\|\Delta Q_j\|_\infty + d_h\cdot \|Q_j\|_\infty\|\Delta K_i\|_\infty + d_h\cdot \|\Delta K_i\|_\infty\|\Delta Q_j\|_\infty \\
& \quad (\text{By } a\cdot b \leq d_h\|a\|_\infty\|b\|_\infty \text{ for all } a, b \in \mathbb{R}^{d_h})
\end{aligned}
$$

$$\leq 2d_h B_{KQV}\epsilon_0 + d_h\epsilon_0^2. \qquad (\text{By } \|K_i\|_\infty, \|Q_j\|_\infty \leq B_{KQV} \text{ and } \|\Delta K_i\|_\infty, \|\Delta Q_j\|_\infty \leq \epsilon_0)$$

Thus, we have

$$(II\text{-}1) \leq |1 - e^{-\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0^2)}|. \tag{D.46}$$

For term $(II\text{-}2)$ in (D.45), we have

$$\begin{aligned}
&(II\text{-}2)\\
&= |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K'_{i'}\cdot Q'_j}}| && (\text{By the definition of } (II\text{-}2))\\
&= |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta (K_{i'}+\Delta K_{i'})\cdot (Q_j+\Delta Q_j)}}| && (\text{By } K'_{i'} = K_{i'} + \Delta K_{i'} \text{ and } Q'_i = Q_i + \Delta Q_i)\\
&= |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta (K_{i'}\cdot Q_j + K_{i'}\cdot \Delta Q_j + Q_j\cdot\Delta K_{i'} + \Delta K_{i'}\cdot\Delta Q_j)}}|,
\end{aligned}$$

and for all i' in the denominator, we have

$$\begin{aligned}
&K_{i'}\cdot Q_j + K_{i'}\cdot\Delta Q_j + Q_j\cdot\Delta K_{i'} + \Delta K_{i'}\cdot\Delta Q_j\\
&\leq K_{i'}\cdot Q_j + d_h\cdot\|K_{i'}\|_\infty\|\Delta Q_j\|_\infty + d_h\cdot\|Q_j\|_\infty\|\Delta K_{i'}\|_\infty + d_h\cdot\|\Delta K_{i'}\|_\infty\|\Delta Q_j\|_\infty\\
&\qquad (\text{By } a\cdot b \leq d_h\|a\|_\infty\|b\|_\infty \text{ for all } a, b \in \mathbb{R}^{d_h})\\
&\leq K_{i'}\cdot Q_j + 2d_h B_{KQV}\epsilon_0 + d_h\epsilon_0^2. \qquad (\text{By } \|K_{i'}\|_\infty, \|Q_j\|_\infty \leq B_{KQV} \text{ and } \|\Delta K_{i'}\|_\infty, \|\Delta Q_j\|_\infty \leq \epsilon_0)
\end{aligned}$$

Thus,

$$\begin{aligned}
&(II\text{-}2)\\
&= |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta (K_{i'}\cdot Q_j + K_{i'}\cdot \Delta Q_j + Q_j\cdot\Delta K_{i'} + \Delta K_{i'}\cdot\Delta Q_j)}}|\\
&\leq |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta (K_{i'}\cdot Q_j + 2d_h B_{KQV}\epsilon_0 + d_h\epsilon_0^2)}}|\\
&= |1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{e^{\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0^2)}\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}| && (e^{\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0^2)} \text{ is independent of } i')\\
&= |1 - e^{-\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0^2)}|,
\end{aligned} \tag{D.47}$$

and the last equality holds since the common factor $\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}$ cancels out.

Combining (D.45), (D.46), and (D.47), we have

$$\begin{aligned}
&|(\text{Softmax}_\beta(K'^\top Q') - \text{Softmax}_\beta(K^\top Q))_{i,j}|\\
&< 2|1 - e^{-\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0^2)}|\\
&\leq 2|1 - e^{-\beta d_h(2B_{KQV}\epsilon_0+\epsilon_0)}|. && (\text{By requiring } 0 < \epsilon_0 \leq 1)
\end{aligned}$$

Thus for any $0 < \epsilon_1 < 2$, when ϵ_0 satisfies

$$0 < \epsilon_0 \leq \min\{1, \frac{-\ln(1-\frac{\epsilon_1}{2})}{\beta d_h(2B_{KQV}+1)}\},$$

we have

$$|(\text{Softmax}_\beta(K'^\top Q') - \text{Softmax}_\beta(K^\top Q))_{i,j}| < \epsilon_1. \tag{D.48}$$

From (D.43) and (D.48), we have

$$(II) \leq n\|V\|_\infty\|\Delta S\|_\infty < nB_{KQV}\epsilon_1, \tag{D.49}$$

since $\|V\|_\infty \leq B_{KQV}$ and $\|\Delta S\|_\infty < \epsilon_1$.

Combining (D.41) with (D.42) and (D.49) yields

$$\|\text{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \text{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\|_\infty < \epsilon_0 + nB_{KQV}\epsilon_1.$$

When we take ϵ_0 and ϵ_1 to be infinitely small, the right-hand side tends to 0.

This completes the proof. □

D.6 Proof of Theorem 4.2

Theorem D.6 (Theorem 4.2 Restate)**.** Let $X \in \mathbb{R}^{d\times n}$ be the input sequence, and let $W_K, W_Q, W_V \in \mathbb{R}^{n\times d}$ be the weight matrices of the target attention. Assume $B = \max\{\|X\|_\infty, \|W_K\|_\infty, \|W_Q\|_\infty, \|W_V\|_\infty\}$ and $\|W_K X\|_\infty, \|W_Q X\|_\infty, \|W_V X\|_\infty \le B_{KQV}$ for $B_{KQV} \ge 0$. Then, for any $\epsilon > 0$, there exists a single-head attention layer Attn_s followed by a multi-head attention layer with linear projections such that

$$\|\mathrm{Attn}_s \circ (\sum_{j=1}^{3n} \mathrm{Attn}_j \circ \mathrm{Linear}_j(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix})) - \underbrace{W_V X}_{n\times n} \mathrm{Softmax}_\beta \underbrace{((W_K X)^\top W_Q X)}_{n\times n} \|_\infty \le \epsilon.$$

Proof. Follow our proof sketch in Appendix A.2, our proof consists of four conceptual steps.

Step 1: Encoding Weights into the Input. For clarity and simplicity, we define

$$k_i := (W_K^\top)_{:,i} \in \mathbb{R}^d,$$
$$q_i := (W_Q^\top)_{:,i} \in \mathbb{R}^d,$$
$$v_i := (W_V^\top)_{:,i} \in \mathbb{R}^d,$$

such that $W_K^\top, W_Q^\top, W_V^\top$ writes out as

$$W_K^\top = \underbrace{[k_1 \quad k_2 \quad \cdots \quad k_n]}_{d\times n}, \quad W_Q^\top = \underbrace{[q_1 \quad q_2 \quad \cdots \quad q_n]}_{d\times n}, \quad W_V^\top = \underbrace{[v_1 \quad v_2 \quad \cdots \quad v_n]}_{d\times n}.$$

Then, we express the input as

$$\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix} = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ k_1 & k_2 & \cdots & k_n \\ q_1 & q_2 & \cdots & q_n \\ v_1 & v_2 & \cdots & v_n \end{bmatrix}}_{4d\times n} \tag{D.50}$$

where x_i, k_i, q_i and v_i are all d dimensional vectors for $i \in [n]$.

Step 2: Multi-Head Approximation of K, Q, V. For the simplicity of presentation, we define

$$K := \underbrace{W_K}_{n\times d} \underbrace{X}_{d\times n}, \quad Q := \underbrace{W_Q}_{n\times d} \underbrace{X}_{d\times n}, \quad V := \underbrace{W_V}_{n\times d} \underbrace{X}_{d\times n}.$$

Writing W_K, W_Q, and W_V row-wise as

$$W_K = \underbrace{\begin{bmatrix} k_1^\top \\ k_2^\top \\ \vdots \\ k_n^\top \end{bmatrix}}_{n\times d}, \quad W_Q = \underbrace{\begin{bmatrix} q_1^\top \\ q_2^\top \\ \vdots \\ q_n^\top \end{bmatrix}}_{n\times d}, \quad W_V = \underbrace{\begin{bmatrix} v_1^\top \\ v_2^\top \\ \vdots \\ v_n^\top \end{bmatrix}}_{n\times d},$$

and $X = [x_1 \quad \cdots \quad x_n]$, we express K, Q, and V entry-wise as

$$K = \begin{bmatrix} k_1^\top x_1 & k_1^\top x_2 & \cdots & k_1^\top x_n \\ k_2^\top x_1 & k_2^\top x_2 & \cdots & k_2^\top x_n \\ \vdots & \vdots & \vdots & \vdots \\ k_n^\top x_1 & k_n^\top x_2 & \cdots & k_n^\top x_n \end{bmatrix},$$

$$Q = \begin{bmatrix} q_1^\top x_1 & q_1^\top x_2 & \cdots & q_1^\top x_n \\ q_2^\top x_1 & q_2^\top x_2 & \cdots & q_2^\top x_n \\ \vdots & \vdots & \vdots & \vdots \\ q_n^\top x_1 & q_n^\top x_2 & \cdots & q_n^\top x_n \end{bmatrix},$$
$$V = \begin{bmatrix} v_1^\top x_1 & v_1^\top x_2 & \cdots & v_1^\top x_n \\ v_2^\top x_1 & v_2^\top x_2 & \cdots & v_2^\top x_n \\ \vdots & \vdots & \vdots & \vdots \\ v_n^\top x_1 & v_n^\top x_2 & \cdots & v_n^\top x_n \end{bmatrix}.$$

Here $k_i^\top$, $q_i^\top$, and $v_i^\top$ identify the i-th row of K, Q and V, while x_j identifies the j-th column.

In this section, our goal is to approximate K, Q, and V. Our strategy is to approximate K, Q, and V row by row, and within each row, entry by entry. More precisely, for each $i \in [n]$, we approximate

$$k_i^\top X, \quad q_i^\top X, \quad v_i^\top X,$$

by approximating the scalar products

$$k_i^\top x_j, \quad q_i^\top x_j, \quad v_i^\top x_j, \quad \text{for all } j \in [n],$$

and then collecting these approximations to form approximations of the full matrices K, Q, and V.

To approximate each scalar $k_i^\top x_j$, $q_i^\top x_j$, and $v_i^\top x_j$, we first determine their joint range over all $i, j \in [n]$. Within this joint range, we construct a set of uniform-space grid points. Then, we approximate each target entry $k_i^\top x_j$, $q_i^\top x_j$, or $v_i^\top x_j$ by an entry-specific weighted sum of these grid points, where grid points closer to the target entry value receive larger weights. In this way, we represent every entry by its own set of interpolation weights, while all approximations share the same global grid.

We introduce our notation for the uniform grid points used in our interpolation scheme.

Interpolations. We recall

$$B = \max(\|X\|_\infty, \|W_K\|_\infty, \|W_Q\|_\infty, \|W_V\|_\infty).$$

Thus, for all $i, j \in [n]$,

$$-d \cdot B^2 \leq k_i^\top x_j, q_i^\top x_j, v_i^\top x_j \leq d \cdot B^2. \tag{D.51}$$

Namely, $[-dB^2, dB^2]$ contains all entries of K, Q, and V.

Then, we take $L_0 := -dB^2$ and $L_P := dB^2$ as the two endpoints of our interpolation and define for $i \in \{0\} \cup [P]$

$$L_i := \frac{iL_P + (P-i)L_0}{P}, \tag{D.52}$$

where P is the number of interpolation steps (the number of equal divisions of $[L_0, L_P]$). The points $\{L_i\}_{i=0}^P$ form our uniform grid over the target range.

We use ΔL to denote the length of the interval between two neighboring grid points. We have

$$\Delta L := \frac{L_P - L_0}{P} = \frac{2dB^2}{P}. \tag{D.53}$$

Now we have all the ingredients needed to approximate each entry using a weighted sum. However, the input,

$$\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix} = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ k_1 & k_2 & \cdots & k_n \\ q_1 & q_2 & \cdots & q_n \\ v_1 & v_2 & \cdots & v_n \end{bmatrix}}_{4d \times n},$$

contains information from all rows in the target K, Q, and V, but does not contain the grid points. We need a mechanism to select a specific k_i, q_i, or v_i (corresponding to one row of K, Q, or V) and to include the grid points for us.

To address this, we introduce row-specific linear transformations Linear_j, where $j \in [3n]$, since we have n rows for each of K, Q, and V. Each Linear_j serves two purposes: it incorporates the input and the uniform grid points, and selects the k_i, q_i, or v_i associated with index i (corresponding to one row of K, Q, or V).

For the clarity of presentation, we relabel these $3n$ linear transformations according to whether they are responsible for K, Q, or V

$$\begin{aligned} \mathrm{Linear}_j^K &\coloneqq \mathrm{Linear}_j, \quad j \in [n], && \text{(Responsible for } K\text{)} \\ \mathrm{Linear}_j^Q &\coloneqq \mathrm{Linear}_{n+j}, \quad j \in [n], && \text{(Responsible for } Q\text{)} \\ \mathrm{Linear}_j^V &\coloneqq \mathrm{Linear}_{2n+j}, \quad j \in [n]. \end{aligned} \tag{D.54}$$

Later in the proof, we specify the explicit form of these Linear_j.

So far, Linear_j allows us to combine the input with the uniform grid points and to select the desired k_i, q_i, or v_i. The next step is to *implement* the entry-specific weighted sums to approximate the entries of K, Q, and V.

For this, we use a row-specific single-head attention: for each $i \in [n]$, we assign one head to approximate $k_i^\top X$ using the weighted sum, one head to approximate $q_i^\top X$ in the same manner, and one head to approximate $v_i^\top X$ in the same manner. Each such head operates token-wise: given its designated row i, the head approximates all scalars $k_i^\top x_j$, $q_i^\top x_j$, or $v_i^\top x_j$ across $j \in [n]$.

Since each of K, Q, and V has n rows and we use a single-head for each row, we use a total of $3n$ heads to approximate K, Q, and V. We use Attn_j to label these $3n$ heads and $j \in [3n]$.

Again, for the clarity of presentation, we provide another equivalent way, as Attn_j, to label these $3n$ heads

$$\begin{aligned} \mathrm{Attn}_j^K &\coloneqq \mathrm{Attn}_j, \quad j \in [n], \\ \mathrm{Attn}_j^Q &\coloneqq \mathrm{Attn}_{n+j}, \quad j \in [n], \\ \mathrm{Attn}_j^V &\coloneqq \mathrm{Attn}_{2n+j}, \quad j \in [n]. \end{aligned} \tag{D.55}$$

Later in our proof, we provide the construction of these Attn_j explicitly.

Now we are ready to approximate each of K, Q, and V. We approximate K first to demonstrate our procedure and deal with Q and V in a similar manner later.

In-Context Calculation of K. First, we define the linear transformation $\mathrm{Linear}_j^K : \mathbb{R}^{4d\times n} \to \mathbb{R}^{(2d+3)\times(P+1)}$ attached before Attn_j^K as:

$$\begin{aligned} \mathrm{Linear}_j^K(Z) \coloneqq\ & \underbrace{\begin{bmatrix} 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & I_d & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} 2L_0 e_j^{(n)} & 2L_1 e_j^{(n)} & \cdots & 2L_P e_j^{(n)} \end{bmatrix}}_{n\times(P+1)} + \\ & && \text{(} Z \in \mathbb{R}^{4d\times n} \text{ denotes any input)} \\ & \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} I_n & 0_{n\times(P+1-n)} \end{bmatrix}}_{n\times(P+1)} + \underbrace{\begin{bmatrix} 0_{2d\times(P+1)} \\ M_1 \\ M_L \end{bmatrix}}_{(2d+3)\times(P+1)}, \end{aligned}$$

where M_1, M_L are

$$M_1 \coloneqq \underbrace{\begin{bmatrix} 1_{1\times n} & 0_{1\times(P+1-n)} \end{bmatrix}}_{1\times(P+1)}, \tag{D.56}$$

$$M_L \coloneqq \underbrace{\begin{bmatrix} L_0 & L_1 & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_P^2 \end{bmatrix}}_{2\times(P+1)}. \tag{D.57}$$

The Linear_j^K layer takes the input $\begin{bmatrix} X^\top & W_K & W_Q & W_V \end{bmatrix}^\top$ and outputs in the following way:

$$
\begin{aligned}
&\text{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) \\
=& \underbrace{\begin{bmatrix} 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & I_d & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ k_1 & k_2 & \cdots & k_n \\ q_1 & q_2 & \cdots & q_n \\ v_1 & v_2 & \cdots & v_n \end{bmatrix}}_{4d\times n} \underbrace{\begin{bmatrix} 2L_0 e_j^{(n)} & 2L_1 e_j^{(n)} & \cdots & 2L_P e_j^{(n)} \end{bmatrix}}_{n\times (P+1)} + \\
& \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ k_1 & k_2 & \cdots & k_n \\ q_1 & q_2 & \cdots & q_n \\ v_1 & v_2 & \cdots & v_n \end{bmatrix}}_{4d\times n} \underbrace{\begin{bmatrix} I_n & 0_{n\times (P+1-n)} \end{bmatrix}}_{n\times (P+1)} + \underbrace{\begin{bmatrix} 0_{2d\times (P+1)} \\ M_1 \\ M_L \end{bmatrix}}_{(2d+3)\times (P+1)} \quad \text{(By (D.50))} \\
=& \underbrace{\begin{bmatrix} 0_{d\times 1} & 0_{d\times 1} & \cdots & 0_{d\times 1} \\ k_1 & k_2 & \cdots & k_n \\ 0_{3\times 1} & 0_{3\times 1} & \cdots & 0_{3\times 1} \end{bmatrix}}_{(2d+3)\times n} \underbrace{\begin{bmatrix} 2L_0 e_j^{(n)} & 2L_1 e_j^{(n)} & \cdots & 2L_P e_j^{(n)} \end{bmatrix}}_{n\times (P+1)} + \\
& \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ 0_{d\times 1} & 0_{d\times 1} & \cdots & 0_{d\times 1} \\ 0_{3\times 1} & 0_{3\times 1} & \cdots & 0_{3\times 1} \end{bmatrix}}_{(2d+3)\times n} \underbrace{\begin{bmatrix} I_n & 0_{n\times (P+1-n)} \end{bmatrix}}_{n\times (P+1)} + \underbrace{\begin{bmatrix} 0_{2d\times (P+1)} \\ M_1 \\ M_L \end{bmatrix}}_{(2d+3)\times (P+1)} \\
& \text{(By selecting } k_i \text{ and } x_i \text{ with } I_d \text{ for all } i \in [n]) \\
=& \underbrace{\begin{bmatrix} 0_{d\times 1} & 0_{d\times 1} & \cdots & 0_{d\times 1} \\ 2L_0 k_j & 2L_1 k_j & \cdots & 2L_P k_j \\ 0_{3\times 1} & 0_{3\times 1} & \cdots & 0_{3\times 1} \end{bmatrix}}_{(2d+3)\times (P+1)} + \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times 1} & \cdots & 0_{d\times 1} \\ 0_{d\times 1} & 0_{d\times 1} & \cdots & 0_{d\times 1} & 0_{d\times 1} & \cdots & 0_{d\times 1} \\ 0_{3\times 1} & 0_{3\times 1} & \cdots & 0_{3\times 1} & 0_{3\times 1} & \cdots & 0_{3\times 1} \end{bmatrix}}_{(2d+3)\times (P+1)} + \\
& \text{(By selecting } k_j \text{ with } e_j^{(n)}) \\
& \underbrace{\begin{bmatrix} 0_{2d\times 1} & 0_{2d\times 1} & \cdots & 0_{2d\times 1} & 0_{2d\times 1} & \cdots & 0_{2d\times 1} \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & -L_n^2 & \cdots & -L_P^2 \end{bmatrix}}_{(2d+3)\times (P+1)} \\
& \text{(By the definition of } M_1 \text{ and } M_L\text{; i.e., (D.56) and (D.57))} \\
=& \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_d & \cdots & 0_d \\ 2L_0 k_j & 2L_1 k_j & \cdots & 2L_{n-1} k_j & 2L_n k_j & \cdots & 2L_P k_j \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & -L_n^2 & \cdots & -L_P^2 \end{bmatrix}}_{(2d+3)\times (P+1)}.
\end{aligned}
\tag{D.58}
$$

Next, we construct $\text{Attn}_j^K : \mathbb{R}^{(2d+3)\times(P+1)} \to \mathbb{R}^{3n\times n}$ to be

$$
\text{Attn}_j^K(D) := \underbrace{W_{\widehat{V}}^{K;j} D}_{3n\times (P+1)} \cdot \underbrace{\text{Softmax}_\beta((W_{\widehat{K}}^{K;j} D)^\top W_{\widehat{Q}}^{K;j} D)}_{(P+1)\times (P+1)} \cdot \underbrace{W_{\widehat{O}}^{K;j}}_{(P+1)\times n},
$$

where $D \in \mathbb{R}^{(2d+3)\times(P+1)}$ denotes any input, and

$$
W_{\widehat{K}}^{K;j} := \underbrace{\begin{bmatrix} 0_{d\times d} & I_d & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 0 & 0 & 1 \end{bmatrix}}_{(d+1)\times (2d+3)},
\tag{D.59}
$$

$$W_{\widehat{Q}}^{K;j} := \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 1 & 0 & 0 \end{bmatrix}}_{(d+1)\times(2d+3)}, \tag{D.60}$$

$$W_{\widehat{V}}^{K;j} := \underbrace{e_j^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} 0_{1\times(2d+1)} & 1 & 0 \end{bmatrix}}_{1\times(2d+3)}, \tag{D.61}$$

$$W_{\widehat{O}}^{K;j} := \underbrace{\begin{bmatrix} I_n \\ 0_{(P+1-n)\times n} \end{bmatrix}}_{(P+1)\times n}. \tag{D.62}$$

We define the $\widehat{K}_j^K$ of Attn_j^K to be:

$$\begin{aligned}
\widehat{K}_j^K &:= W_{\widehat{K}}^{K;j} \cdot \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) \\
&= \underbrace{\begin{bmatrix} 0_{d\times d} & I_d & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 0 & 0 & 1 \end{bmatrix}}_{(d+1)\times(2d+3)} \cdot \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & \cdots & 0_d \\ 2L_0 k_j & 2L_1 k_j & \cdots & 2L_{n-1} k_j & \cdots & 2L_P k_j \\ 1 & 1 & \cdots & 1 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}}_{(2d+3)\times(P+1)} \\
&\qquad \text{(By (D.59) and (D.58))} \\
&= \underbrace{\begin{bmatrix} 2L_0 k_j & 2L_1 k_j & \cdots & 2L_{n-1} k_j & \cdots & 2L_P k_j \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}}_{(d+1)\times(P+1)},
\end{aligned} \tag{D.63}$$

and the last equality holds since I_d selects the $2L_i k_j$ row, and 1 selects the $-L_i^2$ row where $i \in \{0\} \cup [P]$.

We define the $\widehat{Q}_j^K$ of Attn_j^K to be:

$$\begin{aligned}
\widehat{Q}_j^K &:= W_{\widehat{Q}}^{K;j} \cdot \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) \\
&= \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 1 & 0 & 0 \end{bmatrix}}_{(d+1)\times(2d+3)} \cdot \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & \cdots & 0_d \\ 2L_0 k_j & 2L_1 k_j & \cdots & 2L_{n-1} k_j & \cdots & 2L_P k_j \\ 1 & 1 & \cdots & 1 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}}_{(2d+3)\times(P+1)} \\
&\qquad \text{(By (D.60) and (D.58))} \\
&= \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times(P+1-n)} \\ 1 & 1 & \cdots & 1 & 0_{1\times(P+1-n)} \end{bmatrix}}_{(d+1)\times(P+1)},
\end{aligned} \tag{D.64}$$

and the last equality holds since I_d selects the x_i row where $i \in [n]$, and 1 selects the 1_s row.

We define the $\widehat{V}_j^K$ of Attn_j^K to be:

$$\widehat{V}_j^K := W_{\widehat{V}}^{K;j} \cdot \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix})$$

$$= \underbrace{e_j^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} 0_{1\times(2d+1)} & 1 & 0 \end{bmatrix}}_{1\times(2d+3)} \cdot \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_d & \cdots & 0_d \\ 2L_0k_j & 2L_1k_j & \cdots & 2L_{n-1}k_j & 2L_nk_j & \cdots & 2L_Pk_j \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & -L_n^2 & \cdots & -L_P^2 \end{bmatrix}}_{(2d+3)\times(P+1)}$$

(By (D.61) and (D.58))

$$= \underbrace{e_j^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \end{bmatrix}}_{1\times(P+1)}, \tag{D.65}$$

and the last equality holds since the 1 selects the L_i row where $i \in \{0\} \cup [P]$.

Combining the results of $\widehat{K}_j^K$ and $\widehat{Q}_j^K$, we calculate the $\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K)$ in Attn_j^K as

$$\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K)$$

$$= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} 2L_0k_j & 2L_1k_j & \cdots & 2L_{n-1}k_j & \cdots & 2L_Pk_j \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}^\top}_{(P+1)\times(d+1)} \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times(P+1-n)} \\ 1 & 1 & \cdots & 1 & 0_{1\times(P+1-n)} \end{bmatrix}}_{(d+1)\times(P+1)})$$

(By the definition of $\widehat{K}_j^K$ and $\widehat{Q}_j^K$; i.e. (D.63) and (D.64))

$$= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} 2L_0k_j^\top & -L_0^2 \\ 2L_1k_j^\top & -L_1^2 \\ \vdots & \vdots \\ 2L_Pk_j^\top & -L_P^2 \end{bmatrix}}_{(P+1)\times(d+1)} \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times(P+1-n)} \\ 1 & 1 & \cdots & 1 & 0_{1\times(P+1-n)} \end{bmatrix}}_{(d+1)\times(P+1)})$$

(By the transpose of $\widehat{K}_j^K$)

$$= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} 2L_0k_j^\top x_1 - L_0^2 & 2L_0k_j^\top x_2 - L_0^2 & \cdots & 2L_0k_j^\top x_n - L_0^2 & 0_{d\times(P+1-n)} \\ 2L_1k_j^\top x_1 - L_1^2 & 2L_1k_j^\top x_2 - L_1^2 & \cdots & 2L_1k_j^\top x_n - L_1^2 & 0_{d\times(P+1-n)} \\ \vdots & \vdots & & \vdots & \vdots \\ 2L_Pk_j^\top x_1 - L_P^2 & 2L_Pk_j^\top x_2 - L_P^2 & \cdots & 2L_Pk_j^\top x_n - L_P^2 & 0_{d\times(P+1-n)} \end{bmatrix}}_{(P+1)\times(P+1)})$$

(By matrix multiplication)

$$= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} -(k_j^\top x_1 - L_0)^2 + (k_j^\top x_1)^2 & \cdots & -(k_j^\top x_n - L_0)^2 + (k_j^\top x_n)^2 & 0_{d\times(P+1-n)} \\ -(k_j^\top x_1 - L_1)^2 + (k_j^\top x_1)^2 & \cdots & -(k_j^\top x_n - L_1)^2 + (k_j^\top x_n)^2 & 0_{d\times(P+1-n)} \\ \vdots & & \vdots & \vdots \\ -(k_j^\top x_1 - L_P)^2 + (k_j^\top x_1)^2 & \cdots & -(k_j^\top x_n - L_P)^2 + (k_j^\top x_n)^2 & 0_{d\times(P+1-n)} \end{bmatrix}}_{(P+1)\times(P+1)})$$

($2L_ik_j^\top x_m - L_i^2 = -(k_j^\top x_m - L_i)^2 + (k_j^\top x_m)^2$ where $i \in \{0\} \cup [P]$ and $m \in [n]$)

$$= \mathrm{Softmax}_\beta(\underbrace{\begin{bmatrix} -(k_j^\top x_1 - L_0)^2 & \cdots & -(k_j^\top x_n - L_0)^2 & 0_{d\times(P+1-n)} \\ -(k_j^\top x_1 - L_1)^2 & \cdots & -(k_j^\top x_n - L_1)^2 & 0_{d\times(P+1-n)} \\ \vdots & & \vdots & \vdots \\ -(k_j^\top x_1 - L_P)^2 & \cdots & -(k_j^\top x_n - L_P)^2 & 0_{d\times(P+1-n)} \end{bmatrix}}_{(P+1)\times(P+1)}), \tag{D.66}$$

and the last line holds since the following property of $\mathrm{Softmax}_\beta$

$$\mathrm{Softmax}_\beta(v) = \mathrm{Softmax}_\beta(v + C \cdot 1_{(P+1)\times 1}),$$

for any vector $v \in \mathbb{R}^{P+1}$ and $C \in \mathbb{R}$.

From (D.66), we have

$$\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j}$$

$$= \text{Softmax}_\beta(\underbrace{\begin{bmatrix} -(k_j^\top x_1 - L_0)^2 & \cdots & -(k_j^\top x_n - L_0)^2 & 0_{d\times(P-n+1)} \\ -(k_j^\top x_1 - L_1)^2 & \cdots & -(k_j^\top x_n - L_1)^2 & 0_{d\times(P-n+1)} \\ \vdots & & \vdots & \vdots \\ -(k_j^\top x_1 - L_P)^2 & \cdots & -(k_j^\top x_n - L_P)^2 & 0_{d\times(P-n+1)} \end{bmatrix}}_{(P+1)\times(P+1)}) \underbrace{\begin{bmatrix} I_n \\ 0_{(P+1-n)\times n} \end{bmatrix}}_{(P+1)\times n}$$

$$= \text{Softmax}_\beta(\underbrace{\begin{bmatrix} -(k_j^\top x_1 - L_0)^2 & \cdots & -(k_j^\top x_n - L_0)^2 \\ -(k_j^\top x_1 - L_1)^2 & \cdots & -(k_j^\top x_n - L_1)^2 \\ \vdots & & \vdots \\ -(k_j^\top x_1 - L_P)^2 & \cdots & -(k_j^\top x_n - L_P)^2 \end{bmatrix}}_{(P+1)\times n}), \tag{D.67}$$

where the last line follows from the column-wise nature of the $\text{Softmax}_\beta()$ function.

From (D.67), we have

$$(\text{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{r,c} = \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}},$$

for every $r \in \{0\} \cup [P]$ and $c \in [n]$.

Thus, for each column in (D.67), we have

$$(\text{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c} = \underbrace{\sum_{r=0}^P \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} e_{r+1}^{(P+1)}}_{(P+1)\times 1}. \tag{D.68}$$

Combining $\widehat{V}_j^K$ and $(\text{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c}$, we obtain

$$\begin{aligned}
&\widehat{V}_j^K \cdot (\text{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c} \\
&= \underbrace{e_j^{(3n)}}_{3n\times 1} \underbrace{[L_0 \quad L_1 \quad \cdots \quad L_{n-1} \quad L_n \quad \cdots \quad L_P]}_{1\times(P+1)} \underbrace{\sum_{r=0}^P \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} e_{r+1}^{(P+1)}}_{(P+1)\times 1} \\
&\hspace{20em} \text{(By (D.65) and (D.68))} \\
&= \underbrace{e_j^{(3n)}}_{3n\times 1} \sum_{r=0}^P \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} \underbrace{[L_0 \quad L_1 \quad \cdots \quad L_{n-1} \quad L_n \quad \cdots \quad L_P]}_{1\times(P+1)} \underbrace{e_{r+1}^{(P+1)}}_{(P+1)\times 1} \\
&\hspace{14em} \text{(By the distributivity of matrix multiplication)} \\
&= \underbrace{e_j^{(3n)}}_{3n\times 1} \underbrace{\sum_{r=0}^P \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r}_{\text{scalar}} \qquad (e_{r+1}^{(P+1)} \text{ selects } L_r \text{ for every } r \in \{0\} \cup [P]) \\
&= \sum_{r=0}^P \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r e_j^{(3n)},
\end{aligned} \tag{D.69}$$

for every $c \in [n]$.

Hence,

$$\widehat{V}_j^K \cdot (\text{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c},$$

is a weighted average of the vectors $L_r e_j^{(3n)}$, with weights depending on β and the distance between L_r and $k_j^\top x_c$.

We recall: $\widehat{V}_j^K \cdot (\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c}$ gives the c-th column of Attn_j^K. Therefore, each column of Attn_j^K stores a weighted sum as an approximator for each entry in $k_j^\top X$.

We show that (D.69) is close to $k_j^\top x_c$:

$$
\begin{aligned}
&\|\widehat{V}_j^K \cdot (\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c} - k_j^\top x_c \cdot e_j^{(3n)}\|_\infty \\
= & \|\underbrace{\sum_{r=0}^{P} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r}_{\text{scalar}} \cdot e_j^{(3n)} - \underbrace{k_j^\top x_c}_{\text{scalar}} \cdot e_j^{(3n)}\|_\infty && (\text{By (D.69)}) \\
= & \|(\sum_{r=0}^{P} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r - k_j^\top x_c) \cdot e_j^{(3n)}\|_\infty \\
= & |\sum_{r=0}^{P} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r - k_j^\top x_c| && (\text{We have one non-zero entry in } e_j^{(3n)}) \\
= & |\sum_{r=0}^{P} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} L_r - \frac{\sum_{r=0}^P e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} k_j^\top x_c| \\
& && (\text{By } (\textstyle\sum_{r=0}^P e^{-\beta(L_r - k_j^\top x_c)^2})/(\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}) = 1) \\
= & |\sum_{r=0}^{P} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} (L_r - k_j^\top x_c)| \\
= & |\sum_{r:|L_r - k_j^\top x_c| < \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} (L_r - k_j^\top x_c) + \sum_{r:|L_r - k_j^\top x_c| \geq \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} (L_r - k_j^\top x_c)| \\
& (\text{By dividing the } L_r \text{ into two groups: one within } \Delta L \text{ away from } k_j^\top x_c\text{, one at least } \Delta L \text{ away from } k_j^\top x_c) \\
\leq & |\sum_{r:|L_r - k_j^\top x_c| < \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} (L_r - k_j^\top x_c)| + |\sum_{r:|L_r - k_j^\top x_c| \geq \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} (L_r - k_j^\top x_c)| \\
& && (\text{By triangle inequality}) \\
\leq & \underbrace{\sum_{r:|L_r - k_j^\top x_c| < \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} |L_r - k_j^\top x_c|}_{:=(I)} + \underbrace{\sum_{r:|L_r - k_j^\top x_c| \geq \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} |L_r - k_j^\top x_c|}_{:=(II)},
\end{aligned}
\tag{D.70}
$$

and the last inequality holds due to the triangle inequality and the non-negativity of the exponential function.

For term (I) in (D.70), we have

$$
\begin{aligned}
& (I) \\
= & \sum_{r:|L_r - k_j^\top x_c| < \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} |L_r - k_j^\top x_c| && (\text{By the definition of term } (I) \text{ in (D.70)}) \\
< & \sum_{r:|L_r - k_j^\top x_c| < \Delta L} \frac{e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} \Delta L && (\text{In this group of } L_r,\ |L_r - k_j^\top x_c| < \Delta L) \\
\leq & \Delta L,
\end{aligned}
\tag{D.71}
$$

and the last inequality holds since

$$
\frac{\sum_{r:|L_r - k_j^\top x_c| < \Delta L} e^{-\beta(L_r - k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s - k_j^\top x_c)^2}} \leq 1. \qquad (\text{The numerator is part of the denominator})
$$

For term (II) in (D.70), we have

$$
\begin{aligned}
&(II)\\
= & \sum_{r:|L_r-k_j^\top x_c|\geq \Delta L} \frac{e^{-\beta(L_r-k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s-k_j^\top x_c)^2}}|L_r-k_j^\top x_c| \quad \text{(By the definition of term } (II) \text{ in (D.70))}\\
\leq & \sum_{r:|L_r-k_j^\top x_c|\geq \Delta L} \frac{e^{-\beta(L_r-k_j^\top x_c)^2}}{\sum_{s=0}^P e^{-\beta(L_s-k_j^\top x_c)^2}} 2dB^2\\
& \text{(By (D.51) and (D.52), we have } |L_r-k_j^\top x_c| \leq 2dB^2)\\
\leq & \sum_{r:|L_r-k_j^\top x_c|\geq \Delta L} \frac{e^{-\beta\Delta L^2}}{\sum_{s=0}^P e^{-\beta(L_s-k_j^\top x_c)^2}} 2dB^2\\
& \text{(By } |L_r-k_j^\top x_c|\geq \Delta L\text{, we have } e^{-\beta(L_r-k_j^\top x_c)^2}\leq e^{-\beta\Delta L^2})\\
\leq & \sum_{r:|L_r-k_j^\top x_c|\geq \Delta L} \frac{e^{-\beta\Delta L^2}}{\max_s\{e^{-\beta(L_s-k_j^\top x_c)^2}\}} 2dB^2\\
& \text{(We only keep the contribution from the nearest } L_s \text{ to } k_j^\top x_c)\\
\leq & \sum_{r:|L_r-k_j^\top x_c|\geq \Delta L} \frac{e^{-\beta\Delta L^2}}{e^{-\beta\frac{\Delta L^2}{4}}} 2dB^2,
\end{aligned} \tag{D.72}
$$

and the last inequality holds since, by our construction of L_s in (D.52), the distance from $k_j^\top x_c$ to the nearest L_s is at most $\frac{\Delta L}{2}$. That is,

$$
|L_{s_0}-k_j^\top x_c| \leq \frac{\Delta L}{2}, \quad \text{for} \quad s_0 = \mathop{\mathrm{argmin}}_s |L_s - k_j^\top x_c|.
$$

From (D.72), we have

$$
\sum_{|L_r-k_j^\top x_c|\geq \Delta L} e^{-\frac{3}{4}\beta\Delta L^2} 2dB^2 \leq P e^{-\frac{3}{4}\beta\Delta L^2} 2dB^2, \tag{D.73}
$$

and the last inequality holds since, by our construction of L_r in (D.52), at most P points satisfy $|L_r - k_j^\top x_c| \geq \Delta L$. This P-point scenario occurs when the value of $k_j^\top x_c$ equals one of the L_r grid points.

Combining (D.70), (D.71), and (D.73), we have:

$$
\|\widehat{V}_j^K \cdot (\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K)\cdot W_{\widehat{O}}^{K;j})_{:,c} - k_j^\top x_c \cdot e_j^{(3n)}\|_\infty \leq \underbrace{\Delta L}_{:=(a)} + \underbrace{P e^{-\frac{3}{4}\beta\Delta L^2} 2dB^2}_{:=(b)}. \tag{D.74}
$$

For term (a) in (D.74), we recall

$$
\Delta L = \frac{2dB^2}{P}. \quad \text{(By the definition of } \Delta L \text{. i.e., (D.53))}
$$

To bound ΔL, we choose

$$
P \geq \frac{4dB^2}{\epsilon_1},
$$

for any $\epsilon_1 > 0$, such that

$$
\Delta L \leq \frac{\epsilon_1}{2}.
$$

For term (b) in (D.74), we set

$$
\beta \geq \frac{4}{3}\frac{1}{(\Delta L)^2}\ln\left(\frac{4dB^2 P}{\epsilon_1}\right),
$$

such that

$$Pe^{-\frac{3}{4}\beta\Delta L^2}2dB^2 \leq \frac{\epsilon_1}{2}.$$

Thus, from (D.74), we have

$$\begin{aligned}
\|\widehat{V}_j^K \cdot (\mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j})_{:,c} - \underbrace{k_j^\top x_c}_{\text{scalar}} \cdot \underbrace{e_j^{(3n)}}_{3n\times 1}\|_\infty &\leq \Delta L + Pe^{-\frac{3}{4}\beta\Delta L^2}2dB^2 \\
&\leq \frac{\epsilon_1}{2} + \frac{\epsilon_1}{2} \\
&= \epsilon_1,
\end{aligned}$$

and this leads to

$$\|\widehat{V}_j^K \cdot \mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j} - \underbrace{e_j^{(3n)}}_{3n\times 1} \cdot \underbrace{k_j^\top X}_{1\times n}\|_\infty \leq \epsilon_1. \tag{D.75}$$

We recall

$$\begin{aligned}
&\widehat{V}_j^K \cdot \mathrm{Softmax}_\beta((\widehat{K}_j^K)^\top \widehat{Q}_j^K) \cdot W_{\widehat{O}}^{K;j} \\
&= W_{\widehat{V}}^{K;j}\mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) \cdot \mathrm{Softmax}_\beta((W_{\widehat{K}}^{K;j}\mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}))^\top W_{\widehat{Q}}^{K;j}\mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix})) \cdot W_{\widehat{O}}^{K;j} \\
&\qquad \text{(By the definition of } \widehat{K}_j^K, \widehat{Q}_j^K, \text{ and } \widehat{V}_j^K\text{)} \\
&= \mathrm{Attn}_j^K \circ \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}). \qquad \text{(By the definition of } \mathrm{Attn}_j^K\text{)}
\end{aligned}$$

Thus, we write (D.75) as

$$\|\mathrm{Attn}_j^K \circ \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \underbrace{e_j^{(3n)}}_{3n\times 1} \cdot \underbrace{k_j^\top X}_{1\times n}\|_\infty \leq \epsilon_1,$$

and we sum over the index j to obtain the approximation across rows

$$\|\sum_{j=1}^{n} \mathrm{Attn}_j^K \circ \mathrm{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} K \\ 0_{n\times n} \\ 0_{n\times n} \end{bmatrix}\|_\infty \leq \epsilon_1, \tag{D.76}$$

for any $\epsilon_1 > 0$.

In-Context Calculation of Q and V. We approximate Q and V using the same procedure as that of K.

We start with Q.

Again, we define Linear_j^Q preceding Attn_j^Q first. We construct Linear_j^Q similarly to Linear_j^K. The only difference is the position of the identity I_d in the first term. Explicitly,

$$\begin{aligned}
\mathrm{Linear}_j^Q(Z) := &\underbrace{\begin{bmatrix} 0_{d\times d} & 0_{d\times d} & 0_{d\times d} & 0_{d\times d} \\ 0_{d\times d} & 0_{d\times d} & I_d & 0_{d\times d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times d} & 0_{3\times d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} 2L_0 e_j^{(n)} & 2L_1 e_j^{(n)} & \cdots & 2L_P e_j^{(n)} \end{bmatrix}}_{n\times(P+1)} + \\
&\underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} I_n & 0_{n\times(P+1-n)} \end{bmatrix}}_{n\times(P+1)} + \underbrace{\begin{bmatrix} 0_{2d\times(P+1)} \\ M_1 \\ M_L \end{bmatrix}}_{(2d+3)\times(P+1)}.
\end{aligned}$$

Linear_j^Q takes $\begin{bmatrix} X^\top & W_K & W_Q & W_V \end{bmatrix}^\top$ as input and outputs:

$$\mathrm{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_d & \cdots & 0_d \\ 2L_0 q_j & 2L_1 q_j & \cdots & 2L_{n-1} q_j & 2L_n q_j & \cdots & 2L_P q_j \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & -L_n^2 & \cdots & -L_P^2 \end{bmatrix}.$$

Next, we construct $\mathrm{Attn}_j^Q : \mathbb{R}^{(2d+3)\times(P+1)} \to \mathbb{R}^{3n\times n}$ to be

$$\mathrm{Attn}_j^Q(D) \coloneqq \underbrace{W_{\widehat{V}}^{Q;j} D}_{3n\times(P+1)} \cdot \underbrace{\mathrm{Softmax}_\beta((W_{\widehat{K}}^{Q;j} D)^\top W_{\widehat{Q}}^{Q;j} D)}_{(P+1)\times(P+1)} \cdot \underbrace{W_{\widehat{O}}^{Q;j}}_{(P+1)\times n},$$

where $D \in \mathbb{R}^{(2d+3)\times(P+1)}$ denotes any input, and

$$W_{\widehat{K}}^{Q;j} \coloneqq W_{\widehat{K}}^{K;j} = \underbrace{\begin{bmatrix} 0_{d\times d} & I_d & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 0 & 0 & 1 \end{bmatrix}}_{(d+1)\times(2d+3)},$$

$$W_{\widehat{Q}}^{Q;j} \coloneqq W_{\widehat{Q}}^{K;j} = \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 1 & 0 & 0 \end{bmatrix}}_{(d+1)\times(2d+3)},$$

$$W_{\widehat{V}}^{Q;j} \coloneqq \underbrace{e_{n+j}^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} 0_{1\times d} & 0_{1\times d} & 0 & 1 & 0 \end{bmatrix}}_{1\times(2d+3)},$$

$$W_{\widehat{O}}^{Q;j} \coloneqq W_{\widehat{O}}^{K;j} = \underbrace{\begin{bmatrix} I_n \\ 0_{(P+1-n)\times n} \end{bmatrix}}_{(P+1)\times n}.$$

We define the $\widehat{K}_j^Q$ of Attn_j^Q to be

$$\widehat{K}_j^Q \coloneqq W_{\widehat{K}}^{Q;j} \cdot \mathrm{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{\begin{bmatrix} 2L_0 q_j & 2L_1 q_j & \cdots & 2L_{n-1} q_j & \cdots & 2L_P q_j \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}}_{(d+1)\times(P+1)}. \tag{D.77}$$

We define the $\widehat{Q}_j^Q$ of Attn_j^Q to be

$$\widehat{Q}_j^Q \coloneqq W_{\widehat{Q}}^{Q;j} \cdot \mathrm{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times(P+1-n)} \\ 1 & 1 & \cdots & 1 & 0_{1\times(P+1-n)} \end{bmatrix}}_{(d+1)\times(P+1)}. \tag{D.78}$$

We define the $\widehat{V}_j^Q$ of Attn_j^Q to be

$$\widehat{V}_j^Q \coloneqq W_{\widehat{V}}^{Q;j} \cdot \mathrm{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{e_{n+j}^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \end{bmatrix}}_{1\times(P+1)}. \tag{D.79}$$

Then, by going through the same calculations as those of K, we have

$$\| \sum_{j=1}^n \mathrm{Attn}_j^Q \circ \mathrm{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} 0_{n\times n} \\ Q \\ 0_{n\times n} \end{bmatrix} \|_\infty \le \epsilon_1. \tag{D.80}$$

To approximate V, we define

$$\mathrm{Linear}_j^V(Z) := \underbrace{\begin{bmatrix} 0_{d\times d} & 0_{d\times d} & 0_{d\times d} & 0_{d\times d} \\ 0_{d\times d} & 0_{d\times d} & 0_{d\times d} & I_d \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times d} & 0_{3\times d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} 2L_0 e_j^{(n)} & 2L_1 e_j^{(n)} & \cdots & 2L_P e_j^{(n)} \end{bmatrix}}_{n\times(P+1)} +$$
$$\underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 2d} \\ 0_{d\times d} & 0_{d\times d} & 0_{d\times 2d} \\ 0_{3\times d} & 0_{3\times d} & 0_{3\times 2d} \end{bmatrix}}_{(2d+3)\times 4d} \underbrace{Z}_{4d\times n} \underbrace{\begin{bmatrix} I_n & 0_{n\times(P+1-n)} \end{bmatrix}}_{n\times(P+1)} + \underbrace{\begin{bmatrix} 0_{2d\times(P+1)} \\ M_1 \\ M_L \end{bmatrix}}_{(2d+3)\times(P+1)}.$$

Linear_j^V outputs in a similar manner as Linear_j^K:

$$\mathrm{Linear}_j^V(Z) = \begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_d & \cdots & 0_d \\ 2L_0 v_j & 2L_1 v_j & \cdots & 2L_{n-1} v_j & 2L_n v_j & \cdots & 2L_P v_j \\ 1 & 1 & \cdots & 1 & 0 & \cdots & 0 \\ L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & -L_n^2 & \cdots & -L_P^2 \end{bmatrix}.$$

Next, we construct $\mathrm{Attn}_j^V : \mathbb{R}^{(2d+3)\times(P+1)} \to \mathbb{R}^{3n\times n}$ to be

$$\mathrm{Attn}_j^V := \underbrace{W_{\widehat{V}}^{V;j} D}_{3n\times(P+1)} \cdot \underbrace{\mathrm{Softmax}_\beta((W_{\widehat{K}}^{V;j} D)^\top W_{\widehat{Q}}^{V;j} D)}_{(P+1)\times(P+1)} \underbrace{W_{\widehat{O}}^{V;j}}_{(P+1)\times n},$$

where $D \in \mathbb{R}^{(2d+3)\times(P+1)}$ denotes any input, and

$$W_{\widehat{K}}^{V;j} := W_{\widehat{K}}^{K;j} = \underbrace{\begin{bmatrix} 0_{d\times d} & I_d & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 0 & 0 & 1 \end{bmatrix}}_{(d+1)\times(2d+3)},$$

$$W_{\widehat{Q}}^{V;j} := W_{\widehat{Q}}^{K;j} = \underbrace{\begin{bmatrix} I_d & 0_{d\times d} & 0_{d\times 1} & 0_{d\times 1} & 0_{d\times 1} \\ 0_{1\times d} & 0_{1\times d} & 1 & 0 & 0 \end{bmatrix}}_{(d+1)\times(2d+3)},$$

$$W_{\widehat{V}}^{V;j} := \underbrace{e_{2n+j}^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} 0_{1\times d} & 0_{1\times d} & 0 & 1 & 0 \end{bmatrix}}_{1\times(2d+3)},$$

$$W_{\widehat{O}}^{V;j} := W_{\widehat{O}}^{K;j} = \underbrace{\begin{bmatrix} I_n \\ 0_{(P+1-n)\times n} \end{bmatrix}}_{(P+1)\times n}.$$

We define the $\widehat{K}_j^V$ to be

$$\widehat{K}_j^V := W_{\widehat{K}}^{V;j} \cdot \mathrm{Linear}_j^V(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{\begin{bmatrix} 2L_0 v_j & 2L_1 v_j & \cdots & 2L_{n-1} v_j & \cdots & 2L_P v_j \\ -L_0^2 & -L_1^2 & \cdots & -L_{n-1}^2 & \cdots & -L_P^2 \end{bmatrix}}_{(d+1)\times(P+1)}. \quad \text{(D.81)}$$

We define the $\widehat{Q}_j^V$ to be

$$\widehat{Q}_j^V := W_{\widehat{Q}}^{V;j} \cdot \mathrm{Linear}_j^V(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{\begin{bmatrix} x_1 & x_2 & \cdots & x_n & 0_{d\times(P+1-n)} \\ 1 & 1 & \cdots & 1 & 0_{1\times(P+1-n)} \end{bmatrix}}_{(d+1)\times(P+1)}. \quad \text{(D.82)}$$

We define the $\widehat{V}_j^V$ to be

$$\widehat{V}_j^V := W_{\widehat{V}}^{V;j} \cdot \text{Linear}_j^V(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) = \underbrace{e_{2n+j}^{(3n)}}_{3n\times 1} \underbrace{\begin{bmatrix} L_0 & L_1 & \cdots & L_{n-1} & L_n & \cdots & L_P \end{bmatrix}}_{1\times(P+1)}. \quad \text{(D.83)}$$

Similarly, by going through the same calculations as those of K, we have

$$\| \sum_{j=1}^{n} \text{Attn}_j^V \circ \text{Linear}_j^V(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} 0_{n\times n} \\ 0_{n\times n} \\ V \end{bmatrix} \|_\infty \leq \epsilon_1. \quad \text{(D.84)}$$

Then, by combining (D.76), (D.80), and (D.84), we have

$$\| \sum_{j=1}^{n} \text{Attn}_j^K \circ \text{Linear}_j^K(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} K \\ 0_{n\times n} \\ 0_{n\times n} \end{bmatrix} + \sum_{j=1}^{n} \text{Attn}_j^Q \circ \text{Linear}_j^Q(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} 0_{n\times n} \\ Q \\ 0_{n\times n} \end{bmatrix} +$$

$$\sum_{j=1}^{n} \text{Attn}_j^V \circ \text{Linear}_j^V(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \begin{bmatrix} 0_{n\times n} \\ 0_{n\times n} \\ V \end{bmatrix} \|_\infty \leq \epsilon_1.$$

As previously stated in (D.54), $\text{Linear}_j^K, \text{Linear}_j^Q$ and Linear_j^V denote $\text{Linear}_j, \text{Linear}_{n+j}$ and Linear_{2n+j} respectively. Also, as in (D.55), $\text{Attn}_j^K, \text{Attn}_j^Q$ and Attn_j^V denote $\text{Attn}_j, \text{Attn}_{n+j}$ and Attn_{2n+j}.

Thus, we have

$$\| \sum_{j=1}^{3n} \text{Attn}_j \circ \text{Linear}_j(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}) - \underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3n\times n} \|_\infty \leq \epsilon_1. \quad \text{(D.85)}$$

We define

$$\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix} := \sum_{j=1}^{3n} \text{Attn}_j \circ \text{Linear}_j(\begin{bmatrix} X \\ W_K^\top \\ W_Q^\top \\ W_V^\top \end{bmatrix}),$$

such that (D.85) becomes

$$\| \begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix} - \begin{bmatrix} K \\ Q \\ V \end{bmatrix} \|_\infty \leq \epsilon_1. \quad \text{(D.86)}$$

Step 3: Single-Head Assembly of the Attention Output. Our goal in this part is to reconstruct the attention mechanism

$$V'\text{Softmax}_\beta((K')^\top Q'), \quad \text{and} \quad V\text{Softmax}_\beta((K)^\top Q),$$

from K', Q', V' and K, Q, V via Attn_s.

To achieve the reconstruction of attention mechanisms, we build Attn_s as

$$\text{Attn}_s(Z) := [0_{n\times 2n} \quad I_n]\, Z \cdot \text{Softmax}_\beta(([I_n \quad 0_{n\times 2n}]\, Z)^\top [0_{n\times n} \quad I_n \quad 0_{n\times n}]\, Z),$$

where $Z \in \mathbb{R}^{3n\times n}$ denotes any input.

Then, we have

$$\overbrace{\mathrm{Attn}_s(\underbrace{\begin{bmatrix}K\\Q\\V\end{bmatrix}}_{3n\times n})}^{n\times n} = \underbrace{V}_{n\times n}\mathrm{Softmax}_\beta(\underbrace{(K)^\top Q}_{n\times n}),$$

and

$$\overbrace{\mathrm{Attn}_s(\underbrace{\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}}_{3n\times n})}^{n\times n} = \underbrace{V'}_{n\times n}\mathrm{Softmax}_\beta(\underbrace{(K')^\top Q'}_{n\times n}).$$

Step 4: Error Bound From the results of **Step 3**, we have

$$\begin{aligned}&\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\\ &= V'\mathrm{Softmax}_\beta(K'^\top Q') - V\mathrm{Softmax}_\beta(K^\top Q)\\ &= V'\mathrm{Softmax}_\beta(K'^\top Q') - V\mathrm{Softmax}_\beta(K'^\top Q') + V\mathrm{Softmax}_\beta(K'^\top Q') - V\mathrm{Softmax}_\beta(K^\top Q)\\ &= (V'-V)\mathrm{Softmax}_\beta(K'^\top Q') + V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q)).\end{aligned}$$

Thus, we have

$$\begin{aligned}&\|\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\|_\infty\\ &= \|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q') + V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty\\ &\leq \underbrace{\|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q')\|_\infty}_{:=(A)} + \underbrace{\|V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty}_{:=(B)},\end{aligned} \tag{D.87}$$

and the last inequality follows from the triangle inequality.

For term (A) in (D.87), since each column in $\mathrm{Softmax}_\beta(K'^\top Q')$ sums up to 1, then for each column of (A),

$$\underbrace{(V'-V)}_{n\times n}\underbrace{\mathrm{Softmax}_\beta(K'^\top Q')_{:,j}}_{n\times 1},$$

is a weighted sum of the columns from $(V'-V)$.

Then, we have

$$\|(V'-V)\mathrm{Softmax}_\beta(K'^\top Q')_{:,j}\|_\infty \leq \|V'-V\|_\infty \leq \epsilon_1,$$

and the first inequality holds since the column average of $(V'-V)$ has a maximum entry no greater than the maximum entry among the original columns in $(V'-V)$. The second inequality holds since (D.86). This conclusion holds for every column in term (A), so we obtain

$$(A) \leq \epsilon_1. \tag{D.88}$$

Term (B) in (D.87) is

$$(B) = \|V(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))\|_\infty.$$

For the simplicity of presentation, we define

$$\Delta S := \mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q),$$

such that for each entry in (B), we have

$$
\begin{aligned}
|(V\Delta S)_{ij}| &= |\sum_{k=1}^{n} V_{ik}(\Delta S)_{kj}| && \text{(By the definition of matrix multiplication)}\\
&\le \sum_{k=1}^{n} |V_{ik}| \cdot |(\Delta S)_{kj}| && \text{(By triangle inequality and } |ab| = |a|\cdot|b| \text{ for all } a, b \in \mathbb{R})\\
&\le \sum_{k=1}^{n} \|V\|_\infty \cdot \|\Delta S\|_\infty && \text{(By } |V_{ik}| \le \|V\|_\infty \text{ and } |(\Delta S)_{kj}| \le \|\Delta S\|_\infty)\\
&= n\|V\|_\infty \cdot \|\Delta S\|_\infty,
\end{aligned}
$$

and this leads to

$$
(B) \le n\|V\|_\infty \cdot \|\Delta S\|_\infty. \tag{D.89}
$$

For each entry in ΔS, we have

$$
\begin{aligned}
&|(\Delta S)_{i,j}|\\
=& |(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))_{i,j}|\\
=& |\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}| \quad (K_i', Q_i', K_i, Q_i \text{ denote the } i\text{-th column in } K', Q', K, Q)\\
=& |\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} + \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}|\\
\le& |\frac{e^{\beta K_i'\cdot Q_j'} - e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}| + |e^{\beta K_i\cdot Q_j}(\frac{1}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - \frac{1}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}})| \quad \text{(By triangle inequality)}\\
=& \frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}|1 - e^{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')}| + \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}|\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} - 1|\\
& \text{(By non-negativity of exponential)}\\
<& \underbrace{|1 - e^{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')}|}_{:=(B\text{-}1)} + \underbrace{|1 - \frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}|}_{:=(B\text{-}2)},
\end{aligned} \tag{D.90}
$$

and the last inequality holds since

$$
\frac{e^{\beta K_i'\cdot Q_j'}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}} < 1, \quad \frac{e^{\beta K_i\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}} < 1.
$$

To bound term $(B\text{-}1)$ in (D.90), we recall

$$
\| \underbrace{\begin{bmatrix} K' \\ Q' \\ V' \end{bmatrix}}_{3n\times n} - \underbrace{\begin{bmatrix} K \\ Q \\ V \end{bmatrix}}_{3n\times n} \|_\infty \le \epsilon_1,
$$

so

$$
\begin{aligned}
\| \underbrace{K' - K}_{n\times n} \|_\infty \le \epsilon_1,\\
\| \underbrace{Q' - Q}_{n\times n} \|_\infty \le \epsilon_1.
\end{aligned}
$$

Let K_i', Q_i', K_i, Q_i denote the i-th column in K', Q', K, Q, then we have

$$
\begin{aligned}
\underbrace{\Delta K_i}_{n\times 1} := \underbrace{K_i' - K_i}_{n\times 1}, \quad \|\Delta K_i\|_\infty \le \epsilon_1,\\
\underbrace{\Delta Q_i}_{n\times 1} := \underbrace{Q_i' - Q_i}_{n\times 1}, \quad \|\Delta Q_i\|_\infty \le \epsilon_1.
\end{aligned}
$$

Thus, for term $(B\text{-}1)$ in (D.90), we have

$$
\begin{aligned}
&(B\text{-}1)\\
&= |1-\exp\{\beta(K_i\cdot Q_j - K_i'\cdot Q_j')\}|\\
&= |1-\exp\{\beta(K_i\cdot Q_j-(K_i+\Delta K_i)\cdot(Q_j+\Delta Q_j))\}|\\
&\qquad\qquad (\text{By } K_i'=K_i+\Delta K_i \text{ and } Q_i'=Q_i+\Delta Q_i)\\
&= |1-\exp\{-\beta(K_i\cdot\Delta Q_j+Q_j\cdot\Delta K_i+\Delta K_i\cdot\Delta Q_j)\}|,\\
&\qquad (\text{By } K_i\cdot Q_j-(K_i+\Delta K_i)\cdot(Q_i+\Delta Q_i)=-(K_i\cdot\Delta Q_j+Q_j\cdot\Delta K_i+\Delta K_i\cdot\Delta Q_j))
\end{aligned}
$$

and we know

$$
\begin{aligned}
&K_i\cdot\Delta Q_j+Q_j\cdot\Delta K_i+\Delta K_i\cdot\Delta Q_j\\
&\le n\cdot\|K_i\|_\infty\|\Delta Q_j\|_\infty+n\cdot\|Q_j\|_\infty\|\Delta K_i\|_\infty+n\cdot\|\Delta K_i\|_\infty\|\Delta Q_j\|_\infty\\
&\qquad (\text{By } a\cdot b\le n\|a\|_\infty\|b\|_\infty \text{ for all } a,b\in\mathbb{R}^n)\\
&\le 2nB_{KQV}\epsilon_1+n\epsilon_1^2. \qquad (\text{By } \|K_i\|_\infty,\|Q_j\|_\infty\le B_{KQV} \text{ and } \|\Delta K_i\|_\infty,\|\Delta Q_j\|_\infty\le\epsilon_1)
\end{aligned}
$$

Thus, we have

$$
(B\text{-}1)\le|1-e^{-\beta n(2B_{KQV}\epsilon_1+\epsilon_1^2)}|. \tag{D.91}
$$

For term $(B\text{-}2)$ in (D.90), we have

$$
\begin{aligned}
&(B\text{-}2)\\
&= |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta K_{i'}'\cdot Q_j'}}| \qquad (\text{By the definition of } (B\text{-}2))\\
&= |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta(K_{i'}+\Delta K_{i'})\cdot(Q_j+\Delta Q_j)}}| \qquad (\text{By } K_{i'}'=K_{i'}+\Delta K_{i'} \text{ and } Q_i'=Q_i+\Delta Q_i)\\
&= |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta(K_{i'}\cdot Q_j+K_{i'}\cdot\Delta Q_j+Q_j\cdot\Delta K_{i'}+\Delta K_{i'}\cdot\Delta Q_j)}}|,
\end{aligned}
$$

and for all i' in the denominator, we have

$$
\begin{aligned}
&K_{i'}\cdot Q_j+K_{i'}\cdot\Delta Q_j+Q_j\cdot\Delta K_{i'}+\Delta K_{i'}\cdot\Delta Q_j\\
&\le K_{i'}\cdot Q_j+n\cdot\|K_{i'}\|_\infty\|\Delta Q_j\|_\infty+n\cdot\|Q_j\|_\infty\|\Delta K_{i'}\|_\infty+n\cdot\|\Delta K_{i'}\|_\infty\|\Delta Q_j\|_\infty\\
&\qquad (\text{By } a\cdot b\le n\|a\|_\infty\|b\|_\infty \text{ for all } a,b\in\mathbb{R}^n)\\
&\le K_{i'}\cdot Q_j+2nB_{KQV}\epsilon_1+n\epsilon_1^2. \qquad (\text{By } \|K_{i'}\|_\infty,\|Q_j\|_\infty\le B_{KQV} \text{ and } \|\Delta K_{i'}\|_\infty,\|\Delta Q_j\|_\infty\le\epsilon_1)
\end{aligned}
$$

Thus,

$$
\begin{aligned}
&(B\text{-}2)\\
&= |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta(K_{i'}\cdot Q_j+K_{i'}\cdot\Delta Q_j+Q_j\cdot\Delta K_{i'}+\Delta K_{i'}\cdot\Delta Q_j)}}|\\
&\le |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{\sum_{i'=1}^n e^{\beta(K_{i'}\cdot Q_j+2nB_{KQV}\epsilon_1+n\epsilon_1^2)}}|\\
&= |1-\frac{\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}{e^{\beta n(2B_{KQV}\epsilon_1+\epsilon_1^2)}\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}}| \qquad (e^{\beta n(2B_{KQV}\epsilon_1+\epsilon_1^2)} \text{ is independent of } i')\\
&= |1-e^{-\beta n(2B_{KQV}\epsilon_1+\epsilon_1^2)}|,
\end{aligned} \tag{D.92}
$$

and the last equality holds since the common factor $\sum_{i'=1}^n e^{\beta K_{i'}\cdot Q_j}$ cancels out.

Combining (D.90), (D.91), and (D.92), we have

$$
\begin{aligned}
&|(\mathrm{Softmax}_\beta(K'^\top Q')-\mathrm{Softmax}_\beta(K^\top Q))_{i,j}|\\
&< 2|1-e^{-\beta n(2B_{KQV}\epsilon_1+\epsilon_1^2)}|\\
&\le 2|1-e^{-\beta n(2B_{KQV}\epsilon_1+\epsilon_1)}|. \qquad (\text{By requiring } 0<\epsilon_1\le 1)
\end{aligned}
$$

Thus, for any $0 < \epsilon_0 < 2$, when ϵ_1 satisfies

$$0 < \epsilon_1 \le \min\{1, \frac{-\ln\big(1-\frac{\epsilon_0}{2}\big)}{\beta n(2B_{KQV}+1)}\},$$

we have

$$|(\mathrm{Softmax}_\beta(K'^\top Q') - \mathrm{Softmax}_\beta(K^\top Q))_{i,j}| < \epsilon_0. \tag{D.93}$$

From (D.89) and (D.93), we have

$$(B) \le n\|V\|_\infty \|\Delta S\|_\infty < nB_{KQV}\epsilon_0, \tag{D.94}$$

since $\|V\|_\infty \le B_{KQV}$ and $\|\Delta S\|_\infty < \epsilon_0$.

Combining (D.87), (D.88) and (D.94) yields

$$\|\mathrm{Attn}_s(\begin{bmatrix}K'\\Q'\\V'\end{bmatrix}) - \mathrm{Attn}_s(\begin{bmatrix}K\\Q\\V\end{bmatrix})\|_\infty < \epsilon_1 + nB_{KQV}\epsilon_0.$$

When we take ϵ_0 and ϵ_1 to be infinitely small, the right-hand side tends to 0.

This completes the proof. □

D.7 Proof of Corollary 4.2.1

Theorem D.7 (Restate of Corollary 4.2.1: In-Context Emulation of Statistical Methods)**.** Let $\mathcal{A}$ denote the set of all the in-context algorithms that a single-layer attention is able to approximate. For an $a \in \mathcal{A}$ (that is, a specific algorithm), let W_K^a, W_Q^a, W_V^a denote the weights of the attention that implements this algorithm. For any $\epsilon > 0$ and any finite set $\mathcal{A}_0 \in \mathcal{A}$, there exists a 2-layer attention $\mathrm{Attn} \circ \mathrm{Attn}_m$ such that

$$\|\sum_{j=1}^{3n} \mathrm{Attn}_s \circ \mathrm{Attn}_j \circ \mathrm{Linear}_j\left(\begin{bmatrix}X\\W^a\end{bmatrix}\right) - a(X)\|_\infty \le \epsilon, \quad a \in \mathcal{A}_0,$$

where W^a is the W defined as Definition 4.2 using W_K^a, W_Q^a, W_V^a.

Proof. Without loss of generality, assume all W_K^a, W_Q^a, W_V^a to be of the same hidden dimension since we are always able to pad them to the same size. According to Theorem 4.2, there exists a network $\sum_{j=1}^n \mathrm{Attn}_s \circ \mathrm{Attn}_j \circ \mathrm{Linear}_j$ that approximate $a(X)$ with an error no larger than $\epsilon > 0$ when given input of the form:

$$\begin{bmatrix}X\\W_K^{a\top}\\W_Q^{a\top}\\W_V^{a\top}\end{bmatrix}.$$

Then for a set of $a \in \mathcal{A}_0$, define $P_m := \max_{a\in\mathcal{A}_0} P_\epsilon^{(a)}$.

By Theorem 4.2, there exists a network consisting of a self-attention followed by a multi-head attention with a linear layer and parameter P equals to P_m, such that for any $a \in \mathcal{A}_0$, we have

$$\|\sum_{j=1}^{3n} \mathrm{Attn}_s \circ \mathrm{Attn}_j \circ \mathrm{Linear}_j\left(\begin{bmatrix}X\\W_K^{a\top}\\W_Q^{a\top}\\W_V^{a\top}\end{bmatrix}\right) - a(X)\|_\infty \le \epsilon, \quad a \in \mathcal{A}_0.$$

This completes the proof. □

E In-Context Application of Statistical Methods by Modern Hopfield Network

Definition E.1 (Modern Hopfield Network)**.** Define $Y = (y_1, \cdots, y_N)^\top \in \mathbb{R}^{d_y \times N}$ as the raw stored pattern, $R = (r_1, \cdots, r_S)^\top \in \mathbb{R}^{R_r \times S}$ as the raw state pattern, and $W_Q \in \mathbb{R}^{d \times d_r}$, $W_K \in \mathbb{R}^{d \times d_y}$, $W_V \in \mathbb{R}^{d_v \times d}$ as the projection matrices. A Hopfield layer Hopfield is defined as:

$$\text{Hopfield}(R; Y, W_Q, W_K, W_V) := \underbrace{W_V}_{d_v \times d} \overbrace{W_K Y}^{d \times N} \text{Softmax}(\underbrace{\beta (W_K Y)^\top W_Q R}_{N \times S}) \in \mathbb{R}^{d_v \times S}, \tag{E.1}$$

where β is a temperature parameter.
With $K \in \mathbb{R}^{d \times N}$ denoting $W_K Y$, $Q \in \mathbb{R}^{d \times S}$ denoting $W_Q R$ and $V \in \mathbb{R}^{d_v \times N}$ denoting $W_V W_K Y$, (E.1) writes out as:

$$\text{Hopfield}(R; Y, W_Q, W_K, W_V) := V \text{Softmax}(\beta \cdot K^\top Q) \in \mathbb{R}^{d_v \times S}.$$

Theorem E.1. Let $Z = [z_1, z_2, \cdots, z_n] \in \mathbb{R}^{d \times n}$ denote the input from a compact input domain. For any linear transformation $l(z) = a^\top z + b : \mathbb{R}^d \to \mathbb{R}$, and any continuous function $f : \mathbb{R} \to \mathbb{R}^o$ where o is the output dimension, there exists a Hopfield network Hopfield such that

$$\|\text{Hopfield}(Z) - [f(l(z_1)) \quad f(l(z_2)) \quad \cdots \quad f(l(z_n))]\|_\infty \leq \epsilon,$$

for any $\epsilon > 0$.

Proof. We first perform a simple token-wise linear transformation on the input:

$$\text{Linear}(Z) := \begin{bmatrix} I_{d \times d} \\ 0_{1 \times d} \end{bmatrix} Z + \begin{bmatrix} 0_{d \times n} \\ 1_{1 \times n} \end{bmatrix} = \begin{bmatrix} Z \\ 1_{1 \times n} \end{bmatrix} \in \mathbb{R}^{(d+1) \times n}.$$

We then construct W_Q to be:

$$W_Q := I_{(d+1)},$$

which is an identity matrix of dimension $\mathbb{R}^{(d+1) \times (d+1)}$.

This yields that

$$Q := W_Q \text{Linear}(Z) = \begin{bmatrix} Z \\ 1_{1 \times n} \end{bmatrix} \in \mathbb{R}^{(d+1) \times n}.$$

Following the definition of *Interpolations* in Appendix D.6, K, V are constructed as (here we omit Y since it's not the input):

$$K := \begin{bmatrix} 2L_0 a & 2L_1 a & \cdots & 2L_P a \\ 2L_0 b - L_0^2 & 2L_1 b - L_1^2 & \cdots & 2L_P b - L_P^2 \end{bmatrix},$$
$$V := [f(L_0) \quad f(L_1) \quad \cdots \quad f(L_P)].$$

By Definition E.1, we have

$$\text{Hopfield}(Z)$$
$$= [f(L_0) \quad f(L_1) \quad \cdots \quad f(L_P)] \text{Softmax}\left(\beta \begin{bmatrix} 2l(z_1)L_0 - L_0^2 & 2l(z_2)L_0 - L_0^2 & \cdots & 2l(z_n)L_0 - L_0^2 \\ 2l(z_1)L_1 - L_1^2 & 2l(z_2)L_1 - L_1^2 & \cdots & 2l(z_n)L_1 - L_1^2 \\ \vdots & \vdots & & \vdots \\ 2l(z_1)L_P - L_P^2 & 2l(z_2)L_P - L_P^2 & \cdots & 2l(z_n)L_P - L_P^2 \end{bmatrix}\right).$$

This is equivalent to:

$$\text{Hopfield}(Z)$$

$$= [f(L_0) \quad f(L_1) \quad \cdots \quad f(L_P)]\, \mathrm{Softmax}\left(-\beta \begin{bmatrix} (l(z_1)-L_0)^2 & (l(z_2)-L_0)^2 & \cdots & (l(z_n)-L_0)^2 \\ (l(z_1)-L_1)^2 & (l(z_2)-L_1)^2 & \cdots & (l(z_n)-L_1)^2 \\ \vdots & \vdots & & \vdots \\ (l(z_1)-L_P)^2 & (l(z_2)-L_P)^2 & \cdots & (l(z_n)-L_P)^2 \end{bmatrix} \right).$$

For any column $c \in [n]$ in $\mathrm{Hopfield}(Z)$, we have

$$\begin{aligned} \mathrm{Hopfield}(Z)_{:,c} &= [f(L_0) \quad f(L_1) \quad \cdots \quad f(L_P)]\, \mathrm{Softmax}(-\beta \begin{bmatrix} (l(z_c)-L_0)^2 \\ (l(z_c)-L_1)^2 \\ \vdots \\ (l(z_c)-L_P)^2 \end{bmatrix}) \\ &= \sum_{r=1}^{P} \frac{e^{-\beta(l(z_c)-L_r)^2}}{\sum_{r'=1}^{P} e^{-\beta(l(z_c)-L_{r'})^2}} f(L_r). \end{aligned}$$

When β is large enough, we have

$$\sum_{(l(z_c)-L_r)^2 \geq \Delta L} \frac{e^{-\beta(l(z_c)-L_r)^2}}{\sum_{r'=1}^{P} e^{-\beta(l(z_c)-L_{r'})^2}} \leq \sum_{(l(z_i)-L_r)^2 \geq \Delta L} \frac{e^{-\beta \Delta L}}{e^{-\beta \frac{\Delta L}{2}}} \leq P e^{-\frac{\beta \Delta L}{2}} \leq \epsilon_1,$$

for any $\epsilon_1 > 0$.

This means that the proportion of the $f(L_r)$ in $\mathrm{Hopfield}(Z)_{:,c}$ that deviates from $l(z_c)$ is no larger than ϵ_1.

Since f and l are continuous, and Z comes from a compact domain, $l(z_i)$ comes from a compact domain for all $i \in [n]$. Thus f is uniformly continuous on its input domain. This means that for any $\epsilon_2 > 0$, there exists a $\delta > 0$ such that when$(x-y)^2 \leq \delta$, $\|f(x)-f(y)\|_\infty \leq \epsilon_2$.

Configuring $\Delta L \leq \delta$ yields:

$$\begin{aligned} &\|\mathrm{Hopfield}(Z)_{:,c} - f(l(z_c))\|_\infty \\ &\leq \sum_{r=1}^{P} \frac{e^{-\beta(l(z_c)-L_r)^2}}{\sum_{r'=1}^{P} e^{-\beta(l(z_c)-L_{r'})^2}} \|f(L_r) - f(l(z_c))\|_\infty \\ &= \sum_{(l(z_c)-L_r)^2 \geq \Delta L} \frac{e^{-\beta(l(z_c)-L_r)^2}}{\sum_{r'=1}^{P} e^{-\beta(l(z_c)-L_{r'})^2}} \|f(L_r) - f(l(z_c))\|_\infty \\ &\quad + \sum_{(l(z_c)-L_r)^2 \leq \Delta L} \frac{e^{-\beta(l(z_c)-L_r)^2}}{\sum_{r'=1}^{P} e^{-\beta(l(z_c)-L_{r'})^2}} \|f(L_r) - f(l(z_c))\|_\infty \\ &\leq \epsilon_1 \cdot 2B + (1-\epsilon_1)\epsilon_2, \end{aligned}$$

where $B := \|f\|_{L_\infty}$ is the bound of f in infinite norm.

We set $\epsilon_2 \leq \epsilon/2$, $\epsilon_1 \leq \epsilon/(4B)$. This yields:

$$\begin{aligned} \|\mathrm{Hopfield}(Z)_{:,c} - f(l(z_c))\|_\infty &\leq \epsilon_1 \cdot 2B + (1-\epsilon_1)\epsilon_2 \\ &\leq \frac{\epsilon}{4B} \cdot 2B + 1 \cdot \frac{\epsilon}{2} = \epsilon. \end{aligned}$$

This completes the proof. □

Theorem E.2. Define

$$X := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{(d+1)\times n} \quad \text{and} \quad W := [w \quad w \quad \cdots \quad w] \in \mathbb{R}^{d \times n},$$

where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$ are the input-output pairs. $w \in \mathbb{R}^d$ is the linear coefficient to optimize. Suppose x_i, y_i and w are bounded by B in infinite norm.

For any continuous function $f:\mathbb{R}\to\mathbb{R}$, there exists a Hopfield layer Hopfield with linear connections such that

$$\|\mathrm{Hopfield}(W;X)-\begin{bmatrix}f(w^\top x_1-y_1)x_1 & f(w^\top x_2-y_2)x_2 & \cdots & f(w^\top x_n-y_n)x_n\end{bmatrix}\|_\infty\le\epsilon,$$

for any $\epsilon>0$.

Proof. Before plugging input W to the Hopfield layer, we pass it through a linear transformation Linear_w:

$$\mathrm{Linear}_w(W):=\begin{bmatrix}I_d\\0_{(d+n+2)\times d}\end{bmatrix}W+\begin{bmatrix}0_{d\times n}\\-1_{1\times n}\\0_{d\times n}\\-1_{1\times n}\\I_n\end{bmatrix}=\begin{bmatrix}W\\-1_{1\times n}\\0_{d\times n}\\-1_{1\times n}\\I_n\end{bmatrix}\in\mathbb{R}^{(2d+n+2)\times n}.$$

We also pass X through a linear transformation Linear_x:

$$\begin{aligned}&\mathrm{Linear}_x(X)\\
:=&\sum_{i=1}^n\underbrace{\begin{bmatrix}I_{d+1}\\0_{(d+1+n)\times(d+1)}\end{bmatrix}}_{(2d+n+2)\times(d+1)}\underbrace{X}_{(d+1)\times n}\underbrace{\begin{bmatrix}0_{n\times(i-1)(P+1)} & 2L_0e_i^{(n)} & 2L_1e_i^{(n)} & \cdots & 2L_Pe_i^{(n)} & 0_{n\times(n-i)(P+1)}\end{bmatrix}}_{n\times n(P+1)}\\
&+\sum_{i=1}^n\underbrace{\begin{bmatrix}0_{(d+1)\times d} & 0_{(d+1)}\\ I_d & 0_d\\ 0_{(n+1)\times d} & 0_{n+1}\end{bmatrix}}_{(2d+n+2)\times(d+1)}X\begin{bmatrix}0_{n\times(i-1)(P+1)} & f(L_0)e_i^{(i)} & f(L_1)e_i^{(i)} & \cdots & f(L_P)e_i^{(i)} & 0_{n\times(n-i)(P+1)}\end{bmatrix}\\
&+\underbrace{\begin{bmatrix}0_{(2d+1)\times(P+1)} & \cdots & 0_{(2d+1)\times(P+1)}\\ S & \cdots & S\\ (2dB^2+B-\ln\epsilon_0)e_1^{(n)}1_{1\times(P+1)} & \cdots & (2dB^2+B-\ln\epsilon_0)e_n^{(n)}1_{1\times(P+1)}\end{bmatrix}}_{(2d+n+2)\times n(P+1)}\\
=&\begin{bmatrix}T_1 & T_2 & \cdots & T_n\end{bmatrix},\end{aligned}$$

where

$$1_{1\times(P+1)}:=\begin{bmatrix}1 & 1 & \cdots & 1\end{bmatrix}\in\mathbb{R}^{1\times(P+1)},$$

$$S:=\begin{bmatrix}-L_0^2 & -L_1^2 & \cdots & L_P^2\end{bmatrix}\in\mathbb{R}^{1\times(P+1)},$$

$$T_i:=\begin{bmatrix}2L_0x_i & 2L_1x_i & \cdots & 2L_Px_i\\ 2L_0y_i & 2L_1y_i & \cdots & 2L_Py_i\\ f(L_0)x_i & f(L_1)x_i & \cdots & f(L_P)x_i\\ -L_0^2 & -L_1^2 & \cdots & -L_P^2\\ (2dB^2+B-\ln\epsilon_0)e_i^{(n)} & (2dB^2+B-\ln\epsilon_0)e_i^{(n)} & \cdots & (2dB^2+B-\ln\epsilon_0)e_i^{(n)}\end{bmatrix}\in\mathbb{R}^{(2d+n+2)\times(P+1)}.$$

Here ϵ_0 is a parameter that we will designate later according to ϵ.

Now construct W_K,W_Q,W_V to be:

$$\begin{aligned}W_Q&:=I_{2d+n+2},\\W_K&:=I_{2d+n+2},\\W_V^\top&:=\begin{bmatrix}0_{d\times(d+1)} & I_d & 0_{d\times(n+1)}\end{bmatrix}\in\mathbb{R}^{d\times(2d+n+2)}.\end{aligned}$$

Therefore, by Definition E.1, the output becomes:

$$\mathrm{Hopfield}(\mathrm{Linear}_w(W);\mathrm{Linear}_x(X))=W_V\mathrm{Linear}_x(X)\mathrm{Softmax}(\beta\mathrm{Linear}_x(X)^\top\mathrm{Linear}_w(W)),$$

where

$$\mathrm{Softmax}(\mathrm{Linear}_x(X)^\top \mathrm{Linear}_w(W)) = \mathrm{Softmax}(\beta \begin{bmatrix} T_1 & T_2 & \cdots & T_n \end{bmatrix}^\top \begin{bmatrix} W \\ -1_{1\times n} \\ 0_{d\times n} \\ -1_{1\times n} \\ I_n \end{bmatrix}).$$

This is equivalent to:

$$(\mathrm{Linear}_x(X)^\top \mathrm{Linear}_w(W))_{:,c} = \begin{bmatrix} T_1^\top \\ T_2^\top \\ \vdots \\ T_n^\top \end{bmatrix} \cdot \begin{bmatrix} w \\ -1 \\ 0_d \\ -1 \\ e_c^{(n)} \end{bmatrix} = \begin{bmatrix} M_{1,c} \\ M_{2,c} \\ \vdots \\ M_{n,c} \end{bmatrix},$$

where

$$M_{i,c} := T_i^\top \cdot \begin{bmatrix} w \\ -1 \\ 0_d \\ -1 \\ e_i^{(n)} \end{bmatrix} = \begin{bmatrix} 2L_0 x_i^\top w - 2L_0 y_i - L_0^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c} \\ 2L_1 x_i^\top w - 2L_1 y_i - L_1^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c} \\ \cdots \\ 2L_P x_i^\top w - 2L_P y_i - L_P^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c} \end{bmatrix},$$

where $i \in [n]$ and $c \in [n]$, and $\mathbb{1}_{i=c}$ represents the indicator function of $i = c$.

This means that

$$\begin{aligned} &\mathrm{Softmax}(\beta \mathrm{Linear}_x(X)^\top \mathrm{Linear}_w(W))_{:,c} \\ &= \mathrm{Softmax}(\beta \begin{bmatrix} M_{1,c} \\ M_{2,c} \\ \vdots \\ M_{n,c} \end{bmatrix}) \\ &= \beta \sum_{i=1}^{n} \sum_{j=1}^{P} \frac{\exp\left\{2L_j x_i^\top w - 2L_j y_i - L_0^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}}{\sum_{i'=1}^{n} \sum_{j'=0}^{P} \exp\left\{u_{j'}^{(i')} + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}} e_{(i-1)P+j}^{(nP)}. \end{aligned}$$

Thus we have (without loss of generality, we ignore the β parameter in $\mathrm{Softmax}$):

$$\begin{aligned} &\mathrm{Hopfield}(\mathrm{Linear}_w(W); \mathrm{Linear}_x(X))_{:,c} \\ &= W_V \mathrm{Linear}_x(X) \mathrm{Softmax}(\mathrm{Linear}_x(X)^\top \mathrm{Linear}_w(W))_{:,c} \\ &\qquad\qquad (W_v \text{ only retrieves the } (d+2)\text{-th row in } T_i) \\ &= \begin{bmatrix} F_1 & \cdots & F_n \end{bmatrix} \sum_{i=1}^{n} \sum_{j=1}^{P} \frac{\exp\left\{2L_j x_i^\top w - 2L_j y_i - L_j^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}}{\sum_{i'=1}^{n} \sum_{j'=0}^{P} \exp\left\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}} e_{(i-1)P+j}^{(nP)} \\ &= \sum_{i=1}^{n} \sum_{j=0}^{P} \frac{\exp\left\{2L_j x_i^\top w - 2L_j y_i - L_j^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}}{\sum_{i'=1}^{n} \sum_{j'=0}^{P} \exp\left\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\right\}} f(L_j) x_i, \end{aligned}$$

where F is:

$$F_i := \begin{bmatrix} f(L_0)x_i & f(L_1)x_i & \cdots & f(L_P)x_i \end{bmatrix}.$$

For every $i \in [n]$, if $i \neq c$, we have

$$
\begin{aligned}
&\sum_{j=0}^{P} \frac{\exp\{2L_j x_i^\top w - 2L_j y_i - L_j^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \\
&= \sum_{j=0}^{P} \frac{\exp\{2L_j x_i^\top w - 2L_j y_i - L_j^2\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \\
&< \sum_{j=0}^{P} \frac{\exp\{2L_j x_i^\top w - 2L_j y_i - L_j^2\}}{\sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\Big\}} \\
&\hspace{20em} \text{(only taking the } i' = c \text{ part)} \\
&< \sum_{j=0}^{P} \frac{\exp\{2dB^2 + B\}}{P \exp(2dB^2 + B - \ln \epsilon_0)} = \epsilon_0.
\end{aligned}
$$

For $i = c$, since

$$
\sum_{i \neq c}^{n} \sum_{j=0}^{P} \frac{\exp\{2L_j x_i^\top w - 2L_j y_i - L_j^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \leq (n-1)\epsilon_0,
$$

we have

$$
\begin{aligned}
&\frac{\sum_{j=0}^P \exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \\
&= \sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \\
&\geq 1 - (n-1)\epsilon_0.
\end{aligned}
$$

Thus for the parts in the weighted sum output that corresponds to rows in $M_{:,c}$ in the attention score matrix, we have

$$
\begin{aligned}
&\|\sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{j'=0}^P \exp\Big\{2L_{j'} x_{i'}^\top w - 2L_{j'} y_{i'} - L_{j'}^2 + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} f(L_j) x_c - f(x_c^\top w - y_c) x_c\|_\infty \\
&= \|\sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{j'=0}^P \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}} (f(L_j) x_c - f(x_c^\top w - y_c) x_c) \\
&\quad \cdot \frac{\sum_{j'=0}^P \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{k=0}^P \exp\Big\{u_k^{(i')} + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}} \\
&\quad - (1 - \frac{\sum_{j'=0}^P \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{k=0}^P \exp\Big\{u_k^{(i')} + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}}) f(x_c^\top w - y_c) x_c\|_\infty \\
&\leq \sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{j'=0}^P \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}} |f(L_j) - f(x_c^\top w - y_c)| \cdot d\|x_c\|_\infty \\
&\quad - (1 - \frac{\sum_{j'=0}^P \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{i'=1}^n \sum_{k=0}^P \exp\Big\{u_k^{(i')} + (2dB^2 + B - \ln \epsilon_0)\mathbb{1}_{i=c}\Big\}}) |f(x_c^\top w - y_c)| \|x_c\|_\infty
\end{aligned}
$$

$$\leq \sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}}{\sum_{j'=0}^{P} \exp\Big\{u_{j'}^{(c)} + (2dB^2 + B - \ln \epsilon_0)\Big\}} |f(x_c^\top w - y_c)| \|x_c\|_\infty$$
$$+ (n-1)\epsilon_0 B_f \|x_c\|_\infty$$
$$= \sum_{j=0}^{P} \frac{\exp\Big\{u_j^{(c)}\Big\}}{\sum_{j'=0}^{P} \exp\Big\{u_{j'}^{(c)}\Big\}} |f(L_j) - f(x_c^\top w - y_c)| \|x_c\|_\infty + (n-1)\epsilon_0 B_f \|x_c\|_\infty$$
$$= \sum_{j=0}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} |f(L_j) - f(x_c^\top w - y_c)| \|x_c\|_\infty + (n-1)\epsilon_0 B_f \|x_c\|_\infty,$$

where we define $B_f := |f|$ as the bound for f.

For any $\epsilon_1 > 0$, set ΔL to be sufficiently small such that

$$|f(x) - f(y)| \leq \epsilon_1,$$

when $|x - y| \leq \Delta L$.

Then when β is sufficiently large, we have

$$\sum_{|L_i - (x_c^\top w - y_c)| > \Delta L} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} \leq \epsilon_2,$$

for any $\epsilon_2 > 0$.

Thus

$$\sum_{j=0}^{P} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} |f(L_j) - f(x_c^\top w - y_c)|$$
$$= \sum_{|L_i - (x_c^\top w - y_c)| > \Delta L} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} |f(L_j) - f(x_c^\top w - y_c)|$$
$$+ \sum_{|L_i - (x_c^\top w - y_c)| \leq \Delta L} \frac{\exp\{-\beta(x_c^\top w - y_c - L_j)^2\}}{\sum_{j'=0}^{P} \exp\{-\beta(x_c^\top w - y_c - L_{j'})^2\}} |f(L_j) - f(x_c^\top w - y_c)|$$
$$\leq \epsilon_2 \cdot 2B_f + \epsilon_1.$$

This completes the proof. □

Corollary E.2.1 (In-Context GD of Hopfield Layer)**.** Define

$$X := \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \end{bmatrix} \in \mathbb{R}^{(d+1)\times n} \quad \text{and} \quad W := [w \quad w \quad \cdots \quad w] \in \mathbb{R}^{d\times n},$$

where $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$ are the input-output pairs. $w \in \mathbb{R}^d$ is the linear coefficient we aim to optimize. For any differentiable loss function $\ell : \mathbb{R} \to \mathbb{R}$, There exists a Hopfield layer $\mathrm{Hopfield}$ with linear connections such that

$$\|\mathrm{Hopfield}(W; X) - \big[\nabla\ell(w^\top x_1 - y_1)x_1 \quad \nabla\ell(w^\top x_2 - y_2)x_2 \quad \cdots \quad \nabla\ell(w^\top x_n - y_n)x_n\big] \|_\infty \leq \epsilon,$$

for any $\epsilon > 0$.

Proof. Replacing the continuous function f in Theorem E.2 with $\nabla\ell$ completes the proof. □