A Theoretical Study of (Hyper) Self-Attention through the Lens of Mutual Interaction: Representation, Training, Generalization

Muhammed Ustaomeroglu [1] **Guannan Qu** [1]

Abstract

Self-attention has emerged as a core component of modern neural architectures, yet its theoretical underpinnings remain elusive. In this paper, we study self-attention through the lens of *interacting entities*, ranging from agents in multi-agent reinforcement learning to alleles in genetic sequences, and show that a single layer linear self-attention can *efficiently* represent, learn, and generalize functions capturing pairwise interactions, including out-of-distribution scenarios. Our analysis reveals that self-attention acts as a *mutual interaction learner* under minimal assumptions on the diversity of interaction patterns observed during training, thereby encompassing a wide variety of real-world domains. In addition, we validate our theoretical insights through experiments demonstrating that self-attention, along with our proposed extensions, effectively learns interaction functions and generalizes across both population distributions and out-of-distribution scenarios. Extending self-attention, we introduce *HyperFeatureAttention*, a novel neural network module designed to learn *couplings of different feature-level interactions* between entities. Furthermore, we propose *HyperAttention*, a new mechanism that extends beyond pairwise interactions to capture multi-entity dependencies, such as three-way, four-way, or general n-way interactions.

1. Introduction

Ever since the invention of Transformers (Vaswani et al., 2023), *attention* is the building block for many domains, spanning natural language processing (Brown et al., 2020; Devlin et al., 2019), computer vision (Dosovitskiy et al., 2021), protein structure prediction (Jumper et al., 2021), reinforcement learning (Chen et al., 2021). Despite the success of attention, our formal understanding of its representation, optimization, and generalization abilities is in its early stages.

Theoretical investigations illuminated Transformers' representational abilities/limitations (Yun et al., 2020; Edelman et al., 2022; Hu et al., 2024; Lu et al., 2024; Likhosherstov et al., 2021; Zhai et al., 2024; Bhattamishra et al., 2020; Sanford et al., 2023; Liu et al., 2023) and training dynamics (Wang et al., 2024; Yang et al., 2024; Huang et al., 2024; Chen & Li, 2024; Gao et al., 2024; Deora et al., 2023; Jelassi et al., 2022; Ahn et al., 2023) from different perspectives (e.g., language modeling, image patch classification, in context learning, etc.), to some extent. Despite this progress, current theoretical frameworks exhibit critical limitations. First, existing analyses often focus narrowly on isolated problems without developing a unified perspective to characterize Transformers' capabilities across domains. Secondly, many studies offer limited interpretation of learned parameters, whereas our approach not only analyzes model behavior but also deciphers the weight structures. Finally, most theoretical studies overlook a systematic evaluation of test-time generalization, particularly robustness to out-of-distribution (OOD) shifts. Our analysis addresses this gap by probing the underlying mechanisms that drive self-attention's generalization.

In this work, we adopt a broad perspective, which we call the *interacting entities* viewpoint, where each token represents an interacting entity (e.g., agents in multi-agent reinforcement learning, particles in physical simulations, amino acids in protein sequences, or words in natural language). Specifically, we introduce a function that models interactions among these entities and demonstrate its applicability across diverse domains, including the *colliding agents Environment*, *genotype-phenotype mapping task*, *vision task*, and *time series prediction*. Furthermore, we prove that a single layer linear self-attention can *efficiently* represent such functions[1] and show that gradient-descent training converges to the parameters realizing these interactions,

[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, USA. Correspondence to: Muhammed Ustaomeroglu <mustaome@andrew.cmu.edu>.

Proceedings of the 41st International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

[1]If linear self attention requires $\mathcal{O}(k)$ parameters to represent the interaction function, dense layer requires $\mathcal{O}(k \cdot L^2)$ parameters.

under mild assumptions on the data. In addition, we demonstrate versatility requirements on the train data such that the learned parameters generalize both to the test distribution and to out-of-distribution (length generalization).[2] By neither imposing restrictive constraints on model parameters nor limiting ourselves to particular domains, our framework unifies some diverse application scenarios and offers a novel theoretical lens on how Transformers learn dependencies among multiple interacting elements.

We further validate our theoretical insights on representation, convergence, and generalization through controlled experiments, demonstrating that the learned model parameters closely align with theoretical predictions. Beyond analyzing attention patterns, we highlight how the parameters themselves can be directly interpreted to uncover meaningful interactions among entities.

Building on these insights, investigations confirm that self-attention excels at capturing mutual interactions between entities. Motivated by this, we introduce two novel generalizations named (i) *HyperFeatureAttention*, for capturing couplings of different interactions between features of the entities, and (ii) *HyperAttention*, for capturing higher-order interactions (e.g. three-way or four-way) between entities. Extending our single-layer analysis, we show that HyperFeatureAttention can efficiently represent the couplings of feature interactions, while HyperAttention does the same for higher-order interactions.

We summarize our main contributions as follows:

- In Section 3, we present a unified perspective where each token (e.g., agent, amino acid, pixel patch) is treated as an *interacting entity*, seamlessly bridging applications from multi-agent reinforcement learning (MARL) to vision and time series analysis
- In Section 4, we show that gradient descent converges to a solution that exactly recovers the interaction function, under mild assumptions and a standard mean-squared error loss. Furthermore, the learned parameters *generalize* varying sequence lengths under suitable versatility conditions in the training set.
- In Section 7, experiments validate our theoretical predictions, revealing that learned attention parameters are *interpretable*.
- In Section 5, we introduce *HyperFeatureAttention*, a novel mechanism for capturing couplings of feature interactions. In Section 6, we present *HyperAttention*, which models *higher-order* dependencies between entities, such as three-way and four-way interactions.

[2]Linear self-attention preserves key optimization properties of full Transformers while reducing computational complexity and simplifying theoretical analysis (Ahn et al., 2024), making it a valuable tool for theoretical investigations. In Section 2, we provide a more detailed discussion on the usage of linear self-attention.

Related Works.

Transformer Representation Theory. A large body of work has illuminated the representational power of self-attention from various angles. For instance, Transformers have been shown to be Turing-complete and capable of simulating intricate sequential computations (Bhattamishra et al., 2020), act as provably efficient “compilers” for domain-specific languages (Zhai et al., 2024), and approximate key operators such as sparse or local averaging with sublinear complexity (Likhosherstov et al., 2021; Sanford et al., 2023; Edelman et al., 2022). Their abilities and limitations have also been explored in POMDP settings (Lu et al., 2024), automata-theoretic perspectives (Liu et al., 2023), sequence-to-sequence tasks (Yun et al., 2020), and hidden Markov model learning scenarios (Hu et al., 2024). Despite the valuable insights offered by these studies, the majority focus on specific domains or rely on stringent assumptions (e.g. input structures). Moreover, they often do not delve into interpreting the roles of the learned parameters beyond attention patterns. In contrast, our work adopts the unified interacting entities viewpoint encompassing various applications and provides a direct, parameter-level interpretation of how self-attention (and its variants) encodes interactions.

Transformer Convergence Analysis. Parallel to the progress on representation, another line of research has investigated the convergence properties of training Transformers. These studies analyze training via gradient flow in simplified yet insightful settings (Yang et al., 2024), establish conditions under which one can efficiently learn multi-head attention layers (Chen & Li, 2024; Deora et al., 2023), or employ mean-field methods to show global convergence in large-scale regimes (Gao et al., 2024). Additional works examine specialized domains such as masked visual pretraining (Huang et al., 2024) and spatial structure learning (Jelassi et al., 2022), or investigate sparse token selection tasks (Wang et al., 2024). While these studies offer theoretical guarantees and valuable insights into training dynamics, they often assume restricted data distributions or impose domain-specific structures. By contrast, our analysis neither confines itself to narrow tasks nor imposes strong restrictions on the parameters, allowing us to prove that *linear self-attention* recovers the target interaction function under mild assumptions. We further demonstrate that these learned parameters generalize to OOD examples (e.g. different sequence lengths) and uncover interpretable representations of the underlying interactions —an aspect less emphasized in prior works.

2. Preliminaries

Self-Attention. The self-attention mechanism is a core component of Transformers (Vaswani et al., 2023), enabling models to learn dependencies between input tokens effectively. For a sequence of L input tokens represented as a matrix $\mathbf{X} \in \mathbb{R}^{L\times d}$, the self-attention is defined as:

$$\mathbf{SA}^{\sigma}(\mathbf{X}) = \sigma\left(\frac{\mathbf{X}\mathbf{W}^{Q}(\mathbf{X}\mathbf{W}^{K})^{\top}}{\sqrt{d_k}}\right)\mathbf{X}\mathbf{W}^{V},$$

where $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d\times d_k}$ are the learnable projection matrices. Defining $\mathbf{C} = \mathbf{W}^Q(\mathbf{W}^K)^{\top}/\sqrt{d_k}$, we can write the same equation as

$$\mathbf{SA}^{\sigma}(\mathbf{X}) = \sigma\left(\mathbf{X}\mathbf{C}\mathbf{X}^{\top}\right)\mathbf{X}\mathbf{W}^{V}.$$

Since the introduction of *attention* mechanisms (Bahdanau et al., 2016), the function $\sigma : \mathbb{R}^{L\times L} \to \mathbb{R}^{L\times L}$ has been predominantly implemented as a row-wise softmax operation, where the input matrix to σ, commonly referred to as the *attention scores*, determines the relative importance of different tokens in the sequence.

Alternative Attention Functions. Recent advancements have explored alternative implementations of the function σ beyond the traditional softmax. One prominent direction involves linear self-attention mechanisms, which reduce the computational complexity of self-attention from $\mathcal{O}(L^2)$ to $\mathcal{O}(L)$. Linear attention methods approximate the softmax function while maintaining comparable performance in many tasks (Katharopoulos et al., 2020; Choromanski et al., 2022). For instance, the Performer model introduces kernel-based methods to approximate the softmax operation efficiently, achieving scalability without significant loss in accuracy (Choromanski et al., 2022). Moreover, alternative activation functions, such as ReLU, cosine, polynomial and sigmoid based transformations, have been explored, showing competitive or even superior performance compared to softmax in some tasks (Koohpayegani & Pirsiavash, 2024; Kacham et al., 2024; Ramapuram et al., 2024).

To simplify theoretical analysis while shedding light on a diverse range of self-attention implementations, we adopt a linear variant of self-attention, which preserves the core characteristics of self-attention through its reliance on *attention scores*.

Linear Self-Attention. Linear self-attention simplifies the attention by omitting the σ operation, resulting in:

$$\mathbf{SA}^{\text{lin}}(\mathbf{X}) = \left(\mathbf{X}\mathbf{C}\mathbf{X}^{\top}\right)\mathbf{X}\mathbf{W}^{V}. \quad (1)$$

Although omitting the σ operation may appear to be oversimplification, extensive theoretical and empirical studies confirm the power of linear self-attention. Notably, linear self-attention can implement gradient descent and preconditioned gradient descent (von Oswald et al., 2023; Ahn et al., 2023), and variants of softmax, ReLU, and linear transformers (including the exact version used here) can perform functional gradient descent to learn nonlinear functions in context (Cheng et al., 2024). Furthermore, (Ahn et al., 2024) demonstrates that linear self-attention replicates key optimization dynamics of Transformers -such as heavy-tailed gradient noise and ill-conditioned loss landscapes-without softmax or feedforward layers. This simplification retains the computational advantages of linearity while enabling rigorous analysis of phenomena like adaptive optimizer superiority (e.g., Adam over SGD) and gradient convergence. Critically, insights from this abstraction extend to softmax-based Transformers, particularly in understanding optimization stability and generalization under varying data distributions or model depths (Ahn et al., 2024).

In the following sections, we explore the capabilities of linear self-attention across diverse tasks to illustrate its practical and theoretical value. By striking a balance between tractability and expressiveness, linear self-attention offers a powerful framework for investigating and enhancing attention-based architectures.

3. Representing Mutual Interactions with Attention

Consider a discrete domain (or "vocabulary") $\mathcal{S} = \{\alpha, \beta, \gamma, \omega, \dots\}$ with cardinality $N = |\mathcal{S}|$. In our setting we have tuples of L elements (or sequences of length L) from the domain, denoted by $\mathcal{X}$ and entries of which are uniquely indexed by the integers in $[L]$. Here, $[i]$ denotes the set $\{0, 1, \dots, i-1\}$ for any $i \in \mathbb{Z}^{+}$. We define the function, $s : [L] \to \mathcal{S}$, assigning each index i to the corresponding element $s(i) \in \mathcal{S}$. Thus, we can write $\mathcal{X} = (s(0), s(1), \dots, s(L-1))$, which is distributed according to $\mathcal{X} \sim \mathcal{D}$. We also have tuples $\mathcal{Y}$ with elements from a corresponding relatively small set $\mathcal{S}_{\mathcal{Y}}$. The tuples are jointly distributed according to a task distribution $(\mathcal{X}, \mathcal{Y}) \sim \mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$. In order to train a neural network, we map each element of $\mathcal{S}$ to a d-dimensional embedding space via a function $\mathbf{x} : \mathcal{S} \to \mathbb{R}^d$ and each element of $\mathcal{S}_{\mathcal{Y}}$ to a corresponding vector or a scalar depending on the task, via the function $\mathbf{y} : \mathcal{S}_{\mathcal{Y}} \to \mathbb{R}^{d_2}$. Additionally, we stack the embeddings of the elements in $\mathcal{X}$ and $\mathcal{Y}$ as rows of $\mathbf{X} \in \mathbb{R}^{L\times d}$ and $\mathbf{Y} \in \mathbb{R}^{L\times d_2}$ matrices. We denote their distribution as $(\mathbf{X}, \mathbf{Y}) \sim \mathcal{P}_{\mathbf{X}\times\mathbf{Y}}$

Our first result concerns the representation of self-attention for *mutual interactions*, which we define below. We introduce a pairwise effect function. For $\alpha, \beta \in \mathcal{S}$, let $f(\alpha, \beta) \in \mathbb{R}$ measure *how strongly* entity β affects entity α, and let $\mathbf{w}_{\beta} \in \mathbb{R}^{d_2}$ represent *how* that influence is expressed. The *aggregated* effect on the i-th entity, from all other entities, is

$$\mathbf{y}_{s(i)} \;=\; \sum_{j\in[L]} f\big(s(i), s(j)\big)\, \mathbf{w}_{s(j)}, \tag{2}$$

capturing mutual interactions: each entity's behavior or state depends on every other entity in the sequence.

Theorem 3.1 (Representation Ability of Linear Self-Attention)**.** *$d = N$ is sufficient for a single-layer linear self-attention to **exactly** represent any aggregate pairwise interaction functions $\{\mathbf{y}_i\}_{i=1}^{L}$ in Eq. 2 for all entities simultaneously. Also, $d \geq N$ is necessary for a single layer linear self-attention to **exactly** represent any such functions.*

Consequently, self-self attention requires $\mathcal{O}(N^2)$ parameters to capture the interactions. The following theorem demonstrates that self-attention is *efficient* compared to fully connected neural networks.

Theorem 3.2 (Efficiency of Self-Attention)**.** *A linear fully connected network requires $\Omega(L^2 \cdot N^2)$ parameters to represent the aggregate pairwise interaction functions $\{\mathbf{y}_i\}_{i=1}^{L}$ in Eq. 2 **exactly**, for all entities simultaneously.*

The proofs of Theorems 3.1 and 3.2 are in Appendix B. Eq. 2 and Theorem 3.1 serve as the foundation for our subsequent examples. Despite the simplicity, Eq. 2 underlies many complex applications across multiple domains, from multi-agent problems and pixel-level pattern detection to time series forecasting and genotype-phenotype mapping, which we illustrate through the following examples. Though they arise in diverse domains, these examples underscore the versatility of linear self-attention in capturing a wide range of pairwise dependencies, in line with Theorem 3.1. Moreover, unlike much of the prior work, our approach not only interprets attention maps but also reveals a clear, rigorous *interpretation of the learned parameters*. We validate these representations in Appendices B and support our theoretical claims with convergence and generalization results in Section 4. Furthermore, we empirically validate both the theoretical findings and the learned parameter interpretations in Section 7.

Example 1 (Colliding Agents Environment). In a multi-agent system with L *identical* agents positioned in a dim-dimensional space, with position vectors $\mathbf{r}_i \in \mathbb{R}^{\dim}$ (or $\mathbf{r}_i \in [N]^{\dim}$ for a discrete setup). We aim to capture how each agent's value function V_i depends on other agents' initial states (positions).[3] To ensure translational and rotational invariance (i.e., shifting all agents by the same vector or rotating the coordinate axes does not affect the value functions), we define the pairwise effect as a function of relative distances, $f\big(\mathbf{r}_i - \mathbf{r}_j\big)\, w_{\mathbf{r}_j}$, where $w_{r_j} \in \mathbb{R}$ is a scalar weight and f depends only on their *relative position.* Then we consider the value function of the i-th agent as

$$V_i \;=\; \sum_{\substack{j\in[L]\\ j\neq i}} f\big(\mathbf{r}_i - \mathbf{r}_j\big)\, w_{\mathbf{r}_j}.$$

In Appendix B.1, we illustrate how a reward of -1 per distinct collision is captured by the value function above. This function fits directly into (2), allowing a single-layer linear self-attention to represent the multi-agent value function, when discrete positions are encoded orthogonally. In addition in Appendix B.1 we show similar results for non-identical agents setting, too.

Example 2 (Genotype-Phenotype Mapping Task). In many genotype–phenotype models, a DNA sequence of length L is represented by alleles $s(i) \in \mathcal{S}$. Some alleles are *always active*, others' activation level depends on the presence of certain alleles, and some remain *inactive* regardless of context (Frommlet et al., 2016). Formally, in a simplistic setting,

$$P_i \;=\; \mathbb{I}\{s(i) \text{ is always active}\} + \sum_{j\in[L]} \mathbb{I}\{s(i) \text{ is activated by } s(j)\}\; w_{s(j)},$$

where P_i captures activeness of the gene at position i. By orthogonally embedding each allele, Theorem 3.1 again ensures a single-layer linear self-attention can replicate these interactions exactly. For more details see the demonstrations at Appendix B.2.

For time series prediction, and vision task examples see Appendices B.4 and B.3

Why $d = N$? While setting the embedding dimension d equal to the domain size N may be impractical for large vocabularies, it simplifies our analysis without altering core insights. Our goal is to understand *how entities interact* through self-attention, and $d = N$ ensures orthogonal domain embeddings, yielding an exact and transparent representation. Compressing to $d < N$ is perpendicular to this focus and can be addressed separately using standard techniques, such as Johnson-Lindenstrauss projections, to approximate high-dimensional orthogonal embeddings. Starting with $d = N$ allows us to establish clean, exact theorems that elucidate how self-attention captures pairwise interactions, while reducing to $d < N$ merely introduces a small approximation gap without altering the core theory. For completeness, we provide an approximate case in Appendix B (Theorem B.4).

[3]In a reinforcement learning context, the value function V_i for agent i typically denotes the expected return (sum of discounted rewards) from a particular configuration.

4. Training and Generalization

In this section, we analyze how a single layer linear self-attention can achieve zero training error and demonstrate its *generalization* guarantees under mild assumptions. We focus on learning mutual interaction functions of the form (2), which Theorem 3.1 ensures can be represented by linear self-attention.

Setup and Notation. Let $\left\{\left(\mathbf{X}^{(n)}, \mathbf{Y}^{(n)}\right)\right\}_{n=1}^{B}$ be the training data, where $(\mathbf{X}^{(n)}, \mathbf{Y}^{(n)}) \sim \mathcal{P}^{L^*}_{\mathbf{X}\times\mathbf{Y}}$, for a specific L^*, so in the training set all tuples have the same length L. Also, let Let, $\mathcal{P}^{\forall L}_{\mathbf{X}\times\mathbf{Y}}$ be the universal distribution that covers samples of any length. Throughout, we focus on the mean-squared error (MSE) objective

$$L^{\mathrm{MSE}}\left(\mathbf{C}, \mathbf{W}^V\right) \;=\; \frac{1}{B}\sum_{n=1}^{B}\left\|\mathbf{SA}^{\mathrm{lin}}_{\mathbf{C},\mathbf{W}^V}\left(\mathbf{X}^{(n)}\right) \;-\; \mathbf{Y}^{(n)}\right\|^2,$$

We address three key questions: **(1) Convergence:** Under what conditions does gradient flow reach zero training error? **(2) Generalization:** When does a perfect fit on the training set imply zero error on *new* data from $\mathcal{P}^{L^*}_{\mathcal{X}\times\mathcal{Y}}$? **(3) OOD Generalization:** Can such a model generalize to *longer* or *shorter* sequences than those seen in training?

Definition 4.1 (Data Matrix for Element μ)**.** Let $\mathcal{B}_\mu$ as the set of training indices that contain element $\mu \in \mathcal{S}$, Denoting the number of times an element μ appears in tuple $\mathcal{X}^{(n)}$ as $s^{(n)}_\mu$, we define the data matrix for element μ as

$$\mathbf{s}^{(n)} = \begin{bmatrix} s^{(n)}_\alpha & s^{(n)}_\beta & \dots \end{bmatrix}^\top, \mathbf{S}_{\mathcal{B}_\mu} = \begin{bmatrix} \dots & \mathbf{s}^{(n)} & \dots \end{bmatrix}^\top_{n\in\mathcal{B}_\mu}.$$

In short, $\left[\mathbf{S}_{\mathcal{B}_\mu}\right]_{n\nu} = s^{(n)}_\mu$, but for n such that $\mu \in \mathcal{X}^{(n)}$.

Assumption 4.2 (Training Data Versatility)**.** For all $\mu \in \mathcal{S}$, $\mathbf{S}_{\mathcal{B}_\mu}$ is full column rank.

Remark 4.3. This assumption is mild in practice. When elements in $\mathcal{X}^{(n)}$ are drawn from a diverse distribution (e.g., uniformly or with non-degenerate correlations), the counts $\{s^{(n)}_\nu\}$ for $\nu \in \mathcal{S}$ naturally vary. This ensures that the columns of $\mathbf{S}_{\mathcal{B}_\mu}$ remain linearly independent, as redundant patterns (e.g., fixed linear relationships between element counts) are highly unlikely under unstructured or randomized data. Intuitively, as long as the data distribution lacks pathological correlations, $\mathbb{P}\left(\mathrm{rank}(\mathbf{S}_{\mathcal{B}_\mu}) < N\right) \leq Ne^{-\gamma|\mathcal{B}_\mu|}$ for some $\gamma \in \mathbb{R}$.

4.1. Convergence

We first show that if the target function is representable by a single-layer linear self-attention model (as guaranteed by Theorem 3.1), then gradient descent on $L^{\mathrm{MSE}}(\mathbf{C}, \mathbf{W}^V)$ converges *zero training error* under mild conditions. It is stated below with proof in Appendix C.

Theorem 4.4 (Convergence to Zero Training Error)**.** *Letting the embedding dimension $d = |\mathcal{S}|$ and $d_2 = 1$, under Assumptions 4.2, 4.5 and 4.6, gradient flow on $L^{\mathrm{MSE}}\left(\mathbf{C}, \mathbf{W}^V\right)$ converges to zero training error.*

Seeing that our main aim is *understanding* the attention scores, the core mechanism defining self-attention, we choose $d_2 = 1$ to simplify the analysis.

Assumption 4.5 (Weak Realizability)**.** The task is realizable, i.e, there exist $\mathbf{C}^*$ and $\mathbf{w}^*$ that perfectly fits the training data. That is, $\mathbf{y}^{(n)} = \left(X^{(n)}\mathbf{C}^* X^{(n)\top}\right) X^{(n)}\mathbf{w}^* \;\; \forall n \in \mathcal{B}$.

Assumption 4.6. All entries of $\mathbf{w} \triangleq \mathbf{W}^V \in \mathbb{R}^{d\times 1}$ remain non-zero in the embedding base. That is, $\langle \mathbf{x}(\alpha), \mathbf{w}\rangle \neq 0,\ \forall \alpha \in \mathcal{S}$.[4]

4.2. Test Generalization

Under training data versatility and strong realizability, achieving zero training error with linear self-attention implies *perfect generalization* to new data from the same distribution. Moreover, under even milder assumptions, zero test error ensures generalization to unseen sequence *lengths*.

Assumption 4.7 (Strong Realizability)**.** The task is strongly realizable, meaning there exist matrices $\mathbf{C}^\dagger$ and $\mathbf{W}^{V\dagger}$ such that the model perfectly fits the underlying population distribution at a fixed sequence length L^*. Specifically, we assume that $\mathbf{Y} = \left(\mathbf{X}\mathbf{C}^\dagger\mathbf{X}^\top\right)\mathbf{X}\mathbf{W}^{V\dagger}$ holds almost surely for $(\mathbf{X}, \mathbf{Y}) \sim \mathcal{P}^{L^*}_{\mathbf{X}\times\mathbf{Y}}$.

Due to Theorem 3.1 and Appendix B, we can safely assume strong realizability.

Theorem 4.8 (Generalization)**.** *Suppose Assumptions 4.2 and 4.7 hold, than zero training error forces $(\mathbf{C}, \mathbf{W}^V)$ to agree with $(\mathbf{C}^\dagger, \mathbf{W}^{V\dagger})$ on $(\mathbf{X}, \mathbf{Y}) \sim \mathcal{P}_{\mathbf{X}\times\mathbf{Y}}$. That is, the solution achieving zero training error also satisfies*

$$\mathbb{E}_{(\mathbf{X},\mathbf{Y})\sim\mathcal{P}_{\mathbf{X}\times\mathbf{Y}}}\left\|f_{\mathbf{C},\mathbf{W}^V}(\mathbf{X}) - \mathbf{Y}\right\| \;=\; 0.$$

Hence, it generalizes perfectly to new examples from the same distribution.

See Appendix D for the proof of Theorem 4.8.

4.3. Out of Distribution (Length) Generalization

A unique strength of self-attention is its ability to process sequences of *variable length*. In many tasks, we might train on sequences of length L^* yet hope to predict accurately on sequences of different lengths $L \neq L^*$. To state our result Theorem 4.10 (with proof in Appendix D.2) , we need the following assumption, which holds due to Theorem 3.1.

[4]This assumption exists solely to ensure $\langle \mathbf{x}(\alpha), \mathbf{w}\rangle^{-1}$ is well-defined. We also verified it in the colliding agents environment experiments. Moreover, it can be ensured through suitable initialization distributions.

Assumption 4.9 (Universal Realizability)**.** The task is universally realizable, meaning there exist matrices $\mathbf{C}^{\forall \mathrm{L}}$ and $\mathbf{W}^{V,\forall \mathrm{L}}$ such that the model perfectly fits the population distribution for all sequence lengths. Specifically, we assume that $\mathbf{Y} = \left(\mathbf{X}\mathbf{C}^{\forall \mathrm{L}}\mathbf{X}^\top\right)\mathbf{X}\mathbf{W}^{V,\forall \mathrm{L}}$ holds almost surely for $(\mathbf{X}, \mathbf{Y}) \sim \mathcal{P}^{\forall L}_{\mathcal{X}\times\mathcal{Y}}$.

Theorem 4.10 (Length Generalization)**.** *Under the Assumptions 4.2 and 4.9, any $\mathbf{C}^\dagger, \mathbf{W}^{V\dagger}$ that generalizes to $\mathcal{P}^{L^*}_{\mathbf{X}\times\mathbf{Y}}$, must generalize to $\mathcal{P}^{\forall L}_{\mathbf{X}\times\mathbf{Y}}$.*

4.4. Discussion

Our theoretical findings in this section show that a single-layer linear self-attention model not only converges to the correct parameters but also generalizes to both unseen data *and* unseen sequence lengths. In Section 7, we present controlled experiments that align with these theoretical predictions, confirming our convergence and generalization guarantees hold.

A key assumption throughout is *training data versatility* (Assumption 4.2). Intuitively, this requires each element in the domain to appear in sufficiently *diverse* contexts, ensuring the corresponding data matrix $\mathbf{S}_{\mathcal{B}_\mu}$ has full rank. This structural richness is crucial for generalization, as it allows the model to learn meaningful interactions that extend beyond the training set. In realistic settings—where domain elements vary meaningfully (e.g., different agent configurations, protein sequences, or natural language tokens)—such rank deficiencies are unlikely. Consequently, data versatility ensures that the model not only fits the training data but also generalizes effectively to new distributions and sequence lengths.

5. Extension to HyperFeatureAttention

In the previous sections, we showed how self-attention learns pairwise interactions between entities. However, in practical scenarios, the entities $\mu \in \mathcal{S}$ are not *monolithic*. In other words, they are composed of features. For instance, consider μ to be composed of $M \in \mathbb{Z}^+$ features, $\mu = (\mu_{\phi_1}, \mu_{\phi_2}, \ldots, \mu_{\phi_M})$, where $\mu_{\phi_i} \in \mathcal{S}_{\phi_i}$, so $\mathcal{S} = \mathcal{S}_{\phi_1} \times \ldots \times \mathcal{S}_{\phi_M}$. For example, the (red, green, blue) components of an RGB pixel in an image or (height, gender, weight, ethnicity,...) of a person in a population. To illustrate how the *couplings* of interactions between features are crucial, let us revisit the colliding agents environment of Section 3, with modifications.

Colliding Agents Revisited. Assume the same setting as before except now the agents are not identical. We need labels for the agents $\ell_i \in \mathcal{S}_\ell$. Thus, each agent is composed of two features $x_i = (\ell_i, \mathbf{r}_i)$. In Appendix E, we showed that for non-identical agents, it is natural to have the value function of the form

$$V_i = \sum_{j\in[L]} \prod_{a\in[A]} f_a\left(\phi_{a,i}, \theta_{a,j}\right), \tag{3}$$

where ϕ_a, θ_a are the corresponding features (label or position) picked depending on the scenario.

Seeing that $|\mathcal{S}| = |\mathcal{S}_\ell||\mathcal{S}_\mathbf{r}|$, from Theorem 3.1, a linear self-attention requires $d = \mathcal{O}(|\mathcal{S}_\ell||\mathcal{S}_\mathbf{r}|)$, so $\mathcal{O}(|\mathcal{S}_\ell|^2|\mathcal{S}_\mathbf{r}|^2)$ parameters to represent (3). However, defining new attention matrices, $\mathbf{C}^{(h,a)}$ for each f_a^h, we can represent the corresponding function only with $\mathcal{O}(|\mathcal{S}_\ell|^2)$, $\mathcal{O}(|\mathcal{S}_\mathbf{r}|^2)$, or $\mathcal{O}(|\mathcal{S}_\ell||\mathcal{S}_\ell|)$ parameters depending on which features are used for f_a^h. For example, $f(\ell_i, \ell_j)$ requires $\mathcal{O}(|\mathcal{S}_\ell|^2)$ parameters. As a result, if there are M features, self attention requires embedding dimension and number of parameters in $\mathcal{O}(\exp(M))$. However, the smaller functions, f_a, in total require $\mathcal{O}(M)$ parameters. For exact calculations please refer to Appendix E. That brings us to HyperFeatureAttention.

Definition 5.1 (Linear HyperFeatureAttention)**.**

$$\mathbf{HFA}^{\mathrm{lin}}(\mathbf{X}) = \Big\{\prod_{a\in[A]}^{\odot} \mathbf{X}\mathbf{C}^{(a)}\mathbf{X}^\top\Big\}\Big\{\prod_{a\in[A]}^{\odot} \mathbf{X}\mathbf{W}^{V,(a)}\Big\},$$

where $\prod^\odot$ is Hadamard (element-wise) product between the matrices, and $\mathbf{C}^{(a)}, \mathbf{W}^{V,(a)} \in \mathbb{R}^{d\times d}$.

From the preceding discussion, Linear HyperFeatureAttention requires only $\mathcal{O}(M)$ embedding dimension and parameters to express (3). One may concern that in practice we use layers of multi-head attention, which may possibly express Eq. 3, without exponential embedding dimension. However, we showed in Theorem E.3 that even two layer multihead linear self-attention cannot express (3). After all these motivations, we *formally* defined Multihead HyperFeatureAttention in Appendix E.2.

Remark 5.2 (A Short Note on Approximate Embeddings). Although it is somewhat *perpendicular* to our main focus on how entities (or features) interact as we explained in Section 3, we include this Remark completeness. In principle, self-attention may require an embedding dimension exponential in the number of features, but in practice, one often leverages the Johnson–Lindenstrauss lemma: in high-dimensional spaces, random projections yield vectors that are *approximately* orthonormal with embedding dimension only linear in the number of features. That is, each feature ϕ typically contributes $\mathcal{O}(\log|\mathcal{S}_\phi|)$ dimensions rather than $\mathcal{O}(|\mathcal{S}_\phi|)$, so if we consider approximate expressions $\mathcal{O}(M)$ embedding dimension is enough for linear self-attention. However, Johnson-Lindenstrauss argument holds for HyperFeatureAttention, too, which implies total embedding dimension requirement of $\mathcal{O}(\log M)$ (rather than $\mathcal{O}(M)$). Hence, even though both methods rely on approximate embeddings, *the exponential gap remains*: standard self-attention requires dimension linear in the number of features, whereas our module reduces it to logarithmic.

In short, just as self-attention generalizes dense layers by enabling entity (token)-level interactions, HyperFeatureAttention extends Self-Attention by enabling coupling between the feature interactions. It enhances traditional self-attention by allowing attentions to be coupled, which enables model to capture coupled feature level interactions. Unlike self-attention, which is restricted to bilinear forms, HyperFeatureAttention can represent cross-feature factorized interactions such as $\sum_j f^{(1)}(a_i, a_j) f^{(2)}(b_i, b_j)$ or their variants. This capability allows it to capture more complex relationships between features, making it particularly advantageous in tasks where cross-feature dependencies play a critical role.

6. Extension to HyperAttention

Similar to pairwise interactions, some tasks may require capturing higher-order interactions, three-way four-way, n-way. Thus, in a tuple $\mathcal{X}$ we may have $\mu \in \mathcal{S}$ that is influenced by a composite function of ν and $\gamma \in \mathcal{S}$. In this case, we would need our attention scores to be able to capture interaction functions of the form $f(\mu, \nu, \gamma)$. From this need, we developed a novel generalization, named *HyperAttention* for capturing higher-order interactions (e.g., three-way or four-way), alongside (Sanford et al., 2023; Alman & Song, 2023).[5] Here we state third order and linear version, please see Appendix F for the full version.

Definition 6.1 (Third order Linear HyperAttention)**.**

$$A_{ij_1j_2} = \sum_{\alpha\nu_1\nu_2}^{d} C_{\alpha\zeta_1\zeta_2} X_{i\alpha} X_{j_1\zeta_1} X_{j_2\zeta_2}$$

$$V_{j_1j_2\tau} = \sum_{\xi_1\xi_2}^{d} X_{j_1\xi_1} X_{j_2\xi_2} W^V_{\xi_1\xi_2\tau}$$

$$\mathrm{HA}^{\mathrm{lin}}_{i\tau}(\mathbf{X}) = \sum_{j_1j_2}^{L} A_{ij_1j_2} V_{j_1j_2\tau},$$

where we **denote** (i, j, k)-th entry of a tensor $\mathbf{T}$ as T_{ijk} and $\mathbf{C}, \mathbf{W}^V \in \mathbb{R}^{d\times d\times d}$.

Ternary Synergies in Multi-Agent Collaboration Imagine a multi-agent system in which each agent's payoff depends not just on pairwise interactions with other agents, but on *three-way* synergies. For instance, suppose agent i's reward increases only if it forms an alliance with agents j and k, *and* agents j and k also form an alliance with each other. Formally, each agent i has a discrete state $s(i)$ (e.g. its strategic type or coalition membership). Whenever i, j, and k all share a compatible configuration of states, i gains an additional bonus.

[5]However, their formulation is a low-rank approximation that assumes $V_{j_1j_2\tau} = V^1_{j_1\tau} V^2_{j_2\tau}$. In contrast, our approach offers greater representational capacity while maintaining comparable efficiency.

Concretely, define, $f\big(s(i),\, s(j),\, s(k)\big)$, a ternary synergy function which is nonzero only when i, j, and k are all in an appropriate joint configuration (e.g. all three are allied with each other). Let $w_{s(j),\,s(k)}$ be a learned weight that captures how the pair (j, k) specifically contributes to i's reward under that triple configuration. Then agent i's total payoff takes the form:

$$V_i \;=\; \sum_{j\in[L]} \sum_{\substack{k\in[L]\\ k\neq j}} f\big(s(i),\, s(j),\, s(k)\big)\, w_{s(j),\,s(k)}.$$

In this setup, standard *pairwise* interactions (as in ordinary self-attention) are insufficient to capture the *triple* compatibility requirement. However, assigning each state $s(\cdot)$ a suitable embedding enables encoding these ternary synergies into a higher-order extension of (2), allowing the resulting HyperAttention mechanism to learn how three agents jointly influence each other's rewards.

For an additional example illustrating the representational capabilities of HyperAttention, see Appendix F.2, where we analyze the "skip-trigram bug" (Elhage et al., 2021).

A potential concern is the $\mathcal{O}(L^3)$ computational cost introduced by the final equation in Definition 6.1, which may pose efficiency challenges for long sequences. Leveraging techniques similar to those in (Katharopoulos et al., 2020; Choromanski et al., 2022), we address this issue in Appendix F.3.

7. Experiments

We empirically validate the core theoretical claims of linear self-attention, i.e. its representational capacity (Theorem 3.1), convergence behavior (Theorem 4.4), and generalization guarantees (Theorem 4.8, Theorem 4.10) -using the colliding agents environment.

Setup: Colliding Agents on a Cylindrical Grid. Consider L *identical* agents on a cylindrical grid of size $[N] \times [N]$, with initial position vectors $\mathbf{r}_i = \begin{bmatrix} n_i^x & n_i^y \end{bmatrix} \in [N]^2$, where y axis is looped that is the distance between $n_i^y = 1$ and $n_i^y = N - 1$ is just 2. Each agent is modeled as a circle of radius R executes a simple "move-right" policy (one step to the right per time step if there is space to move, $n_i^x < N - 1$, otherwise stays where it is). Our goal is to capture how each agent's final value function V_i depends on the initial states (positions) of other agents: whenever agent i collides with agent j (distinct), it receives a penalty of -1.

We leverage Theorem 3.1 and show how this reward structure and the final value function can be exactly represented by a single layer linear self-attention. Under this setting, the final value function for each agent can be expressed as

$$V_i = -\sum_{\substack{j\in[L]\\ j\neq i}} \mathbb{I}\{\min\left(\left|n_i^y - n_j^y\right|, N - \left|n_i^y - n_j^y\right|\right) \leq 2R\} \tag{4}$$

Seeing that the value function depends only on the y-coordinates, we focus our discussion on a one-dimensional case for simplicity. The extension to value functions with higher dimension dependence follows naturally and is illustrated in Appendix B.1.

Under $L = 20$, $N = 360$, and $B = 100$k, we trained linear self-attention, for different embeddings *one-hot* and *sinusoidal*. It has an embedding dimension of N and its parameters are initialized with standard normal distribution, see Remark 4.3.

One-Hot Embedding. We first use a one-hot embedding: each position $n \in [N]$ is represented by the standard basis vector $\mathbf{e}_n \in \mathbb{R}^N$. In Appendix B.1, we demonstrate how a single-layer linear self-attention with $\mathbf{W}^V = -\mathbf{1}_N$, and

$$C_{mn} = \begin{cases} 1, & \min(|m-n|, N-|m-n|) \leq 2R, \\ 0, & \text{otherwise}, \end{cases} \tag{5}$$

can *exactly* implement the value function in (4), irrespective of L, so all realizability assumptions are satisfied. Under these, Theorem 4.4 predicts that the training mean squared error (MSE) converges to zero, which matches our observations in practice. Furthermore, as the Theorems 4.8 and 4.10 predict, we see generalization results: negligible error ($\Theta(10^{-7})$) on test sets both when $L = 20$ and when $L \in \{2, 5, 10, 30, 40\}$ varies.

Sinusoidal Embedding. We next consider a sinusoidal embedding, inspired by common positional encodings in attention models (Vaswani et al., 2023; Su et al., 2023). For even N, the position n_i of agent i is mapped to

$$\mathbf{p}_i = \Big[\frac{1}{\sqrt{2}}, \dots, \sin\left(\frac{2\pi k}{N} n_i\right), \cos\left(\frac{2\pi k}{N} n_i\right), \dots, \\ \frac{1}{\sqrt{2}}\cos\left(\frac{2\pi}{N}\left(\frac{N}{2}-1\right)n_i\right), \frac{1}{\sqrt{2}}\cos\left(\frac{2\pi}{N}\frac{N}{2}n_i\right)\Big]^T \in \mathbb{R}^N. \tag{6}$$

Note that $\mathbf{p}_i \top \mathbf{p}_j = \frac{N}{2}\delta_{i,j}$. Due to Lemma B.7 and Threorem B.9, a single layer linear self-attention with $\mathbf{W}^{V,\forall\mathrm{L}} = \begin{bmatrix} -\sqrt{2} & 0 & 0 & \dots \end{bmatrix}^\top \in \mathbb{R}^{N\times 1}$ and diagonal $\mathbf{C}^{\forall\mathrm{L}} \in \mathbb{R}^{N\times N}$, such that

$$C_{\mu\mu} = \begin{cases} \frac{4}{N}\left(2R + \frac{1}{2}\right) & \text{if } \mu = 0, \\ \frac{2}{N}\frac{\sin\left[\frac{2\pi}{N}\left(2R+\frac{1}{2}\right)\left(\frac{\mu+1}{2}\right)\right]}{\sin\left[\frac{2\pi}{N}\frac{1}{2}\left(\frac{\mu+1}{2}\right)\right]} & \text{if } \mu \text{ odd}, \\ \frac{2}{N}\frac{\sin\left[\frac{2\pi}{N}\left(2R+\frac{1}{2}\right)\left(\frac{\mu}{2}\right)\right]}{\sin\left[\frac{2\pi}{N}\frac{1}{2}\left(\frac{\mu}{2}\right)\right]} & \text{if } \mu \text{ even}, \end{cases}$$

can represent the value functions at Eq. 4 exactly, for any L, which is plotted at Figure 1. The entries of $\mathbf{C}$ are simply Fourier transform of the interaction function, which is an artifact of the sinusoidal embedding. Seeing that the same assumptions are satisfied for sinusoidal embedding, we observe the same convergence and generalization results for sinusoidal embedding.

Although the results match the theorems, the learned parameters do not overlap with the parameters we originally devised, expecially for the sinusoidal embedding, seen in Figure 2 -these learned parameters lack an intuitive, easy interpretation. However, as discussed in Corollary D.5, this outcome is a natural consequence of the generalization theories (Theorems 4.8 and 4.10). Therefore, we focus on $C_{\mu\nu}W^V_{\nu,0}$ and compare it with $C^{\forall\mathrm{L}}_{\mu\nu}W^{V,\forall\mathrm{L}}_{\nu,0}$. As shown in Figures 3 and 4, these matrices are indistinguishable from the theoretical counterparts. With only $\Theta\left(10^{-4}\right)$ mean square distance between their entries. Thus, in this simple setting, we can fully *interpret* the meaning of the parameters.

In summary, the experiments validates all our theories about self-attention. We gained a comprehensive understanding of the parameters it converges to, the functions it implements, and the fundamental aspects of linear self-attention.

8. Conclusion

We introduced a unifying theoretical framework that interprets tokens in self-attention as *interacting entities* and showed that a single layer linear self-attention mechanism can capture pairwise interactions across a broad range of domains. Our analysis clarified how self-attention learns and generalizes these interaction functions without relying on domain-specific assumptions. In particular, we proved that (i) such models converge under simple gradient-based training, (ii) they generalize perfectly to both unseen data, and (iii) and variable sequence lengths when the training set is sufficiently diverse

Beyond pairwise interactions, we introduced novel *HyperFeatureAttention* and *HyperAttention* modules that extend self-attention's abilities to capturing couplings of different feature level interactions and higher-order interactions among multiple entities.

Taken together, our results offer a transparent view of self-attention as a mechanism that *efficiently* learns and represents entity interactions. Experiments confirm our theoretical findings, demonstrating perfect training convergence and strong out-of-distribution performance. We hope these insights inspire the design of more interpretable and versatile Transformer-based architectures and motivate further exploration of softmax self-attention, deeper models, and alternative attention mechanisms from a similar broad perspective.

Impact Statement

This work advances the theoretical understanding of self-attention and its generalizations -HyperFeatureAttention and HyperAttention- by providing a unifying framework for learning complex interactions. Our insights could benefit diverse applications, including NLP, computer vision, multi-agent systems, and scientific modeling. Improved interpretability and training efficiency may enhance model reliability and user trust.

While theoretical, our findings could indirectly impact real-world risks, such as biased decision-making or misuse in misinformation and surveillance. Mitigating such risks requires policy measures rather than changes to fundamental theory. Additionally, optimizing attention mechanisms may contribute to more efficient models, reducing the environmental footprint of large-scale training. Overall, this work presents no unique societal risks beyond typical concerns in machine learning research.

References

Ahn, K., Cheng, X., Daneshmand, H., and Sra, S. Transformers learn to implement preconditioned gradient descent for in-context learning, November 2023. URL http://arxiv.org/abs/2306.00297. arXiv:2306.00297 [cs].

Ahn, K., Cheng, X., Song, M., Yun, C., Jadbabaie, A., and Sra, S. Linear attention is (maybe) all you need (to understand transformer optimization). *arXiv preprint arXiv:2310.01082*, 2024. URL https://arxiv.org/abs/2310.01082.

Alman, J. and Song, Z. How to Capture Higher-order Correlations? Generalizing Matrix Softmax Attention to Kronecker Computation, October 2023. URL https://arxiv.org/abs/2310.04064v1.

Bahdanau, D., Cho, K., and Bengio, Y. Neural Machine Translation by Jointly Learning to Align and Translate, May 2016. URL http://arxiv.org/abs/1409.0473. arXiv:1409.0473 [cs].

Bhattamishra, S., Patel, A., and Goyal, N. On the Computational Power of Transformers and its Implications in Sequence Modeling, October 2020. URL http://arxiv.org/abs/2006.09286. arXiv:2006.09286 [cs].

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfcb4967418bfb8ac142f64a-Abstract.html.

Chen, L., Lu, K., Rajeswaran, A., Lee, K., Grover, A., Laskin, M., Abbeel, P., Srinivas, A., and Mordatch, I. Decision Transformer: Reinforcement Learning via Sequence Modeling, June 2021. URL http://arxiv.org/abs/2106.01345. arXiv:2106.01345 [cs].

Chen, S. and Li, Y. Provably learning a multi-head attention layer, February 2024. URL http://arxiv.org/abs/2402.04084. arXiv:2402.04084 [cs, stat].

Cheng, X., Chen, Y., and Sra, S. Transformers implement functional gradient descent to learn non-linear functions in context. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 8002–8037. PMLR, July 2024. doi: 10.48550/arXiv.2312.06528. URL https://proceedings.mlr.press/v235/cheng24a.html. arXiv:2312.06528 [cs.LG].

Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., Belanger, D., Colwell, L., and Weller, A. Rethinking Attention with Performers, November 2022. URL http://arxiv.org/abs/2009.14794. arXiv:2009.14794 [cs].

Deora, P., Ghaderi, R., Taheri, H., and Thrampoulidis, C. On the Optimization and Generalization of Multi-head Attention, October 2023. URL http://arxiv.org/abs/2310.12680. arXiv:2310.12680 [cs, math, stat].

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, May 2019. URL http://arxiv.org/abs/1810.04805. arXiv:1810.04805 [cs].

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021. URL http://arxiv.org/abs/2010.11929. arXiv:2010.11929 [cs].

Edelman, B. L., Goel, S., Kakade, S., and Zhang, C. Inductive Biases and Variable Creation in Self-Attention Mechanisms, June 2022. URL http://arxiv.org/abs/2110.10090. arXiv:2110.10090 [cs, stat].

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

Frommlet, F., Bogdan, M., and Ramsey, D. *Phenotypes and Genotypes: The Search for Influential Genes*, volume 18 of *Computational Biology*. Springer, 2016. ISBN 978-1-4471-5309-2. doi: 10.1007/978-1-4471-5310-8. URL https://link.springer.com/book/10.1007/978-1-4471-5310-8.

Gao, C., Cao, Y., Li, Z., He, Y., Wang, M., Liu, H., Klusowski, J. M., and Fan, J. Global convergence in training large-scale transformers. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

Hu, J., Liu, Q., and Jin, C. On Limitation of Transformer for Learning HMMs, June 2024. URL http://arxiv.org/abs/2406.04089. arXiv:2406.04089 [cs].

Huang, Y., Wen, Z., Chi, Y., and Liang, Y. How Transformers Learn Diverse Attention Correlations in Masked Vision Pretraining. 2024.

Jelassi, S., Sander, M. E., and Li, Y. Vision Transformers provably learn spatial structure, October 2022. URL http://arxiv.org/abs/2210.09221. arXiv:2210.09221 [cs].

Jensen, H. J. *Complexity Science: The Study of Emergence*. Higher Education from Cambridge University Press, November 2022. doi: 10.1017/9781108873710. URL https://www.cambridge.org/highereducation/books/complexity-science/E0761D26BDAB25D75C6AB868AECE2F2D. ISBN: 9781108873710, Publisher: Cambridge University Press.

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P., and Hassabis, D. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. ISSN 1476-4687. doi: 10.1038/s41586-021-03819-2. URL https://www.nature.com/articles/s41586-021-03819-2. Publisher: Nature Publishing Group.

Kacham, P., Mirrokni, V., and Zhong, P. PolySketchFormer: Fast Transformers via Sketching Polynomial Kernels, March 2024. URL http://arxiv.org/abs/2310.01655. arXiv:2310.01655 [cs].

Katharopoulos, A., Vyas, A., Pappas, N., and Fleuret, F. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention. *Proceedings of Machine Learning Research*, August 2020. doi: 10.48550/arXiv.2006.16236. URL http://arxiv.org/abs/2006.16236. arXiv:2006.16236 [cs].

Koohpayegani, S. A. and Pirsiavash, H. SimA: Simple Softmax-free Attention for Vision Transformers, March 2024. URL http://arxiv.org/abs/2206.08898. arXiv:2206.08898 [cs].

Likhosherstov, V., Choromanski, K., and Weller, A. On the Expressive Power of Self-Attention Matrices, June 2021. URL http://arxiv.org/abs/2106.03764. arXiv:2106.03764 [cs].

Liu, B., Ash, J. T., Goel, S., Krishnamurthy, A., and Zhang, C. Transformers Learn Shortcuts to Automata, May 2023. URL http://arxiv.org/abs/2210.10749. arXiv:2210.10749 [cs, stat].

Lu, C., Shi, R., Liu, Y., Hu, K., Du, S. S., and Xu, H. Rethinking Transformers in Solving POMDPs, May 2024. URL http://arxiv.org/abs/2405.17358. arXiv:2405.17358 [cs].

Ramapuram, J., Danieli, F., Dhekane, E., Weers, F., Busbridge, D., Ablin, P., Likhomanenko, T., Digani, J., Gu, Z., Shidani, A., and Webb, R. Theory, Analysis, and Best Practices for Sigmoid Self-Attention, September 2024. URL http://arxiv.org/abs/2409.04431. arXiv:2409.04431 [cs].

Sanford, C., Hsu, D., and Telgarsky, M. Representational Strengths and Limitations of Transformers, November 2023. URL http://arxiv.org/abs/2306.02896. arXiv:2306.02896 [cs, stat].

Su, J., Lu, Y., Pan, S., Murtadha, A., Wen, B., and Liu, Y. RoFormer: Enhanced Transformer with Rotary Position Embedding. *arXiv preprint arXiv:2104.09864*, November 2023. doi: 10.48550/arXiv.2104.09864. URL http://arxiv.org/abs/2104.09864. Available at http://arxiv.org/abs/2104.09864.

Tropp, J. A. *An Introduction to Matrix Concentration Inequalities*. Foundations and Trends in Machine Learning, 2014. URL https://www.its.caltech.edu/~tropp/notes/Tropp-Matrix-Concentration-Inequalities.pdf. FnTML Draft, Revised, 24 December 2014.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention Is All You Need, August 2023. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs].

von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. Transformers learn in-context by gradient descent, 2023. URL http://arxiv.org/abs/2212.07677. arXiv preprint arXiv:2212.07677.

Wang, Z., Wei, S., Hsu, D., and Lee, J. D. Transformers Provably Learn Sparse Token Selection While Fully-Connected Nets Cannot, June 2024. URL http://arxiv.org/abs/2406.06893. arXiv:2406.06893 [cs, math, stat].

Yang, H., Kailkhura, B., Wang, Z., and Liang, Y. Training Dynamics of Transformers to Recognize Word Co-occurrence via Gradient Flow Analysis, October 2024. URL http://arxiv.org/abs/2410.09605. arXiv:2410.09605.

Yun, C., Bhojanapalli, S., Rawat, A. S., Reddi, S. J., and Kumar, S. Are Transformers universal approximators of sequence-to-sequence functions?, February 2020. URL http://arxiv.org/abs/1912.10077. arXiv:1912.10077 [cs, stat].

Zhai, X., Zhou, R., Zhang, L., and Du, S. S. Transformers are Efficient Compilers, Provably, October 2024. URL http://arxiv.org/abs/2410.14706. arXiv:2410.14706 [cs].

A. Notation and Definitions

Consider a discrete domain (or "vocabulary") $\mathcal{S} = \{\alpha, \beta, \gamma, \omega, \dots\}$ with cardinality $N = |\mathcal{S}|$. In our setting we have tuples of L elements (sequences of length L) from the domain, denoted by $\mathcal{X}$ and entries of which are uniquely indexed by the integers in $[L]$. Here, $[i]$ denotes the set $\{0, 1, \dots, i-1\}$ for any $i \in \mathbb{Z}^+$, positive integer. We define the function, $s : [L] \to \mathcal{S}$ assign each index i to the corresponding element $s(i) \in \mathcal{S}$. Thus, we can write $\mathcal{X} = (s(0), s(1), \dots, s(L-1))$, which is distributed according to $\mathcal{X} \sim \mathcal{D}$. We also have tuples $\mathcal{Y}$, length of which depends on the specific task we have, with elements from a corresponding relatively small set $\mathcal{S}_\mathcal{Y}$. The tuples are distributed according to a task distribution $(\mathcal{X}, \mathcal{Y}) \sim \mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$. When needed we use $\mathcal{D}^L$ and $\mathcal{D}^L_{\mathcal{X}\times\mathcal{Y}}$ to denote the same distributions for specifically tuple length of L. Similarly, $\mathcal{D}^{\forall L}$ means the distribution covering all possible lengths. In our training dataset, we have B such pairs that $\left(\mathcal{X}^{(n)}, \mathcal{Y}^{(n)}\right) \sim \mathcal{D}_{\mathcal{X}\times\mathcal{Y}}$, which are uniquely indexed by the elements in the set $\mathcal{B} = [B]$. We also define, $\mathcal{B}_\mu$ as the set of training indices that contain the element μ. We denote the number of times an element μ appears in tuple $\mathcal{X}^{(n)}$ as $s_\mu^{(n)}$. We define the corresponding $\mathbf{s}^{(n)} = \begin{bmatrix} s_\alpha^{(n)} & s_\beta^{(n)} & \dots \end{bmatrix}^\top$, $\mathbf{S} = \begin{bmatrix} \mathbf{s}^{(1)} & \mathbf{s}^{(2)} & \dots & \mathbf{s}^{(B)} \end{bmatrix}^\top \in \mathbb{R}^{B\times N}$ and $\mathbf{S}_{\mathcal{B}_\mu} = \begin{bmatrix} \dots & \mathbf{s}^{(n)} & \dots \end{bmatrix}^\top_{n\in\mathcal{B}_\mu}$ matrices.

In order to train a neural network, we map each element of $\mathcal{S}$ to a d-dimensional embedding space via a function $\mathbf{x} : \mathcal{S} \to \mathbb{R}^d$ and each element of $\mathcal{S}_\mathcal{Y}$ to a corresponding vector or a scalar depending on the task. We can now define the domain embedding matrix

$$\mathbf{B} = \begin{bmatrix} \mathbf{x}^\top(\alpha) \\ \mathbf{x}^\top(\beta) \\ \vdots \end{bmatrix}. \tag{7}$$

Additionally, we stack the embeddings of the elements in $\mathcal{X}$ and $\mathcal{Y}$ as rows of $\mathbf{X} \in \mathbb{R}^{L\times d}$ and $\mathbf{Y} \in \mathbb{R}^{L\times d_2}$ matrices, which we say are distributed according to $(\mathbf{X}, \mathbf{Y}) \sim \mathcal{P}^L_{\mathbf{X}\times\mathbf{Y}}$. For training sample n it can be written as,

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^\top\left(s(0)\right) \\ \mathbf{x}^\top\left(s(1)\right) \\ \vdots \\ \mathbf{x}^\top\left(s(L-1)\right) \end{bmatrix} = \begin{bmatrix} \mathbf{x}_0^\top \\ \mathbf{x}_1^\top \\ \vdots \\ \mathbf{x}_{L-1}^\top \end{bmatrix},$$

where in the last equality we denote the embedding of the i-th element as $\mathbf{x}_i := \mathbf{x}\big(s(i)\big) \in \mathbb{R}^d$, to reduce the notational cluttering.

We denote matrices and vectors by bold characters. Let $\mathbf{M} \in \mathbb{R}^{d_1\times d_2}$, $\mathbf{w} \in \mathbb{R}^d$ be any matrix and vector respectively. We denote k-th row of a $\mathbf{M}$ as $\mathbf{M}_{k,:}$ or $\mathbf{m}_k$, similarly we denote k-th column as $\mathbf{M}_{:,k}$ or $\mathbf{m}^k$. We denote k-th entry of $\mathbf{m}$ as m_k and (k,l)-th entry of $\mathbf{M}$ as M_{kl}. Similarly, if a vector is defined in terms of blocks we explicity state it and denote each the blocks as $\mathbf{w} = \begin{bmatrix} \mathbf{w}_1^\top & \mathbf{w}_2^\top & \dots \end{bmatrix}^\top$. The same notations naturally extend to tensors, e.g., we denote $(i, j, k, \dots)$-th entry of a tensor $\mathbf{T} \in \mathbb{R}^{d\times d\times d\times\cdots}$ as $T_{ijk\dots}$.

Lastly we have some operators including diagonalization

$$\operatorname{diag}(\mathbf{w}) = \begin{pmatrix} w_1 & & \\ & \ddots & \\ & & w_d \end{pmatrix}$$

and Kronecker product $\otimes$.

B. Representation Abilities of Linear Self-Attention

Proof of Theorem 3.1 (Representation Ability of Linear Self Attention). We first show **sufficiency** of $d = |\mathcal{S}|$. We can define an *orthonormal* embedding $\mathbf{x} : \mathcal{S} \to \mathbb{R}^N$ such that

$$\mathbf{x}(\alpha)^\top \mathbf{x}(\beta) \;=\; \delta_{\alpha,\beta} \quad \forall a, b \in \mathcal{S},$$

where $\delta_{a,b}$ is the Kronecker delta. Recall that $(\mathcal{X}, \mathcal{Y}) \sim \mathcal{D}^{\mathcal{X}\times\mathcal{Y}}$ are tuples whose elements are indexed from the set $[L]$. Also, recall that we let $s : [L] \to \mathcal{S}$ map each index i to its corresponding symbol $s(i) \in \mathcal{S}$. We arrange the embeddings $\mathbf{x}\big(s(i)\big)$ row-wise into $\mathbf{X} \in \mathbb{R}^{L\times N}$.

Goal. We wish to represent the family of functions

$$\mathbf{y}_i \;=\; \sum_{j\in[L]} f\big(s(i),\, s(j)\big)\, \mathbf{w}_{\,s(j)}, \quad \text{for each } i \in [L],$$

using a *single-layer linear self-attention* of dimension $d = N$. Here, $f : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ measures *how strongly* one entity affects another, and $\mathbf{w}_{\,s(j)} \in \mathbb{R}^{d_2}$ encodes *how* that influence is expressed (e.g., a direction vector or contribution to a subsequent feature).

Recall that a single-layer linear self-attention can be written as

$$\textbf{Attn}^{\text{lin}}(\mathbf{X}) \;=\; \big(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\big)\,\mathbf{X}\,\mathbf{W}^V,$$

where $\mathbf{C} \in \mathbb{R}^{d\times d}$ captures the *attention scores* (keys $\times$ queries) in a linear setting, $\mathbf{W}^V \in \mathbb{R}^{d\times d_2}$ represents the values transformation.

Constructing C and V. We define $\mathbf{C}$ and $\mathbf{W}^V$ to satisfy:

$$\mathbf{x}(\alpha)^\top\,\mathbf{C}\,\mathbf{x}(\beta) \;=\; f(\alpha,\beta), \quad \text{and} \quad \mathbf{x}(a)^\top\,\mathbf{W}^V \;=\; \mathbf{w}_a.$$

Such $\mathbf{C}$ and $\mathbf{V}$ exist because, $\{\mathbf{x}(a)\}_{a\in\mathcal{S}}$ forms an orthonormal basis in $\mathbb{R}^d$, due to $d = N$. Concretely, $\mathbf{C}$ can be chosen so that its bilinear form on basis vectors $\mathbf{x}(\alpha)$, $\mathbf{x}(\beta)$ equals $f(\alpha,\beta)$. Likewise, $\mathbf{W}^V$ can be chosen so that $\mathbf{x}(\alpha)$ maps to $\mathbf{w}_a$.

Verification. Given a sequence of length L, the matrix $\mathbf{X} \in \mathbb{R}^{L\times N}$ is

$$\mathbf{X} \;=\; \begin{bmatrix} \mathbf{x}\big(s(0)\big)^\top \\ \mathbf{x}\big(s(1)\big)^\top \\ \vdots \\ \mathbf{x}\big(s(L-1)\big)^\top \end{bmatrix}.$$

Thus,

$$\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top \;=\; \begin{bmatrix} f\big(s(0), s(0)\big) & \cdots & f\big(s(0), s(L-1)\big) \\ \vdots & \ddots & \vdots \\ f\big(s(L-1), s(0)\big) & \cdots & f\big(s(L-1), s(L-1)\big) \end{bmatrix},$$

and

$$\mathbf{X}\,\mathbf{W}^V \;=\; \begin{bmatrix} \mathbf{w}_{s(0)}^\top \\ \mathbf{w}_{s(1)}^\top \\ \vdots \\ \mathbf{w}_{s(L-1)^\top} \end{bmatrix}.$$

Multiplying these terms in the linear self-attention expression yields, for each row $i \in [L]$:

$$\big[\big(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\big)\,\mathbf{X}\,\mathbf{W}^V\big]_{i,:} \;=\; \sum_{j=0}^{L-1} f\big(s(i), s(j)\big)\,\mathbf{w}_{s(j)}.$$

This matches the desired function $\mathbf{F}_i \;=\; \sum_{j\in[L]} f\big(s(i), s(j)\big)\,\mathbf{w}_{s(j)}$. Hence, a single-layer linear self-attention with dimension N can represent *any* pairwise interaction function of this form.

Now, we focus on **necessity** of $d \geq |\mathcal{S}|$. Remember, $f : \mathcal{S} \times \mathcal{S} \to \mathbb{R}$ represent any pairwise interaction function. For fixed embeddings $\{\mathbf{x}(a) \in \mathbb{R}^d\}_{a\in\mathcal{S}}$ and any matrix $\mathbf{C} \in \mathbb{R}^{d\times d}$, the product $\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top$ is at most rank d. Formally, if we stack all dictionary embeddings $\mathbf{x}(a)$ into a matrix $\mathbf{X} \in \mathbb{R}^{N\times d}$, then

$$\text{rank}(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top) \;\leq\; d.$$

Thus, any pairwise function $f(a,b) = \mathbf{x}(a)^\top \mathbf{C}\,\mathbf{x}(b)$ produces an $N \times N$ matrix with rank at most d.

Consider the pairwise function $f^*(a,b)$ whose corresponding matrix $F^* \in \mathbb{R}^{N\times N}$ is full-rank, i.e., $\mathrm{rank}(F^*) = N$. For example, take $F^* = \mathbf{I}_N$ (the identity matrix), corresponding to $f^*(a,b) = \delta_{a,b}$ (the Kronecker delta).

If $d < N$, then any matrix $\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top$ has $\mathrm{rank} \leq d$ and hence cannot equal F^*, which has $\mathrm{rank}(F^*) = N > d$. Therefore, the self-attention mechanism cannot represent $f^*(a,b)$ exactly when $d < N$.

Since a single-layer linear self-attention mechanism of dimension $d < N$ cannot represent the full-rank function f^*, it follows that $d \geq N$ is *necessary* to exactly represent *all* pairwise functions $f(a,b)$. □

Remark B.1. Consequently, it requires $\mathcal{O}(N^2)$ parameters to exactly represent any such sequence to sequence interaction mapping seen in (2).

Remark B.2. The output dimension, d_2, plays a largely peripheral role in the self-attention mechanism's ability to capture pairwise interactions. Its primary function is to align with the desired output representation—whether it be the dimensionality of subsequent features or the final embedding size —without restricting the expressiveness of the attention scores. The latter is dictated by setting $d = |\mathcal{S}|$, ensuring that all necessary pairwise interactions $f(\alpha,\beta)$ can be effectively captured. Thus, d_2 should not be viewed as a bottleneck; it is simply a design choice for how the represented interactions are ultimately projected or stored.

Corollary B.3 (Orthogonal Transforms Preserve Representational Capacity)**.** *Under $d = |\mathcal{S}|$, let $\mathbf{x}(\alpha)$ be the orthonormal embedding constructed therein. For any orthogonal matrix $\mathbf{Q} \in \mathbb{R}^{d\times d}$, define the new embedding $\mathbf{x}'(\alpha) = \mathbf{Q}\,\mathbf{x}(\alpha)$. Then the same pairwise function $f(\alpha,\beta)$ can be represented exactly by redefining*

$$\mathbf{C}' \;=\; \mathbf{Q}\,\mathbf{C}\,\mathbf{Q}^\top \quad \textit{and} \quad \mathbf{W}'^V \;=\; \mathbf{Q}\,\mathbf{W}^V.$$

Hence, any orthonormal transformation of $\mathbf{x}(\alpha)$ leaves the representational capacity of the single-layer linear self-attention unchanged. Since the construction in Proof B relies only on the orthogonality of $\mathbf{x}(\alpha)$, the same argument applies to any embedding $\mathbf{x}'(\alpha) = \mathbf{Q}\,\mathbf{x}(\alpha)$ with $\mathbf{Q} \in \mathbb{R}^{d\times d}$ orthogonal. Thus, representational capacity is invariant to the choice of orthonormal basis.

Proof of Theorem 3.2 (Efficiency of Linear Self Attention). Let input sequence be $(s(1), s(2), \ldots, s(L))$, which we embed before feeding into fully connected network. Also, we have the corresponding output tuple $(y(1), y(2), \ldots y(L))$, which may have vector or scalar elements depending on the experiment.

- We know that a fully connected network is not permutation invariant, unlike the self attention, it will require L times the number of parameters what it requires for representing a single output $y(i)$.
- In order to represent $y(i)$, it requires to calculate $f\,(s(i), s(j))$ for all $j \in [L]$. It may not require if $f\,(s(i), s(3)) + f\,(s(i), s(4))$ is constant although we change $s(3)$ or $s(4)$. In this case it only requires to calculate the summation of those two instead of two of them separately, because we are summing all the pairwise functions. However, in this theorem we are looking at the most general interaction case where we do not assume anything about the interactions. Seeing that a fully connected network does not share parameters, it requires L times the number of parameters that it require to represent $f\,(s(i), s(j))$ for a single j.
- Seeing that $f\,(s(i), s(j))$ may have different value for all i, j pairs, so f may take N^2 possible values. To represent those different values exactly, any neural network requires N^2 number of parameters.

Combining these results, a fully connected neural network requires $\mathcal{O}(L^2N^2)$ parameters to represent any sequence to sequence mapping of the form (2). □

Theorem B.4 (Approximate Representation Abilities of Single-Layer Linear Self-Attention)**.** *Recall $\mathcal{S}$ is a finite set ($|\mathcal{S}| = N$) and $f : \mathcal{S}\times\mathcal{S} \to \mathbb{R}$ is any pairwise interaction function. For each symbol $a \in \mathcal{S}$, recall that $\mathbf{w}_a \in \mathbb{R}^{d_2}$ is (value) representation. Suppose we have a dimension $d \leq N$ and an embedding $\mathbf{x} : \mathcal{S} \to \mathbb{R}^d$. Then for every $\varepsilon > 0$, there exist parameters $\mathbf{C} \in \mathbb{R}^{d\times d}$ and $\mathbf{W}^V \in \mathbb{R}^{d\times d_2}$ for a single-layer linear self-attention mechanism such that, for* any *sequence $s : [L] \to \mathcal{S}$, the output of the linear self-attention block in Eq. 1 approximates the "pairwise-sum" mapping*

$$\mathbf{y}_i \;=\; \sum_{j=0}^{L-1} f\big(s(i),\, s(j)\big)\, \mathbf{w}_{s(j)}, \quad \textit{for all } i = 0, 1, \ldots, L-1, \tag{8}$$

to within ε (under any chosen norm, uniformly over i). In particular, as d grows, the approximation error can be made arbitrarily small even if $d < N$.

Proof. Index $\mathcal{S}$ by $\{1, 2, \ldots, N\}$ and form an $N \times N$ matrix F with (a, b)-entry $F_{a,b} = f(a, b)$. If we embed all symbols $a \in \mathcal{S}$ in $\mathbb{R}^d$ via $\mathbf{x}(a) \in \mathbb{R}^d$, we can collect these rows into a matrix $\mathbf{X} \in \mathbb{R}^{N \times d}$. Then any product $\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top$ is at most rank-d, so it represents a rank-d factorization of some $N \times N$ matrix in the row-column space of $\mathbf{X}$.

By the Eckart–Young theorem (or standard low-rank approximation results), among all rank-d matrices there is one (call it $\mathbf{B}_*$) that minimizes $\|\mathbf{B} - F\|$. Hence we can choose $\mathbf{C}_* \in \mathbb{R}^{d \times d}$ so that

$$\mathbf{B}_* \;=\; \mathbf{X}\,\mathbf{C}_*\,\mathbf{X}^\top \quad \text{and} \quad \|\mathbf{B}_* - F\| \;\le\; \varepsilon$$

for a suitable d. In other words, $\big|\mathbf{x}(a)^\top \mathbf{C}_* \mathbf{x}(b) - f(a, b)\big| \le \varepsilon$ for all $(a, b) \in \mathcal{S} \times \mathcal{S}$, under an appropriate norm.

Likewise, each $\mathbf{w}_a \in \mathbb{R}^{d_2}$ can be realized (or approximated) by a linear map from $\mathbf{x}(a)$. Formally, choose $\mathbf{W}^V_* \in \mathbb{R}^{d \times d_2}$ to solve

$$\min_{\mathbf{W}^V} \sum_{a=1}^{N} \Big\| \mathbf{x}(a)^\top\,\mathbf{W}^V - \mathbf{w}_a \Big\|^2.$$

Thus each row $\mathbf{x}(s(j))^\top \mathbf{W}^V_*$ is within ε' of $\mathbf{w}_{s(j)}$, where $\varepsilon' \to 0$ as d increases (assuming no rank deficit).

As a final check, for a sequence $\{s(i)\}_{i=0}^{L-1} \subset \mathcal{S}$, let $\mathbf{X} \in \mathbb{R}^{L \times d}$ be the row-stack of embeddings $\mathbf{x}(s(i))^\top$. Then the single-layer linear self-attention output at row i is

$$\Big[(\mathbf{X}\,\mathbf{C}_*\,\mathbf{X}^\top)\,\mathbf{X}\,\mathbf{W}^V_*\Big]_{i,:} \;=\; \sum_{j=0}^{L-1} \underbrace{\big[\mathbf{X}\,\mathbf{C}_*\,\mathbf{X}^\top\big]_{i,j}}_{\mathbf{x}(s(i))^\top\,\mathbf{C}_*\,\mathbf{x}(s(j))} \; \underbrace{\big[\mathbf{X}\,\mathbf{W}^V_*\big]_{j,:}}_{\mathbf{x}(s(j))^\top\,\mathbf{W}^V_*}.$$

By construction, $\mathbf{x}(s(i))^\top\,\mathbf{C}_*\,\mathbf{x}(s(j)) \approx f\big(s(i),\, s(j)\big)$ and $\mathbf{x}(s(j))^\top\,\mathbf{W}^V_* \approx \mathbf{w}_{s(j)}$. Hence the final output vector approximates $\sum_{j=0}^{L-1} f\big(s(i), s(j)\big)\,\mathbf{w}_{s(j)}$ to within $\varepsilon + \varepsilon'$.

Thus, even when $d < N$, we can realize (or approximate arbitrarily well) the pairwise interaction function (8). □

Justification for the $d = N$ Assumption. A common concern may arise from our theoretical results, which assume the embedding dimension d equals the domain size N. While this may seem restrictive for large vocabularies, this assumption is well-motivated for the following reasons:

- **Clarifying the Core Mechanism and Simplifying Analysis.** Our primary goal is to study *how self-attention models interactions between entities*, independent of embedding dimension constraints. Setting $d = N$ ensures an exact orthonormal representation, making both representational power and training dynamics fully transparent. This choice highlights the fundamental ability of self-attention to encode all pairwise interactions while simplifying the theoretical analysis without losing generality. By removing extraneous complexities, we preserve the core insights into representation, convergence, and generalization.
- **Establishing a Natural Theoretical Baseline.** The assumption $d = N$ serves as an idealized yet expressive starting point in theoretical analysis, providing a *one-to-one mapping from domain elements to embeddings*. This eliminates unnecessary confounding factors, allowing a clean study of self-attention's structural properties. Future work can systematically extend these results to cases where $d < N$, exploring compressed or approximate embeddings.
- **Scalability to Lower Dimensions.** Although we analyze $d = N$, our insights extend to $d < N$ through approximate orthonormal embeddings. Techniques such as *random projections* (e.g., via the Johnson–Lindenstrauss lemma) allow for efficient lower-dimensional embeddings while preserving essential properties. Thus, our results remain applicable in practical settings with small approximation errors.

In summary, the $d = N$ assumption is a deliberate choice to make our analysis transparent and tractable, enabling exact theorems that clarify the *core design principles* of self-attention. While not always practical, the insights gained from this assumption shed light on why self-attention mechanisms are so effective in learning interactions, thereby paving the way for future research on more approximate and scalable extensions.

Overview and Motivation of the Following Examples In this appendix, we present several illustrative examples that showcase how *single-layer linear self-attention* can capture important pairwise interactions across diverse domains: multi-agent collision settings, time series forecasting, genotype-phenotype mapping, simple vision tasks, and more. Although there exist many possible configurations, we focus on relatively straightforward, yet representative cases that make it clear how to embed domain-specific features into our theoretical framework.

B.1. Colliding Agents Environment

We consider L *identical* agents on a cylindrical grid of size $[N] \times [N]$, with initial position vector $\mathbf{r}_i = \begin{bmatrix} n_i^x & n_i^y \end{bmatrix} \in [N]^2$, where y axis is looped that is the distance between $n_i^y = 1$ and $n_i^y = N-1$ is just 2. It is like ants moving on the surface of a water pipe. Each agent is modeled as a circle of radius R executes a simple "move-right" policy (one step to the right per time step if there is space to move, $n_i^x < N-1$, otherwise stays where it is). Our goal is to capture how each agent's value function V_i depends on the initial states (positions) of other agents: whenever agent i collides with agent j (distinct), it receives a penalty of -1. Leveraging Theorem 3.1, we show how this reward structure (and hence the value function if continuing in time) can be exactly represented by a single-layer linear self-attention mechanism.

Under this setting, adding -1 bias to all value functions for simplicity, the value function for each agent can be expressed as

$$V_i = \sum_{j \in [L]} \mathbb{I}\{\min\left(\left|n_i^y - n_j^y\right|, N - \left|n_i^y - n_j^y\right|\right) \leq 2R\} \cdot (-1) \tag{9}$$

Seeing that the value function depends only on the y-coordinates, we focus our discussion on a one-dimensional case for simplicity. The extension to value functions with higher dimension dependence follows naturally and is illustrated after the one-dimensional case discussed here. In addition, we provide two versions of the similar representation construction for different positional embeddings.

B.1.1. One-Hot Embedding

One of the most straightforward approaches is to embed each agent's position $n_i \in [N]$ as the standard basis vector $\mathbf{e}_{n_i} \in \mathbb{R}^N$. Concretely, n_i-th entry of $\mathbf{e}_{n_i}$ is one and the rest are zeros. This guarantees $\mathbf{e}_{n_i}^\top \mathbf{e}_{n_j} = \delta_{n_i, n_j}$, making the embedding orthonormal.

Representing the Collision Value Function. By Theorem 3.1, there exists a single-layer linear self-attention model (dimension $d = N$) that encodes the collision-based value function V_i via an appropriate choice of $\mathbf{C} \in \mathbb{R}^{N \times N}$ and $\mathbf{W}^V \in \mathbb{R}^{N \times 1}$. To represent the value function in Eq. (9) using a single-layer linear self-attention mechanism, we explicitly construct the attention weight matrix $\mathbf{C}$ and value projection matrix $\mathbf{W}^V$. Define $\mathbf{W}^V \in \mathbb{R}^{N \times 1}$ as a vector of all -1s:

$$\mathbf{W}^V = -\mathbf{1}_N = \begin{bmatrix} -1 & -1 & \dots & 1 \end{bmatrix}^\top. \tag{10}$$

This ensures that the output of self-attention sums over the interactions weighted by $\mathbf{C}$. The interaction matrix $\mathbf{C} \in \mathbb{R}^{N \times N}$ is defined to capture the penalty for collisions:

$$C_{mn} = \begin{cases} 1, & \text{if } \min(|m-n|, N - |m-n|) \leq 2R, \\ 0, & \text{otherwise.} \end{cases} \tag{11}$$

Here, $C_{mn} = 1$ whenever the y-coordinates m and n are within a radius of $2R$, accounting for cylindrical wrapping. The resulting attention computation effectively sums over all nearby agents within the collision range.

B.1.2. Sinusoidal Embedding

We use sinusoidal embeddings, inspired by various positional embedding techniques that leverage sinusoidal representations. These include absolute positional embeddings from (Vaswani et al., 2023) and the more recent rotational embeddings in (Su et al., 2023). Specifically, assuming N is an even number, we embed the position of the i-th agent as:

$$\mathbf{p}_i = \begin{bmatrix} \frac{1}{\sqrt{2}} & \dots & \sin\left(\frac{2\pi k}{N} n_i\right) & \cos\left(\frac{2\pi k}{N} n_i\right) & \dots & \frac{1}{\sqrt{2}} \cos\left(\frac{2\pi}{N}\left(\frac{N}{2} - 1\right) n_i\right) & \frac{1}{\sqrt{2}} \cos\left(\frac{2\pi}{N} \frac{N}{2} n_i\right) \end{bmatrix}^\top \in \mathbb{R}^N. \tag{12}$$

Remark B.5. It is a simple exercise to check that $\mathbf{p}_i \top \mathbf{p}_j = \frac{N}{2} \delta_{i,j}$

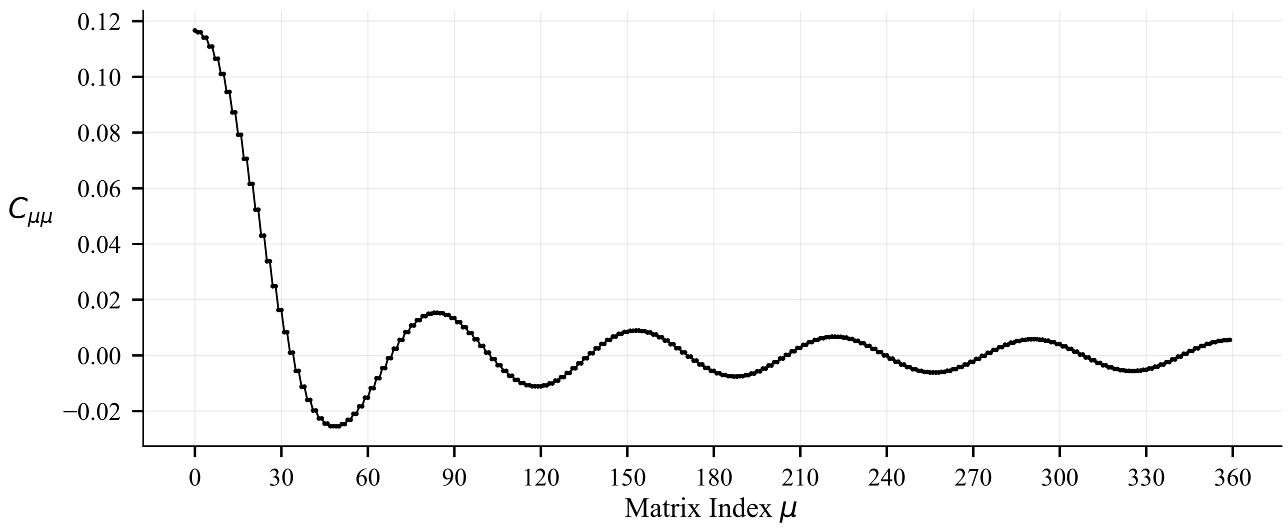

Figure 1. Diagonal entries of the interaction matrix C for $N = 360$ agents with radius $R = 5$. The pattern emerges from Fourier analysis of collision dynamics in the agent movement model. The matrix indices are from 0 to $N - 1$.

The *main result* of this section is that: due to Lemma B.7 and Threreom B.9, a single layer linear self attention 1 with $\mathbf{W}^V = \begin{bmatrix} -\sqrt{2} & 0 & 0 & \dots \end{bmatrix}^\top \in \mathbb{R}^{N \times 1}$ and diagonal $\mathbf{C} \in \mathbb{R}^{N \times N}$, such that

$$C_{\mu\mu} = \begin{cases} \frac{4}{N}\left(2R + \frac{1}{2}\right) & \text{if } \mu = 0, \\ \frac{2}{N}\frac{\sin\left[\frac{2\pi}{N}\left(2R+\frac{1}{2}\right)\left(\frac{\mu+1}{2}\right)\right]}{\sin\left[\frac{2\pi}{N}\frac{1}{2}\left(\frac{\mu+1}{2}\right)\right]} & \text{if } \mu \text{ odd}, \\ \frac{2}{N}\frac{\sin\left[\frac{2\pi}{N}\left(2R+\frac{1}{2}\right)\left(\frac{\mu}{2}\right)\right]}{\sin\left[\frac{2\pi}{N}\frac{1}{2}\left(\frac{\mu}{2}\right)\right]} & \text{if } \mu \text{ even}, \end{cases}$$

can represent the value functions at Eq. (9) exactly, **for any** L, which is plotted at Figure 1.Remember that the matrix indices are from 0 to $N - 1$.

Definition B.6 (Discrete-Time Fourier Series (DTFS))**.** DTFS of a function $f : [N] \to \mathbb{R}$ is defined as

$$F[k] = \sum_{n=0}^{N-1} f[n] e^{-j\frac{2\pi}{N}kn}.$$

Lemma B.7 (Discrete-Time Fourier Series Expansion of Window Function)**.** *Let f be:*

$$f[n] = \mathbb{I}\left[|n| \le 2R\right]$$

Then the DFS, F is:

$$F[k] = \begin{cases} \frac{1}{N}\frac{\sin\left[\frac{2\pi}{N}\left(2R+\frac{1}{2}\right)k\right]}{\sin\left[\frac{2\pi}{N}\frac{1}{2}k\right]} & \textit{if } k \neq 0 \\ \frac{2}{N}\left(2R + \frac{1}{2}\right) & \textit{if } k = 0 \end{cases}$$

Proof. Straightforward. □

Lemma B.8 (Real Valued, i.e. Sinusoidal, Expression of Discrete-Time Fourier Series)**.** *The Discrete Fourier Series (DFS) of a periodic discrete function $x[n]$ with period N is given by:*

$$x[n] = \sum_{k=0}^{N-1} X[k] e^{i\frac{2\pi}{N}kn}$$

Letting $a_k = 2\,\mathrm{Re}\{X[k]\}$ and $b_k = -2\,\mathrm{Im}\{X[k]\}$, the same function can be expressed as

$$x[n] = \begin{cases} \frac{a_0}{2} + \sum_{k=1}^{\frac{N-1}{2}} \left(a_k \cos\left(\frac{2\pi}{N}kn\right) + b_k \sin\left(\frac{2\pi}{N}kn\right)\right) & \textit{, for odd } N \\ \frac{a_0}{2} + \frac{a_{N/2}}{2}\cos\left(\frac{2\pi}{N}\frac{N}{2}n\right) + \sum_{k=1}^{\frac{N}{2}-1} \left(a_k \cos\left(\frac{2\pi}{N}kn\right) + b_k \sin\left(\frac{2\pi}{N}kn\right)\right) & \textit{, for even } N \end{cases}$$

Proof. Using Euler's formula, the complex exponential term can be rewritten as:

$$e^{i\frac{2\pi}{N}kn} = \cos\left(\frac{2\pi}{N}kn\right) + i\sin\left(\frac{2\pi}{N}kn\right)$$

Substitute this expression into the DTFS equation:

$$x[n] = \sum_{k=0}^{N-1} X[k]\left(\cos\left(\frac{2\pi}{N}kn\right) + i\sin\left(\frac{2\pi}{N}kn\right)\right)$$

Separate the real and imaginary parts of the equation:

$$\begin{aligned} x[n] = &\sum_{k=0}^{N-1}\left(\mathrm{Re}\{X[k]\}\cos\left(\frac{2\pi}{N}kn\right) - \mathrm{Im}\{X[k]\}\sin\left(\frac{2\pi}{N}kn\right)\right) \\ &+ i\sum_{k=0}^{N-1}\left(\mathrm{Im}\{X[k]\}\cos\left(\frac{2\pi}{N}kn\right) + \mathrm{Re}\{X[k]\}\sin\left(\frac{2\pi}{N}kn\right)\right) \end{aligned}$$

$x[n]$ is a real function, so the imaginary part sums to zero. Therefore, for odd N we have:

- $\mathrm{Im}\{X[0]\} = 0$
- $\mathrm{Im}\{X[k]\} + \mathrm{Im}\{X[N-k]\} = 0$ because cosine function has period of π
- $\mathrm{Re}\{X[k]\} - \mathrm{Re}\{X[N-k]\} = 0$ because sine function has period of π

As for the even N we have an additional requirement:

- $\mathrm{Im}\{X[N/2]\} = 0$

Thus $x[n]$ for odd N is:

$$x[n] = \mathrm{Re}\{X[0]\} + \sum_{k=1}^{\frac{N-1}{2}}\left(2\,\mathrm{Re}\{X[k]\}\cos\left(\frac{2\pi}{N}kn\right) - 2\,\mathrm{Im}\{X[k]\}\sin\left(\frac{2\pi}{N}kn\right)\right),$$

As for even N we have:

$$\begin{aligned} x[n] = &\,\mathrm{Re}\{X[0]\} + \mathrm{Re}\{X[N/2]\}(-1)^n \\ &+ \sum_{k=1}^{\frac{N}{2}-1}\left(2\,\mathrm{Re}\{X[k]\}\cos\left(\frac{2\pi}{N}kn\right) - 2\,\mathrm{Im}\{X[k]\}\sin\left(\frac{2\pi}{N}kn\right)\right), \end{aligned}$$

Letting $a_k = 2\,\mathrm{Re}\{X[k]\}$ and $b_k = -2\,\mathrm{Im}\{X[k]\}$, we get the final form as:

$$x[n] = \begin{cases} \frac{a_0}{2} + \sum_{k=1}^{\frac{N-1}{2}} \left(a_k \cos\left(\frac{2\pi}{N}kn\right) + b_k \sin\left(\frac{2\pi}{N}kn\right)\right) & \text{, for odd } N \\ \frac{a_0}{2} + \frac{a_{N/2}}{2} \cos\left(\frac{2\pi}{N}\frac{N}{2}n\right) + \sum_{k=1}^{\frac{N}{2}-1} \left(a_k \cos\left(\frac{2\pi}{N}kn\right) + b_k \sin\left(\frac{2\pi}{N}kn\right)\right) & \text{, for even } N \end{cases}$$

□

Theorem B.9. *Let us denote DTFS of a function $f[n]$ as $F[k]$. Defining $a_k = 2\,\mathrm{Re}\{F[k]\}$ and $b_k = -2\,\mathrm{Im}\{F[k]\}$ a attention score matrix $\mathbf{C}$ of the form*

$$\mathbf{C} = \begin{bmatrix} a_0 & & & & & & \\ & a_1 & b_1 & & & & \\ & -b_1 & a_1 & & & & \\ & & & \ddots & & & \\ & & & & a_{N/2-1} & b_{N/2-1} & \\ & & & & -b_{N/2-1} & a_{N/2-1} & \\ & & & & & & a_{N/2} \end{bmatrix}, \tag{13}$$

can represent any function $f[n_i - n_j] = \mathbf{p}_i \mathbf{C} \mathbf{p}_j^T$, where $\mathbf{p}_i$ is given at Eq. (12).

Proof. Setting $\mathbf{C}$ as in (13),

$$\mathbf{p}_i \mathbf{C} \mathbf{p}_j^T = \frac{a_0}{2} + \frac{a_{N/2}}{2} \cos\left(\frac{2\pi}{N}\frac{N}{2}n\right) + \sum_{k=1}^{\frac{N}{2}-1} \left(a_k \cos\left(\frac{2\pi}{N}kn\right) + b_k \sin\left(\frac{2\pi}{N}kn\right)\right).$$

That is equal to $f[n]$ owing to Lemma B.8, □

Remark B.10. If f is symmetric, then the corresponding $\mathbf{C}$ is diagonal.

B.1.3. More Complex Colliding Agents Environments

Value Functions That Depend on Both Coordinates. Assume the same basic grid setup described earlier, but now suppose each agent's value function depends on *both* coordinates. A trivial example is when agents also receive penalties for potential future collisions in the x-direction. Suppose agents with initial positions satisfying $|n_i^x - n_j^x| \le T_{\text{max}}$ (where T_{max} is the time horizon) *and* $|n_i^y - n_j^y| \le 2R$ collide during their rightward motion. The value function becomes $V_i = -\sum_{j\neq i} \mathbb{I}\{|n_i^y - n_j^y| \le 2R\} \cdot \mathbb{I}\{|n_i^x - n_j^x| \le T_max\}$, where the x-coordinate condition encodes temporal proximity. This can be captured by augmenting the embeddings in Eq. 12 with terms encoding n_i^x similarly, and using a bilinear attention kernel that interacts both coordinates.

Concretely, consider L agents on a two-dimensional $[N] \times [N]$ grid, with each agent i occupying a cell at integer coordinates $\mathbf{r}_i = \begin{bmatrix} n_i^x & n_i^y \end{bmatrix}^\top \in [N]^2$. Each agent's value function is then influenced by pairwise interactions of the form

$$V_i = \sum_{\substack{j\in[L] \\ j\neq i}} f\big(\mathbf{r}_i - \mathbf{r}_j\big)\, w_{\mathbf{r}_j} = \sum_{\substack{j\in[L] \\ j\neq i}} f\big(n_i^x - n_j^x,\; n_i^y - n_j^y\big)\, w_{\mathbf{r}_j},$$

where $f : [N]^2 \to \mathbb{R}$ measures *how* agent j (via its relative position) influences agent i. To represent f exactly as a linear self-attention kernel (as in Theorem 3.1), we embed *each* 2D position $\mathbf{r}_i$ into a vector of dimension N^2, defined by the Kronecker (outer) product

$$\mathbf{p}_i = \mathbf{p}_i^x \otimes \mathbf{p}_i^y \in \mathbb{R}^{N^2}.$$

Here, $\mathbf{p}_i^x, \mathbf{p}_i^y \in \mathbb{R}^N$ are the 1D sinusoidal embeddings from Equation (12), applied separately to the x- and y-coordinates. By construction, $\{\mathbf{p}_i\}_{i=1}^L$ spans an orthogonal set in $\mathbb{R}^{N^2}$, with one vector per distinct grid cell.

From 1D to 2D Discrete-Time Fourier Series. Recall from Lemma B.7 and Theorem B.9 that any function $g(n)$ on a 1D grid $[N]$ can be written as $\mathbf{p}_i^\top \mathbf{C}\, \mathbf{p}_j$ for a suitably chosen matrix $\mathbf{C}$. Precisely the same reasoning applies in two dimensions via a 2D Discrete-Time Fourier Series (DTFS). Specifically, one writes

$$f\big(n_i^x - n_j^x,\ n_i^y - n_j^y\big) \;=\; \sum_{k_x=0}^{N-1}\sum_{k_y=0}^{N-1} F\big[k_x, k_y\big]\ e^{\,i\,\frac{2\pi}{N}\,\big(k_x\,(n_i^x - n_j^x) + k_y\,(n_i^y - n_j^y)\big)},$$

and then rewrites each exponential in terms of sines/cosines matching the tensor-product embedding $\mathbf{p}_i^x \otimes \mathbf{p}_i^y$. Consequently, there exists a block-structured $\mathbf{C} \in \mathbb{R}^{N^2 \times N^2}$ such that

$$\mathbf{p}_i^\top\, \mathbf{C}\, \mathbf{p}_j \;=\; f\big(\mathbf{r}_i - \mathbf{r}_j\big).$$

Hence, by choosing $\mathbf{W}^V$ so that $\mathbf{p}_i \mapsto w_{\mathbf{r}_i}$, a *single-layer linear self-attention* (dimension $d = N^2$) recovers exactly the mapping

$$V_i \;=\; \sum_{j \neq i} f\big(\mathbf{r}_i - \mathbf{r}_j\big)\ \ w_{\mathbf{r}_j}.$$

All the 1D collision arguments from earlier carry over: once the domain is embedded into $\mathbb{R}^{N^2}$, Theorem 3.1 implies that single-layer linear self-attention can represent any pairwise function $f(\mathbf{r}_i - \mathbf{r}_j)$. Of course, this comes at the cost of embedding dimension N^2, so the resulting kernel $\mathbf{C}$ has size $N^2 \times N^2$, i.e. an $\mathcal{O}(N^4)$ parameter count for a fully general 2D interaction.

Non-Identical Agents. One can likewise handle agents with *different* behavior or policies. For example, suppose half of the agents always move right until they reach the boundary, while the others always move up. Label these two behaviors (or policies) via a discrete set $\mathcal{S}_q = \{\mathrm{R},\ \mathrm{U}\}$, and let each agent i carry an extra label $q_i \in \mathcal{S}_q$. Then the value function is

$$V_i \;=\; \sum_{\substack{j \in [L] \\ j \neq i}} f\Big(\mathbf{r}_i - \mathbf{r}_j,\ q_i,\ q_j\Big)\, w_{\mathbf{r}_j}. \tag{14}$$

Following precisely the same steps, we now view the domain of each agent as

$$\mathcal{S} \;=\; \mathcal{S}_q \;\times\; \mathcal{S}_x \;\times\; \mathcal{S}_y,$$

where $\mathcal{S}_x$ and $\mathcal{S}_y$ are each $[N]$. Its cardinality is $|\mathcal{S}| = 2\,N^2$, in this simple example. We then embed each agent's label and position as a vector in $\mathbb{R}^{2N^2}$. For instance, if $q_i = \mathrm{R}$ we might set

$$q_i \;=\; \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad \mathbf{p}_i^x,\ \mathbf{p}_i^y \;\in\; \mathbb{R}^N, \quad \mathbf{z}_i \;=\; \mathbf{q}_i \;\otimes\; \mathbf{p}_i^x \;\otimes\; \mathbf{p}_i^y \;\in\; \mathbb{R}^{2N^2}.$$

Likewise if $q_i = \mathrm{U}$, we flip those two bits in $\mathbf{q}_i$. By expanding the definition of $f\big(\mathbf{r}_i - \mathbf{r}_j, q_i, q_j\big)$ via a suitable DTFS, one again obtains a matrix $\mathbf{C}$ of size $(2N^2) \times (2N^2)$ that captures all pairwise interactions. Thus a single-layer linear self-attention with $d = 2N^2$ dimensions suffices to represent Eq. (14). The parameter count grows to $\mathcal{O}(4\,N^4)$ in the fully general case, but the construction exactly parallels the identical-agent scenario.

B.2. Genotype–Phenotype Mapping Task

Each *allele* (or gene variant) in the domain $\mathcal{S}$ is labeled with a unique integer and embedded using a one-hot vector. We consider a DNA sequence of length L in which every position corresponds to a unique allele. In other words, each allele appears at most once in the sequence (no duplicates). Our experiments randomly sample *activation relations*: an allele $\mu \in \mathcal{S}$ is *activated* if another allele ν exists somewhere in the sequence. Symbolically, we might store these relations in a dictionary of the form

$$\mu : [\nu_1, \nu_2, ...],$$

meaning "allele i is activated by the set of alleles-ν_k" If i : [ ] (an empty list), then allele i is *always* active, while alleles not present in the dictionary behave like redundant or inactive genetic material.

Constructing C and $\mathbf{W}^V$. To model these activation patterns via a single-layer linear self-attention, we build $\mathbf{C} \in \mathbb{R}^{d \times d}$ and $\mathbf{W}^V \in \mathbb{R}^{d \times 1}$ as follows (assuming each allele is one-hot embedded into $\mathbb{R}^d$, with $d = |\mathcal{S}|$):

- For every dictionary entry of the form $i : [j_1, \dots, j_m]$, set

$$\mathbf{C}_{i,\,j_k} \;=\; 1/m \quad \text{for each } k = 1, \dots, m,$$

  and set all other entries in row i to zero.

- For entries $i : [\,]$ (i.e., allele i is always active), assign each entry in row i to $\frac{1}{L}$. This ensures allele i gains a constant contribution from the entire sequence.

- Set every entry of $\mathbf{W}^V$ to 1 (i.e., $\mathbf{W}^V \in \mathbb{R}^{d \times 1}$ is a vector of all ones).

An example dictionary may be $\{$1:[3], 2:[], 4:[2]$\}$. Seeing that we have one hot encoding

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1/L & 1/L & 1/L & 1/L & 1/L \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

This is an example task for which a single layer linear self-attention cannot length generalize. However, a simple dense layer with ReLU activation addition after single layer linear self-attention, trivially enables it to generalize any length

B.3. Vision Task

In this example, each *token* or *entity* corresponds to a single pixel in an image. Suppose there are d possible pixel *positions* in the image, each with a positional embedding $\mathbf{p}_i \in \mathbb{R}^d$. We also have a binary indicator $b_i \in \{0, 1\}$ denoting whether pixel i is black ($b_i = 1$) or white ($b_i = 0$). We embed each pixel *jointly* as

$$\mathbf{x}_i \;=\; \big[b_i,\; \mathbf{p}_i\big] \;\in\; \mathbb{R}^{1+d}.$$

Thus, the first coordinate captures color, and the remaining coordinates capture position. We want to detect whether a specific *pattern of black pixels* occurs around the position $\mathbf{p}_i$. Formally, we want to detect if pixels in a set of relative offsets

$$\Delta \;=\; \{\Delta_1, \Delta_2, \dots\}$$

are all black around position i. For example, Δ might define a small pattern (e.g., a 3×3 cross), such that for each $\Delta_k \in \Delta$, the pixel at $\mathbf{p}_i + \Delta_k$ must be black for us to declare “shape present at i.”

Block-Structured C Matrix. Consider a single-layer linear self-attention of dimension $d + 1$. Let $\mathbf{C} \in \mathbb{R}^{(d+1)\times(d+1)}$ be block-structured, meaning:

$$\mathbf{C} \;=\; \begin{bmatrix} \mathbf{0}_{1\times 1} & \mathbf{0}_{1\times d} \\ \mathbf{0}_{d\times 1} & \widetilde{\mathbf{C}}_{d\times d} \end{bmatrix},$$

where $\mathbf{0}$ denotes a zero block (ensuring that the binary coordinate b_i does not directly alter *which* positions are relevant). If we want to detect a shape around $\mathbf{p}_i$ by checking offsets $\Delta = \{\Delta_1, \dots, \Delta_m\} \subseteq \mathbb{R}^d$, then for each row i (representing $\mathbf{p}_i$ in the $\widetilde{\mathbf{C}}$ submatrix), we set

$$\widetilde{\mathbf{C}}_{i,j} \;=\; \begin{cases} 1, & \text{if } \mathbf{p}_j \text{ is in } \mathbf{p}_i + \Delta, \\ 0, & \text{otherwise.} \end{cases}$$

This ensures $\mathbf{p}_i$ attends to exactly those pixel positions $\mathbf{p}_j$ that lie within the shape region around i.

Capturing the Binary Color. To incorporate the notion that only black pixels ($b_j = 1$) contribute, we define $\mathbf{W}^V \in \mathbb{R}^{(d+1)\times 1}$ so that its entries corresponding to the *binary part* of $\mathbf{x}_j$ are nonzero, while its entries corresponding to the *positional part* are zero. Symbolically,

$$\mathbf{W}^V \;=\; \begin{bmatrix} 1 \\ \mathbf{0}_d \end{bmatrix} \;\in\; \mathbb{R}^{1+d}.$$

Hence, when we multiply $\mathbf{x}_j$ by $\mathbf{W}^V$, the outcome is simply b_j. In other words, if $b_j = 1$, the pixel contributes; if $b_j = 0$, it does not.

Overall Operation. Putting it together, our single-layer linear self-attention

$$\left(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\right)\mathbf{X}\,\mathbf{W}^V$$

behaves as follows: (i) $\widetilde{\mathbf{C}}$ identifies the relevant offsets $\mathbf{p}_j$ around each $\mathbf{p}_i$, i.e., it checks which pixels could participate in a pattern at i. (ii) $\mathbf{W}^V$ converts the embedding $\left[b_j,\ \mathbf{p}_j\right]$ into b_j, effectively a blackness indicator. (iii) Summing across the image yields, for each position i, the count of black pixels at the appropriate offsets Δ.

If this count matches the size of Δ, we conclude that the shape is present around pixel i. Thus, the linear self-attention mechanism can recognize patterns of arbitrary size and structure *without changing the kernel size* or other architectural hyperparameters.

CNN Versus Transformer: Theoretical Insight. A single-layer Convolutional Neural Network (CNN) with a fixed 3×3 kernel cannot detect patterns larger than 3×3 within that same layer. Extending the receptive field would require either deeper networks or larger kernels, which *must be specified in advance*. This design constraint makes CNNs less flexible when the optimal pattern scale is not known a priori. That is the case for a layer of deep neural networks, the task of the layer is not known a priori, so we cannot choose the kernel accordingly. By contrast, Transformers can *attend* to any subset of positions in the input—whether nearby or distant—in a single layer. In our example, the shape offsets Δ might be large or irregular, yet the same $\widetilde{\mathbf{C}}$ submatrix can be adapted to capture those long-range relationships. Consequently, Transformers provide a more *versatile* framework for shape detection (which is simply learning how different parts of image interact with each other) or other vision tasks where the required pattern scale or geometry may vary significantly from one scenario to another.

B.4. Time Series Prediction

Consider a univariate or multivariate time series $\{\mathbf{m}[t]\}_{t=1}^{L+1}$, where each $\mathbf{m}[t] \in \mathbb{R}^{d_2}$ is the observed value at time t. We assume $\mathbf{m}[t]$ depends on a set of specific past delays $D = \{t_1, t_2, \dots\} \subset \mathbb{N}$, along with scalar multipliers $\{a_k\}_{k\in D}$ and a linear transform $\mathbf{A} \in \mathbb{R}^{d_2\times d_2}$ capturing how past values affect the current state:

$$\mathbf{m}[t] \;=\; \sum_{k\in D} a_k\,\mathbf{A}\,\mathbf{m}[t-k].$$

For instance, if $D = \{2, 5\}$, then $\mathbf{m}[t] = 3\,\mathbf{A}\,\mathbf{m}[t-2] + 7\,\mathbf{A}\,\mathbf{m}[t-5]$. For example, if $D = \{2, 5\}$, we get $\mathbf{m}[t] = 3\,\mathbf{A}\,\mathbf{m}[t-2] + 7\,\mathbf{A}\,\mathbf{m}[t-5]$.

To embed each time step t as an entity, define

$$\mathbf{x}_t \;=\; \big[\mathbf{m}[t],\,\mathbf{p}[t]\big] \;\in\; \mathbb{R}^{d+d_2},$$

where $\mathbf{m}[t] \in \mathbb{R}^{d_2}$ is the observed state, and $\mathbf{p}[t] \in \mathbb{R}^{d}$ is a positional embedding (e.g., one-hot or continuous encoding). An indicator-based formulation ensures the attention mechanism recovers delays in D:

$$\mathbf{m}[L+1] \;=\; \sum_{j\in[L]}\Big(\sum_{k\in D} a_k\,\mathbb{I}\big\{(L+1)-j=k\big\}\Big)\mathbf{A}\,\mathbf{m}[j].$$

Our goal is to show how a *single-layer linear self-attention* can represent this dependency structure exactly, following Theorem 3.1.

We treat each time step t as a distinct entity in the sequence. Its embedding $\mathbf{x}_t \in \mathbb{R}^{d+d_2}$ combines:

$$\mathbf{x}_t \;=\; \big[\mathbf{m}[t],\ \mathbf{p}[t]\big],$$

where:

- $\mathbf{m}[t] \in \mathbb{R}^{d_2}$ is the observed state at time t (univariate or multivariate).
- $\mathbf{p}[t] \in \mathbb{R}^{d}$ is a *positional embedding* encoding the index t. This could be a one-hot vector of length L, or a sinusoidal embedding.

Define a block-structured matrix $\mathbf{C} \in \mathbb{R}^{(d+d_2)\times(d+d_2)}$ to separate the $\mathbf{m}[t]$ coordinates from the positional coordinates $\mathbf{p}[t]$. We can denote:

$$\mathbf{C} \;=\; \begin{bmatrix} \mathbf{0}_{\,d_2\times d_2} & \mathbf{0}_{\,d_2\times d} \\ \mathbf{0}_{\,d\times d_2} & \widetilde{\mathbf{C}}_{\,d\times d} \end{bmatrix},$$

so that the *positional* part $\widetilde{\mathbf{C}}$ encodes which time delays matter, while the $\mathbf{m}[t]$ portion does not directly determine *which* indices to attend to.

Concretely, let $\widetilde{\mathbf{C}}_{u,v} = 1$ if position v is a valid activator for position u under one of the delays in D, and 0 otherwise. Equivalently, you can define

$$\widetilde{\mathbf{C}}_{u,v} \;=\; \sum_{k\in D} a_k \mathbf{I}\big[u - v = k\big],$$

if you index the positional embedding such that $u, v \in \{1, \dots, L\}$. This ensures that time step u attends to time step v if v is exactly one of the valid delays k behind u.

Next, we define $\mathbf{W}^V \in \mathbb{R}^{(d+d_2)\times d_2}$ to extract the actual state $\mathbf{m}[t]$ from each token:

$$\mathbf{W}^V \;=\; \begin{bmatrix} \mathbf{A} \\ \mathbf{0}_{\,d\times d_2} \end{bmatrix},$$

where $\mathbf{A} \in \mathbb{R}^{d_2\times d_2}$ is precisely the transformation from the autoregressive model. This construction means that when we multiply $\mathbf{x}_t$ by $\mathbf{W}^V$, we obtain $\mathbf{A}\,\mathbf{m}[t]$, and the positional coordinates are ignored in this step (since their block is zero).

Putting it all together, a single-layer linear self-attention computes

$$\Big(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\Big)\,\mathbf{X}\,\mathbf{W}^V.$$

For the row corresponding to the final time step $L+1$, the multiplication by $\mathbf{C}$ picks out those time steps j such that $(L+1) - j \in D$. Then multiplying by $\mathbf{W}^V$ retrieves $\mathbf{A}\,\mathbf{m}[j]$. Summing the contributions yields precisely

$$\mathbf{m}[L+1] \;=\; \sum_{k\in D} a_k\;\mathbf{A}\;\mathbf{m}[\,L+1-k\,],$$

mirroring the autoregressive formula.

C. Proof of Theorem 4.4: Convergence to Zero Training Error

We suggest a quick review of Appendix A (Notation and Definitions).

Lemma C.1. *If the domain embeddings are orthonormal, that is , $\langle x(\alpha), x(\beta)\rangle = \delta_{a,b},\ \forall\,\alpha,\ \beta \in \mathcal{S}$, then $\mathbf{X}^\top\mathbf{X}$ is diagonal in the embedding basis, and $\mathbf{B}\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{B}^\top = \mathrm{diag}(\mathbf{s}^{(n)})$.*

Proof. Just simple linear algebra. □

Since we set $d = N$, we can freely adopt an orthonormal domain embedding, ensuring that

$$\langle \mathbf{B}_{i,:}, \mathbf{B}_{j,:}\rangle = \delta_{i,j}.$$

For the remainder of this section, we conduct all calculations in the domain embedding basis. To formalize this, we define the following orthonormal transformations:

$$\mathbf{X}^{(n)\,\mathrm{one-hot}} = \mathbf{X}^{(n)}\mathbf{B}^\top,$$
$$\mathbf{C}^{\mathrm{one-hot}} = \mathbf{B}\mathbf{C}\mathbf{B}^\top,$$
$$\mathbf{w}^{\mathrm{one-hot}} = \mathbf{B}\mathbf{w}.$$

For notational simplicity, we omit the $\mathrm{one-hot}$ superscripts in the rest of this section, even though we are always referring to the one-hot transformed versions. That is, whenever we write $\mathbf{X}^{(n)}, \mathbf{C}, \mathbf{w}$, they actually represent

$\mathbf{X}^{(n)\,\text{one-hot}}$, $\mathbf{C}^{\text{one-hot}}$, $\mathbf{w}^{\text{one-hot}}$. Also, seeing that $\mathbf{B}$ is orthonormal matrix, when we establish the convergence of the one-hot parameter representations, it directly implies the convergence of the original parameters, which can be recovered by applying the inverse transformation in the $\mathbf{B}$ basis. What we do is simply change our perspective to how we look at coordinate system. Lastly, just for this section we define $\mathbf{e}_\mu \in \mathbb{R}^N$ unique one-hot encoded vector for all $\mu \in \mathbb{R}^N$.

Proof of Theorem 4.4 (Convergence to Zero Training Error). For convenience, define the residual (error) on the n-th example:

$$\mathbf{D}^{(n)} \;=\; f\big(\mathbf{X}^{(n)}\big) \;-\; \mathbf{y}^{(n)}.$$

When the training converges perfectly, $\mathbf{D}^{(n)} = \mathbf{0}$ for all n.

Gradients with Respect to C and w

$$\frac{\partial L}{\partial \mathbf{C}} \;=\; \frac{1}{B}\sum_{n=1}^{B}\big(\mathbf{D}^{(n)}\big)^\top \frac{\partial}{\partial \mathbf{C}} f\big(\mathbf{X}^{(n)}\big), \quad \frac{\partial L}{\partial \mathbf{w}} \;=\; \frac{1}{B}\sum_{n=1}^{B}\big(\mathbf{D}^{(n)}\big)^\top \frac{\partial}{\partial \mathbf{w}} f\big(\mathbf{X}^{(n)}\big).$$

Partial Derivatives. Since $d_2 = 1$ the linear self attention in Eq.1 can be written as

$$f(\mathbf{X}) \;=\; \Big(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\Big)\,\mathbf{X}\,\mathbf{w},$$

where $\mathbf{w} \in \mathbb{R}^d$. We have

$$\frac{\partial f(\mathbf{X})}{\partial C_{\mu\nu}} \;=\; \mathbf{X}_{:\mu}\,\Big[\mathbf{X}^\top \mathbf{X}\,\mathbf{w}\Big]_\nu.$$

Hence,

$$\frac{\partial L}{\partial C_{\mu\nu}} \;=\; \frac{2}{B}\sum_{n=1}^{B}\Big[\mathbf{X}^{(n)\,\top}\,\mathbf{X}^{(n)}\,\mathbf{w}\Big]_\nu\,\Big(\mathbf{X}^{(n)}_{:\mu}\Big)^\top \mathbf{D}^{(n)}.$$

Similarly,

$$\frac{\partial f(\mathbf{X})}{\partial w_\alpha} \;=\; \Big(\mathbf{X}\,\mathbf{C}\,\mathbf{X}^\top\Big)\,\mathbf{X}_{:\alpha}, \quad \text{so} \quad \frac{\partial L}{\partial w_\alpha} \;=\; \frac{2}{B}\sum_{n=1}^{B}\Big(\mathbf{X}^{(n)\,\top}\Big)_{\alpha:}\Big(\mathbf{X}^{(n)}\,\mathbf{C}^\top\,\mathbf{X}^{(n)\,\top}\Big)\,\mathbf{D}^{(n)}.$$

Gradient Flow for the Residuals and the Loss

Let t denote the (continuous) gradient-descent time, with

$$\frac{d\mathbf{C}}{dt} \;=\; -\,\eta\,\frac{\partial L}{\partial \mathbf{C}}, \quad \frac{d\mathbf{w}}{dt} \;=\; -\,\eta\,\frac{\partial L}{\partial \mathbf{w}}.$$

Defining $\mathbf{D}^{(m)} = f\big(\mathbf{X}^{(m)}\big) - \mathbf{y}^{(m)}$, the time derivative satisfies

$$\frac{d\mathbf{D}^{(m)}}{dt} \;=\; \frac{\partial f\big(\mathbf{X}^{(m)}\big)}{\partial \mathbf{C}}\,\frac{d\mathbf{C}}{dt} \;+\; \frac{\partial f\big(\mathbf{X}^{(m)}\big)}{\partial \mathbf{w}}\,\frac{d\mathbf{w}}{dt},$$

leading to an expression of the form

$$\frac{d\mathbf{D}^{(m)}}{dt} \;=\; -\,\frac{2\,\eta}{B}\sum_{n=1}^{B}\Big[\Big(\mathbf{w}^\top \mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{X}^{(m)\,\top}\mathbf{X}^{(m)}\mathbf{w}\Big) \otimes \Big(\mathbf{X}^{(n)}\mathbf{X}^{(m)\,\top}\Big)$$
$$+\,\mathbf{X}^{(n)}C\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{X}^{(m)\,\top}\mathbf{X}^{(m)}C^\top\mathbf{X}^{(m)\,\top}\Big]\,\mathbf{D}^{(n)}, \quad (15)$$

Stacking different samples with the following definitions

$$\mathbf{D} = \begin{bmatrix} \vdots \\ \mathbf{D}^{(n)} \\ \vdots \end{bmatrix},\ \mathbf{M} = \begin{bmatrix} \vdots \\ \Big(\mathbf{w}^\top \mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\Big) \otimes \mathbf{X}^{(n)} \\ \vdots \end{bmatrix},\ \mathbf{M_2} = \begin{bmatrix} \vdots \\ \mathbf{X}^{(n)}C\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)} \\ \vdots \end{bmatrix},$$

we can write the Eq. 15 as

$$\frac{d\mathbf{D}}{dt} = -\frac{2\eta}{B}\left[\mathbf{M}\mathbf{M}^\top + \mathbf{M}_2\mathbf{M}_2^\top\right]\mathbf{D}.$$

We can write the derivative of the loss as

$$\frac{dL}{dt} \;=\; \frac{2}{B}\sum_{m=1}^{B}\mathbf{D}^{(m)\,\top}\frac{d\mathbf{D}^{(m)}}{dt} = -\frac{4\eta}{B^2}\mathbf{D}^\top\left[\mathbf{M}\mathbf{M}^\top + \mathbf{M}_2\mathbf{M}_2^\top\right]\mathbf{D}.$$

Clearly, both $\mathbf{M}\mathbf{M}^\top$ and $\mathbf{M}_2\mathbf{M}_2^\top$ are at least positive semidefinite, so

$$\frac{dL}{dt} \;\le\; -\frac{4\,\eta}{B^2}\mathbf{D}^\top\mathbf{M}\mathbf{M}^\top\mathbf{D}. \tag{16}$$

Now, we will write Eq. 16 differently by re-expressing $\mathbf{D}$. Thanks to the realizability, we can write,

$$\mathbf{D}^{(n)} \;=\; \left(\mathbf{X}^{(n)}\,\mathbf{C}\,\mathbf{X}^{(n)\,\top}\right)\mathbf{X}^{(n)}\,\mathbf{w} \;-\; \left(\mathbf{X}^{(n)}\,\mathbf{C}^*\,\mathbf{X}^{(n)\,\top}\right)\mathbf{X}^{(n)}\,\mathbf{w}^*.$$

With the Assumption 4.6($\mathbf{w}_i \neq 0$), $\mathbf{w}^*{}_i/\mathbf{w}_i$ is defined for all $i \in [d]$. Thus we can define, diag $\left(\frac{\mathbf{w}^*}{\mathbf{w}}\right)$ to be the diagonal matrix, whose entries are $\mathbf{w}^*{}_i/\mathbf{w}_i$ in order.

$$\mathbf{D}^{(n)} \;=\; \left(\mathbf{X}^{(n)}\,\mathbf{C}\,\mathbf{X}^{(n)\,\top}\right)\mathbf{X}^{(n)}\,\mathbf{w} \;-\; \left(\mathbf{X}^{(n)}\,\mathbf{C}^*\,\mathbf{X}^{(n)\,\top}\right)\mathbf{X}^{(n)}\mathrm{diag}\left(\frac{\mathbf{w}^*}{\mathbf{w}}\right)\mathbf{w}.$$

In the orthonormal basis, $\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}$ is diagonal, which allows reordering to obtain

$$\mathbf{D}^{(n)} \;=\; \mathbf{X}^{(n)}\left[\mathbf{C} \;-\; \mathbf{C}^*\,\mathrm{diag}\!\left(\tfrac{\mathbf{w}^*}{\mathbf{w}}\right)\right]\mathbf{X}^{(n)\,\top}\,\mathbf{X}^{(n)}\,\mathbf{w}.$$

Vectorizing (using $\mathrm{vec}(\mathbf{A}\,\mathbf{X}\,\mathbf{B}) = (\mathbf{B}^\top \otimes \mathbf{A})\,\mathrm{vec}(\mathbf{X})$) yields

$$\mathbf{D}^{(n)} \;=\; \mathbf{M}^{(n)}\,\mathrm{vec}\Big[\mathbf{C} \;-\; \mathbf{C}^*\,\mathrm{diag}\!\left(\tfrac{\mathbf{w}^*}{\mathbf{w}}\right)\Big].$$

Stacking over n produces $\mathbf{D} = \mathbf{M}\,\mathrm{vec}\big(\mathbf{C} - \mathbf{C}^*\mathrm{diag}(\mathbf{w}^*/\mathbf{w})\big)$. Thus, Eq. 16 can be written as

$$\frac{dL}{dt} \;\le\; -\frac{4\,\eta}{B^2}\mathbf{D}^\top\mathbf{M}\mathbf{M}^\top\mathbf{D} \;=\; -\frac{4\,\eta}{B^2}\mathrm{vec}\Big[\mathbf{C} \;-\; \mathbf{C}^*\,\mathrm{diag}\!\left(\tfrac{\mathbf{w}^*}{\mathbf{w}}\right)\Big]^\top\mathbf{M}^\top\mathbf{M}\,\mathbf{M}^\top\mathbf{M}\mathrm{vec}\Big[\mathbf{C} \;-\; \mathbf{C}^*\,\mathrm{diag}\!\left(\tfrac{\mathbf{w}^*}{\mathbf{w}}\right)\Big]$$

Using the Lemma C.2, the same inequality can be written as

$$\frac{dL}{dt} \;\le\; -\frac{4\,\eta}{B^2}\lambda_{\min}\left(\mathbf{M}^\top\mathbf{M}\right)\left\|\mathbf{M}\mathrm{vec}\Big[\mathbf{C} \;-\; \mathbf{C}^*\,\mathrm{diag}\!\left(\tfrac{\mathbf{w}^*}{\mathbf{w}}\right)\Big]\right\|^2 = -\frac{4\,\eta}{B^2}\lambda_{\min}\left(\mathbf{M}^\top\mathbf{M}\right)\|\mathbf{D}\|^2$$

Thus, if $\mathbf{M}^\top\mathbf{M}$ is strictly positive definite, the training loss stops decreasing only when $\mathbf{D}$ reaches to all zero vector, i.e, training loss stops decreasing only when it reaches to zero.

Positive Definiteness of $\mathbf{M}^\top\mathbf{M}$. For a vector $\mathbf{u}$, the statement of $\mathbf{M}^\top\mathbf{M}\mathbf{u} = 0$ implies $\mathbf{u} = 0$, is equivalent to saying $\mathbf{M}^\top\mathbf{M}$ is positive definite. Now, let us look at the statement

$$\begin{aligned}
&\mathbf{M}^\top\mathbf{M}\mathbf{u} = 0,\\
&\Big\{\sum_n \left(\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{w}\mathbf{w}^\top\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\right)\otimes\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\Big\}\mathbf{u} = 0,\\
&\left(\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{w}\mathbf{w}^\top\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\right)\otimes\left(\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\right)\mathbf{u} = 0,\ \forall\, n
\end{aligned} \tag{17}$$

because all terms in the summation are positive-semi definite. Seeing that we do not have any empty training point that is $\mathbf{X}^{(n)} \neq \mathbf{0}$ and thanks to our $\mathbf{w}_i \neq 0$ assumption, $\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}\mathbf{w}\mathbf{w}^\top\mathbf{X}^{(n)\,\top}\mathbf{X}^{(n)}$ has one nonzero eigenvalue with corresponding eigenvector of $\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{w}$. Also, eigenvectors, with nonzero eigenvalue, of $\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}$ covers $\operatorname{span}\{\mathbf{e}_\mu\}_{\mu\in\mathcal{X}^{(n)}}$. Thus, by Eq. 17, $\forall\, k \in \mathcal{X}^{(n)}$

$$\left(\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{w}\otimes\mathbf{e}_\mu\right)^\top \mathbf{u} = 0\,,$$

just rearranging the entries of $\mathbf{u}$,

$$\left(\mathbf{e}_\mu\otimes\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{w}\right)^\top \tilde{\mathbf{u}} = 0\,.$$

We can also divide $\tilde{\mathbf{u}}$ into blocks of length d, so $\tilde{\mathbf{u}} = \begin{bmatrix}\tilde{\mathbf{u}}_1^\top & \tilde{\mathbf{u}}_2^\top & \dots\end{bmatrix}^\top$. Then the same equation can be written as,

$$\mathbf{w}^\top\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\tilde{\mathbf{u}}_\mu = 0\,,\ \ \forall\,\mu\in\mathcal{X}^{(n)}$$

Stacking it for different samples,

$$\begin{bmatrix}\vdots\\ \mathbf{w}^\top\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\\ \vdots\end{bmatrix}_{n\in\mathcal{B}_\mu}\tilde{\mathbf{u}}_\mu = 0\,, \tag{18}$$

Reminding ourself the definitions, $\mathbf{s}^{(n)}\in\mathbb{R}^d$ and $\mathbf{S}_{\mathcal{B}_\mu}$ (from Appendix A), we can write

$$\begin{bmatrix}\vdots\\ \mathbf{w}^\top\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\\ \vdots\end{bmatrix}_{n\in\mathcal{B}_\mu} = \begin{bmatrix}\vdots & \vdots & \vdots & \vdots\\ w_1 s_1^{(n)} & w_2 s_2^{(n)} & \dots & w_d s_d^{(n)}\\ \vdots & \vdots & \vdots & \vdots\end{bmatrix}_{n\in\mathcal{B}_\mu} = \mathbf{S}_{\mathcal{B}_\mu}\operatorname{diag}(\mathbf{w})$$

Due to the Assumption 4.2 and to Eq. 18, $\forall\ \mu$, $\tilde{\mathbf{u}}_\mu = \mathbf{0}$, so $\mathbf{u} = \mathbf{0}$. We showed that $\mathbf{M}^\top\mathbf{M}\mathbf{u} = 0$ implies $\mathbf{u} = \mathbf{0}$. Consequently, $\mathbf{M}^\top\mathbf{M}$ is positive definite. □

Lemma C.2. *For any matrix $\mathbf{M}$, and any vector $\mathbf{x}$ such that $\mathbf{M}\mathbf{x}$ is defined, then*

$$\mathbf{x}^\top\mathbf{M}^\top\mathbf{M}\,\mathbf{M}^\top\mathbf{M}\mathbf{x} \,\geq\, \lambda_{\min}\left(\mathbf{M}^\top\mathbf{M}\right)\|\mathbf{M}\mathbf{x}\|^2, \tag{19}$$

where $\lambda_{\min}\left(\mathbf{M}^\top\mathbf{M}\right)$ corresponds to minimum eigenvalue of $\left(\mathbf{M}^\top\mathbf{M}\right)$ matrix.

Proof. Obviously $\mathbf{A} = \left(\mathbf{M}^\top\mathbf{M}\right)$ is a symmetric and positive semi definite matrix, so it can be diagonalized as $\mathbf{A} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\mathbf{T}$ and its square root $\mathbf{A} = \mathbf{A}^{\frac{1}{2}}\mathbf{A}^{\frac{1}{2}}$ defined uniquely $\mathbf{A}^{\frac{1}{2}} = \mathbf{Q}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{Q}^\top$. It follows that

$$\begin{aligned}\mathbf{x}^\top\mathbf{A}^2\,\mathbf{x} &= \mathbf{x}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{A}\,\mathbf{A}^{\frac{1}{2}}\mathbf{x}\\ &= \mathbf{x}^\top\mathbf{A}^{\frac{1}{2}\,\top}\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{x}\\ &= \left(\mathbf{Q}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{x}\right)^\top\mathbf{\Lambda}\left(\mathbf{Q}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{x}\right)\\ &\geq \lambda_{\min}\|\mathbf{Q}^\top\mathbf{A}^{\frac{1}{2}}\mathbf{x}\|^2 = \lambda_{\min}\mathbf{x}^\top\mathbf{A}\mathbf{x} = \lambda_{\min}\|\mathbf{M}\mathbf{x}\|^2.\end{aligned}$$

□

D. Generalization Analysis of Linear Self-Attention

Lets denote the parameters we get from training as $\hat{\mathbf{C}}$ and $\hat{\mathbf{W}}^V$.

D.1. Proof of Theorem 4.8: Generalization

Proof of Theorem 4.8. The zero training error condition corresponds to $\forall\, n \in \mathcal{B}$

$$\mathbf{X}^{(n)}\hat{\mathbf{C}}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\hat{\mathbf{W}}^{V} = \mathbf{X}^{(n)}\mathbf{C}^{\dagger}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{W}^{V\dagger},$$

writing the same equation for each column separately, we get

$$\mathbf{X}^{(n)}\hat{\mathbf{C}}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\hat{\mathbf{w}}^{k} = \mathbf{X}^{(n)}\mathbf{C}^{\dagger}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{w}^{k\dagger},$$

where $\mathbf{w}^{k\dagger}$ is defined as k-th column of $\mathbf{W}^{V\dagger}$. We can, also, write it in the domain embedding base,

$$\mathbf{X}^{(n)}\mathbf{B}^{\top}\mathbf{B}\hat{\mathbf{C}}\mathbf{B}^{\top}\mathbf{B}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{B}^{\top}\mathbf{B}\hat{\mathbf{w}}^{k} = \mathbf{X}^{(n)}\mathbf{B}^{\top}\mathbf{B}\mathbf{C}^{\dagger}\mathbf{B}^{\top}\mathbf{B}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{B}^{\top}\mathbf{B}\mathbf{w}^{k\dagger}$$

By Lemma C.1,

$$\mathbf{B}\mathbf{X}^{(n)\top}\mathbf{X}^{(n)}\mathbf{B}^{T} = \operatorname{diag}\left(s_{\alpha}^{(n)}, s_{\beta}^{(n)}, \dots\right)$$

Defining $C_{\mu\nu} = \mathbf{x}(\mu)^{\top}\mathbf{C}\mathbf{x}(\nu)$ and $w_{\alpha} = \mathbf{x}(\alpha)^{\top}\mathbf{w}$ for any $\mu, \nu, \alpha \in \mathcal{S}$,

$$\sum_{\nu\in\mathcal{S}} \hat{C}_{\mu\nu} s_{\nu}^{(n)} \hat{w}_{\nu}^{k} = \sum_{\nu\in\mathcal{S}} C_{\mu\nu}^{\dagger} s_{\nu}^{(n)} w_{\nu}^{k\dagger},$$

$$\sum_{\nu\in\mathcal{S}} s_{\nu}^{(n)}\left(\hat{C}_{\mu\nu}\hat{w}_{\nu}^{k} - C_{\mu\nu}^{\dagger} w_{\nu}^{k\dagger}\right) = 0\,, \quad \forall\, \mu \in \mathcal{X}^{(n)} \text{ and } \forall\, n \in \mathcal{B} \tag{20}$$

For each specific μ and k, the terms in the last equation in parentheses can be construed as a vector with indices over ν. Denoting that vector as $\mathbf{a}^{\mu k}$ and remembering $\mathbf{S}_{\mathcal{B}_{\mu}}$ definition (from Appendix A), Eq. 20 can be written as

$$\mathbf{S}_{\mathcal{B}_{\mu}}\mathbf{a}^{\mu k} = 0$$

Thanks to the Assumption 4.2, $\mathbf{S}_{\mathcal{B}_{\mu}}$ is full column rank so only solution to last equation is $\mathbf{a}^{\mu k} = \mathbf{0}$, which corresponds to $\hat{C}_{\mu\nu}\hat{w}_{\nu}^{k} = C_{\mu\nu}^{\dagger} w_{\nu}^{k\dagger} \;\; \forall\, \mu \in \mathcal{S},\; k \in [d]$. Consequently, for any $\mathbf{X}$ which is embedding of a corresponding $\mathcal{X} \sim \mathcal{D}$

$$\mathbf{X}\hat{\mathbf{C}}\mathbf{X}^{\top}\mathbf{X}\hat{\mathbf{W}}^{V} = \mathbf{X}\mathbf{C}^{\dagger}\mathbf{X}^{\top}\mathbf{X}\mathbf{W}^{V\dagger} = \mathbf{Y}$$

Thus the learned parameters generalizes to all population, that is,

$$\mathbb{E}_{\mathcal{D}_{\mathcal{X}\times\mathcal{Y}}}\left[\left\|\mathbf{Y} - \left(\mathbf{X}\hat{\mathbf{C}}\mathbf{X}^{\top}\right)\mathbf{X}\hat{\mathbf{W}}^{V}\right\|\right] = 0$$

□

Remark D.1. The conclusions of the Assumption 4.2 are so strong that they even lead to generalization to all population distribution from which data collected by making sure that the set of all parameters that lead to zero training error also leads to zero population error, i.e, there is not any set of parameters that lead to zero training error but has some population level error.

D.2. Length Generalization

In the preceding analysis, we showed that under some mild assumptions, achieving zero training error leads to robust population-level generalization for the specific length L from which training data is sampled. However, many of our interaction-based experiments naturally suggest *length generalization*: the outputs of these models are not inherently tied to a fixed sequence length, nor did our experimental design depend on a specific number of entities. That brings us to the Assumption 4.9. Despite this, our test-time generalization results do not formally guarantee out-of-distribution generalization to unseen sequence lengths. In this section, we analyze the conditions on $\mathcal{D}^{L^*}$ under which the parameters that generalize to the distribution $\mathcal{D}^{L^*}$ generalize to $\mathcal{D}^{\forall L}$.

Let us first look at possibility of parameters that has zero error for $\mathcal{X}^{L^*} \sim \mathcal{D}^{L^*}$ but does not generalize to any other length L. Writing them in terms of a specific length generalizing parameters, $\mathbf{C}^{L^*} = \mathbf{C}^{\forall \mathrm{L}} + \mathbf{C}^{\Delta}$ and $\mathbf{w}^{k,L^*} = \mathbf{w}^{k,\forall \mathrm{L}} + \mathbf{w}^{\Delta}$, $\forall\, \mu \in \mathcal{X}^{L^*}$:

$$y_{\mu}^{k,L^*}\left(\mathcal{X}^{L^*}\right) = \sum_{\nu\in\mathcal{X}^{L^*}} \left(\mathbf{C}^{\forall \mathrm{L}} + \mathbf{C}^{\Delta}\right)_{\mu\nu}\left(\mathbf{w}^{k,\forall \mathrm{L}} + \mathbf{w}^{k\Delta}\right)_{\nu}$$

$$= \sum_{\nu \in \mathcal{X}^{L^*}} \left(C^{\forall \mathrm{L}}_{\mu\nu} w^{k,\forall \mathrm{L}}_{\nu} + C^{\forall \mathrm{L}}_{\mu\nu} w^{k,\Delta}_{\nu} + C^{\Delta}_{\mu\nu} w^{k,\,LG}_{\nu} + C^{\Delta}_{\mu\nu} w^{k,\,LG}_{\nu} \right) \tag{21}$$

Defining $\Delta^{L^*}_{\mu\nu} = C^{\forall \mathrm{L}}_{\mu\nu} w^{k,\Delta}_{\nu} + C^{\Delta}_{\mu\nu} w^{k,\,LG}_{\nu} + C^{\Delta}_{\mu\nu} w^{k,\,LG}_{\nu}$, and combining the effect of biasses on the output,

$$y^{k,L^*}_{\mu}\left(\mathcal{X}^{L^*}\right) = \sum_{\nu \in \mathcal{X}^{L^*}} C^{\forall \mathrm{L}}_{\mu\nu} w^{k,\forall \mathrm{L}}_{\nu} + \sum_{\nu \in \mathcal{X}^{L^*}} \Delta^{L^*}_{\mu\nu}$$

We can write the bias on the function output that does not change the function output for the inputs such that $|\mathcal{X}| = L^*$, but may change the output for inputs with other L, as

$$\sum_{\nu \in \mathcal{X}} \Delta^{L^*}_{\mu\nu}.$$

To illustrate this point, consider $\Delta_{\mu,\mu} = a$ and $\Delta_{\mu,\nu} = \frac{-a}{L^*-1}\ \forall\ \nu \neq \mu$, seeing that Eq. 21 is written $\forall\ \mu \in \mathcal{X}$,

$$\sum_{\nu \in \mathcal{X}^{L^*}} \Delta^{L^*}_{\mu\nu} = \Delta^{L^*}_{\mu\nu} + \sum_{\substack{\nu \in \mathcal{X}^{L^*} \\ \nu \neq \mu}} \Delta^{L^*}_{\mu\nu} = a - \sum_{\substack{\nu \in \mathcal{X}^{L^*} \\ \nu \neq \mu}} \frac{a}{L^*-1} = 0.$$

However, when we feed an input $\mathcal{X}^L \sim \mathcal{D}^L$ that $L \neq L^*$, we get

$$\begin{aligned}
y^{k,L^*}_{\mu}\left(\mathcal{X}^{L}\right) &= \sum_{\nu \in \mathcal{X}^{L}} C^{\forall \mathrm{L}}_{\mu\nu} w^{k,\forall \mathrm{L}}_{\nu} + \sum_{\nu \in \mathcal{X}^{L}} \Delta^{L^*}_{\mu\nu} \\
&= y^{k}_{\mu} + \sum_{\nu \in \mathcal{X}^{L}} \Delta^{L^*}_{\mu\nu} \\
&= y^{k}_{\mu} + \Delta^{L^*}_{\mu\nu} + \sum_{\substack{\nu \in \mathcal{X}^{L} \\ \nu \neq \mu}} \Delta^{L^*}_{\mu\nu} \\
&= y^{k}_{\mu} + a - \sum_{\substack{\nu \in \mathcal{X}^{L} \\ \nu \neq \mu}} \frac{a}{L^*-1} \\
&= y^{k}_{\mu} + a\frac{L^*-L}{L^*-1}
\end{aligned}$$

Proof of Theorem 4.10. To generalize to any length we should make sure that the population distribution for any length L^* does not allow such a bias, that is, $\Delta^{L^*}_{\mu\nu} = 0\,, \forall\ \mu, \nu$. From the previous discussion, for the set of parameters that generalize to population distribution $\mathcal{D}^{L^*}$ we know that,

$$\sum_{\nu \in \mathcal{X}^{L^*}} \Delta^{L^*}_{\mu\nu} = 0\,,\ \ \forall\ \mathcal{X}^{L^*} \sim \mathcal{D}^{L^*} \tag{22}$$

For each μ, we can think Eq. 22 as a linear system of equations. Defining, $\delta^{L^*}_{\mu} = \begin{bmatrix} \Delta^{L^*}_{\mu\alpha} & \Delta^{L^*}_{\mu\beta} & \Delta^{L^*}_{\mu\gamma} & \dots \end{bmatrix}^{\top} \in \mathbb{R}^N$, and an infinite sample extension of $\mathbf{S}_{\mathcal{B}_\mu}$ that is

$$\mathbf{S}^{L^*}_{\mathcal{B}^{\infty}_{\mu}} = \begin{bmatrix} \dots & \mathbf{s}^{L^*} & \dots \end{bmatrix}^{\top}_{s_\nu \neq 0}, \tag{23}$$

where $\mathbf{S}^{L^*}_{\mathcal{B}^{\infty}_{\mu}}$ is such a matrix that each row of it sums to L^*, Eq. 22 can be written as

$$\mathbf{S}_{\mathcal{B}^{\infty}_{\mu}} \delta^{\mu} = 0, \forall\ \ \mu \in \mathcal{S}.$$

If $\mathbf{S}_{\mathcal{B}^{\infty}_{\mu}}$ is full column rank for all μ, which is satisfied by Assumption 4.2, then $\mathbf{\Delta} = 0$. Thus, it generalizes to any length. □

Remark D.2. Seein that Assumption 4.2 inherently encompasses "$\mathbf{S}_{\mathcal{B}^{\infty}_{\mu}}$ is full column rank", it may seem that test generalization would always lead to length generalization, without any assumption. However, it is important to note that Assumption 4.2 is not the minimal requirement for ensuring test generalization.

Remark D.3. For clarity, let us look at an example class of tasks that for which $\mathbf{S}^{L^*}_{\mathcal{B}^\infty_\mu}$ is not full column rank. If there is a constraint on the distribution $\mathcal{X} \sim \mathcal{D}^{L^*}$ that the elements within $\mathcal{X}$ tuple are unique, than $\mathbf{S}^{L^*}_{\mathcal{B}^\infty_\mu}$ is not full rank since the column of $\mathbf{S}^{L^*}_{\mathcal{B}^\infty_\mu}$ that corresponds to element μ will be all 1 vector. Which is means, that there are some possible parameters that ensure zero error on $\mathcal{D}^{L^*}$ but does not generalize to $\mathcal{D}^L$ for any L.

Remark D.4. Under the realizability assumption, skip connections do not affect the values of the residues $\mathbf{D}^{(n)}$, allowing the same proof on generalization to apply in the skip-connected scenario.

Corollary D.5. *Defining*

$$C_{\mu\nu} = \mathbf{x}^\top(\mu)\,\mathbf{C}\mathbf{x}(\nu)\,,$$
$$W_{\nu k} = \mathbf{x}^\top(\nu)\,\mathbf{W}_{:,k},$$

from the Proof D.2, we see that the generalizing $\mathbf{C}, \mathbf{W}^V$ *satisfy,*

$$C_{\mu\nu} W^V_{\nu k} = C^{\forall \mathrm{L}}_{\mu\nu} W^{V,\forall \mathrm{L}}_{\nu k}$$

E. HyperFeatureAttention

E.1. Motivation

The preceding discussions in (Appendix B.1) on value functions that depend on single coordinates, both coordinates, and both coordinates with nonidentical agents reveal a fundamental pattern: as the number of features describing agents increases, the embedding dimension sufficient to fully capture pairwise interactions grows *exponentially*. As a quick recap of representations:

- For a single coordinate (e.g., x), the embedding dimension is proportional to $|\mathcal{S}_x| = N$, the number of possible positions along the x-axis.
- When extending to two coordinates (x and y), the domain becomes $\mathcal{S} = \mathcal{S}_x \times \mathcal{S}_y$, resulting in an embedding dimension of $|\mathcal{S}| = N^2$, reflecting all possible 2D positions.
- Adding agent-specific policies (e.g., $\mathcal{S}_q = \{\mathrm{R}, \mathrm{U}\}$) introduces an additional multiplicative factor, so the domain expands to $\mathcal{S} = \mathcal{S}_q \times \mathcal{S}_x \times \mathcal{S}_y$, with $|\mathcal{S}| = 2N^2$.
- We can even add a new feature species, from a completely different nature, such as species $\in \{\mathrm{H}, \mathrm{P}\}$. Which would make the the domain size even larger

Now let us look at the colliding agents environment, with a slightly more complex example, to motivate the *novel* HyperFeatureAttention.

Non-Identical Agents Revisited. Consider L agents on a $\mathcal{S}_{\mathbf{r}} = [N] \times [N]$ grid, each with initial coordinates $\mathbf{r}_i = \begin{bmatrix} n_i^x & n_i^y \end{bmatrix} \in [N]^2$. The agents have identities from $\ell_i \in \mathcal{S}_{\mathrm{spcs}} = \{\mathrm{H}, \mathrm{P}\}$. The Hs get $+1$ reward if they catch (collide with) a P, but the Ps get -1 reward when they are caught. The agents also have fixed policies $q_i \in \mathcal{S}_q = \{\mathrm{R}, \mathrm{U}\}$, that agents with R policy always moves to right. From, our earlier picture the value functions for the agents can be written as

$$V_i \;=\; \sum_{\substack{j \in [L] \\ j \neq i}} f\big(n_i^x, n_i^y, q_i, \ell_i, n_j^x, n_j^y, q_j, \ell_j\big)\; w_{n_j^x, n_j^y, q_j, \ell_j}. \tag{24}$$

With our earlier discussion (Appendix B), we know that a linear self-attention needs $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{S}_{\mathrm{spcs}}|^2 |\mathcal{S}_q|^2)$ parameters to represent such functions, which is exponential in the number of features.

After a careful thinking on each case in our experiment, one can write the value functions as

$$V_i \;=\; \sum_{j \in [L]:\, j \neq i} \Big\{ \mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\}\, \mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{U}\}\, \mathbb{I}\big\{n_i^x - n_j^x \approx n_i^y - n_j^y\big\}$$

$$
\begin{aligned}
&+\mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\}\,\mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{U}\}\,\mathbb{I}\left\{n_i^x - n_j^x \approx n_i^y - n_j^y\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\}\,\mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{R}\}\,\mathbb{I}\left\{n_i^x - n_j^x \approx n_i^y - n_j^y\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\}\,\mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{R}\}\,\mathbb{I}\left\{n_i^x - n_j^x \approx n_i^y - n_j^y\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\}\,\mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{R}\}\,\mathbb{I}\left\{n_i^y \approx n_j^y\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\}\,\mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{R}\}\,\mathbb{I}\left\{n_i^y \approx n_j^y\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\}\,\mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{U}\}\,\mathbb{I}\left\{n_i^x \approx n_j^x\right\}\\
&+\mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\}\,\mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{U}\}\,\mathbb{I}\left\{n_i^x \approx n_j^x\right\}\Big\}\left(\mathbb{I}\{\ell_j = \mathrm{P}\} - \mathbb{I}\{\ell_j = \mathrm{H}\}\right).
\end{aligned}
$$

Defining

$$
\begin{aligned}
f_{a_1}^{h_1}(\ell_i,\, \ell_j) &= \mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\} + \mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\},\\
f_{a_2}^{h_1}(q_i,\, q_j) &= \mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{U}\} + \mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{R}\},\\
f_{a_3}^{h_1}(\mathbf{r}_i,\, \mathbf{r}_j) &= \mathbb{I}\left\{n_i^x - n_j^x \approx n_i^y - n_j^y\right\},\\
f_{a_1}^{h_2}(\ell_i,\, \ell_j) &= \mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\} + \mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\},\\
f_{a_2}^{h_2}(q_i,\, q_j) &= \mathbb{I}\{q_i = \mathrm{R},\, q_j = \mathrm{R}\},\\
f_{a_3}^{h_2}\left(n_i^y,\, n_j^y\right) &= \mathbb{I}\left\{n_i^y \approx n_j^y\right\},\\
f_{a_1}^{h_3}(\ell_i,\, \ell_j) &= \mathbb{I}\{\ell_i = \mathrm{P},\, \ell_j = \mathrm{H}\} + \mathbb{I}\{\ell_i = \mathrm{H},\, \ell_j = \mathrm{P}\},\\
f_{a_2}^{h_3}(q_i,\, q_j) &= \mathbb{I}\{q_i = \mathrm{U},\, q_j = \mathrm{U}\},\\
f_{a_3}^{h_3}\left(n_i^x,\, n_j^x\right) &= \mathbb{I}\left\{n_i^x \approx n_j^x\right\},\\
w_{a_j}^{h_i}(\ell_j) &= \mathbb{I}\{\ell_j = \mathrm{P}\} - \mathbb{I}\{\ell_j = \mathrm{H}\},
\end{aligned}
$$

and $\mathcal{H} = \{h_1, h_2, h_3\}$, $\mathcal{H} = \{a_1, a_2, a_3\}$ the same equation can be organized into

$$
V_i \;=\; \sum_{h \in \mathcal{H}} \left[\sum_{j \in [L]} \Big\{ \prod_{a \in \mathcal{A}} f_a^h\left(\phi_{a,i}^h, \theta_{a,j}^h\right) \Big\} \Big\{ \prod_{a \in \mathcal{A}} w_a^h\left(\gamma_{a,j}^h\right) \Big\} \right], \tag{25}
$$

Where $\phi_{a,i}^h$, $\theta_{a,j}^h$ $\gamma_{a,j}^h$ are the corresponding features picked by the corresponding functions. Owing to our discussion from Appendix B, we can easily conclude that the functions $f_{a_1}^h$ can be represented *exactly* with a attention score matrix of $\mathbf{C}_{a_1}^h \in \mathbb{R}^{|\mathcal{S}_{\mathrm{spcs}}| \times |\mathcal{S}_{\mathrm{spcs}}|}$ similarly for $f_{a_2}^h$ we need $\mathbf{C}_{a_2}^h \in \mathbb{R}^{|\mathcal{S}_q| \times |\mathcal{S}_q|}$ and for $f_{a_3}^h$ we need $\mathbf{C}_{a_3}^h \in \mathbb{R}^{|\mathcal{S}_{\mathbf{r}}| \times |\mathcal{S}_{\mathbf{r}}|}$. As a result total number of parameters required is $\mathcal{O}(|\mathcal{S}_{\mathrm{spcs}}|^2 + |\mathcal{S}_q|^2 + |\mathcal{S}_{\mathbf{r}}|^2)$ which is linear in the number of features, much better than linear self-attention which was exponential. You can calculate the exact number of parameters instead of $\mathcal{O}$ versions. For N=360, $\approx 10^6$ parameters is sufficient for HyperFeatureAttention, while self-attention requires$\approx 2 \cdot 10^7$ parameters.

Implications for Attention Mechanisms. While the linear self-attention mechanism discussed in Theorem 3.1 can represent pairwise interactions exactly, its parameter requirements also scale with the embedding dimension d. This limits its applicability in cases where the number of features (or their cardinality) is very large. For instance, in scenarios with additional discrete features such as temporal information, behavioral categories, or hierarchical roles, the exponential growth in $|\mathcal{S}|$ quickly becomes a bottleneck.

The exponential growth observed here highlights the need for alternative attention mechanisms that can efficiently handle high-dimensional domains without explicitly embedding all feature combinations. This sets the stage for the introduction of a novel attention module *HyperFeatureAttention*, designed specifically to address exponential embedding growth while preserving the ability to model complex interactions across diverse features.

Also, one may concern that in practice we use layers of multi-head attention, which may possibly express Eq. 3, without exponential embedding dimension. However, we showed in Theorem E.3 that even two layer multihead linear self-attention cannot express (3).

E.2. HyperFeatureAttention Definition

The non-identical agents revised discussion was just a toy example for setting the stage for our novel *HyperFeatureAttention* model. Let us first generalize the discussion. We have Φ as the set of all features and $M = |\Phi|$ as total number of features in our setting that are distinct in nature. For feature ϕ, denote its domain $\mathcal{S}_\phi$, domain size $N_\phi = |\mathcal{S}_\phi|$, and allocated embedding size d_ϕ. From the previous discussion, to represent any function of the form Eq (25) the total embedding dimension for linear self-attention grows exponentially as $d = \prod_{\phi\in\Phi} |\mathcal{S}_\phi|$ and the corresponding number of parameters $\mathcal{O}\left(\prod_{\phi\in\Phi} |\mathcal{S}_\phi|^2\right)$. However, using a function of the form,

$$\mathbf{HFA}^{\text{lin}}(\mathbf{X}) = \sum_{h\in\mathcal{H}} \left[\left\{ \prod_{a\in\mathcal{A}}^{\odot} \mathbf{X}\mathbf{C}^{(h,a)}\mathbf{X}^\top \right\} \left\{ \prod_{a\in\mathcal{A}}^{\odot} \mathbf{X}\mathbf{W}^{V,(h,a)} \right\} \right],$$

we only need embedding dimension of $d = \sum_{\phi\in\Phi} |\mathcal{S}_\phi|$ and the corresponding number of parameters grows linearly in the number of features $\mathcal{O}(M)$.

A Short Note on Approximate Embedding. This note is mostly copied from the main text. Although the following discussion is somewhat *perpendicular* to our main focus on how entities (or features) interact, we include it briefly for completeness. In principle, self-attention may require an embedding dimension exponential in the number of features, but in practice, one often leverages the Johnson–Lindenstrauss lemma: in high-dimensional spaces, random projections yield vectors that are *approximately* orthonormal with embedding dimension only linear in the number of features. Concretely, each feature ϕ typically contributes $\mathcal{O}(\log|\mathcal{S}_\phi|)$ dimensions rather than $\mathcal{O}(|\mathcal{S}_\phi|)$, so total embedding dimension becomes $d = \mathcal{O}(|\Phi|)$.[6] However, in HyperFeatureAttention module, this same Johnson-Lindenstrauss argument implies a dimension requirement of $\mathcal{O}(\log|\Phi|)$ (rather than $\mathcal{O}(|\Phi|)$). Hence, even though both methods rely on approximate embeddings, *the exponential gap remains*: standard self-attention requires dimension linear in the number of features, whereas our module reduces it to logarithmic.

After all these motivations, we can now *formally* define the Multihead *HyperFeature* Attention. First, copy how Multihead Self Attention is defined in (Vaswani et al., 2023) here for comparison.

Definition E.1 (Multihead Self-Attention)**.** Let $\mathbf{X} \in \mathbb{R}^{T\times d}$ denote the input sequence of T tokens, where d is the embedding dimension. Multihead self-attention computes a sequence of contextualized embeddings as follows:

$$\text{head}^h = \text{Softmax}\left(\mathbf{Q}^h(\mathbf{K}^h)^\top\right)\mathbf{V}^h, \quad \forall h \in \{1,\dots,H\}, \tag{26}$$

$$\text{Multihead}(\mathbf{X}) = \text{Concat}(\text{head}^1,\dots,\text{head}^H)\mathbf{W}^O, \tag{27}$$

where:

- $\mathbf{Q}^h = \mathbf{X}\mathbf{W}^{Q^h}$, $\mathbf{K}^h = \mathbf{X}\mathbf{W}^{K^h}$, and $\mathbf{V}^h = \mathbf{X}\mathbf{W}^{V^h}$ are the query, key, and value matrices for the h-th head, respectively.
- $d_h = \frac{d}{H}$ is the dimension of each attention head.
- $\mathbf{W}^{Q^h}, \mathbf{W}^{K^h}, \mathbf{W}^{V^h} \in \mathbb{R}^{d\times d_h}$ and $\mathbf{W}^O \in \mathbb{R}^{d\times d}$ are learnable weight matrices.
- $\text{Softmax}(\cdot)$ is applied along the last dimension.

[6]Of course, some features are more crucial than others and thus allocated more dimensions. In addition, while simpler or ordered features (e.g. spatial coordinates) can often be embedded in far fewer dimensions. For example, spatial x-coordinates, are intrinsically one-dimensional, allowing them to be embedded into a much lower-dimensional vector space than $|\mathcal{S}_x|$. This is because x-coordinates form an ordered set with a linear relationship between positions. Conversely, features like the nucleotide bases A, G, T, and C lack such linear relationships —there is no linear relationship between ”adenineness” or ”guanineness”— and therefore, these features tend to occupy a 4-dimensional vector space. Some features, such as color, lie somewhere in between. Scientifically, a single frequency (e.g., 400–484 THz) is sufficient to specify the color ”rose red,” but in practical encodings like RGB, it is represented by three integers (e.g., $[255, 3, 62]$). However, in the language it is even more complex. As explained in (Jensen, 2022), the qualities of the color red, such as its warmth or contrast with green, cannot be deduced solely from its frequency as an electromagnetic wave. Even less can its emotional associations —such as romance or warmth— be reduced to a property of the wave. Instead, these properties emerge from the collective processes generated by light absorbed through the eyes, which trigger a hierarchy of neural processes in the brain. These processes ultimately lead to thoughts and emotions that we experience as color perception.

Definition E.2 (Multihead HyperFeatureAttention)**.** Let $\mathbf{X} \in \mathbb{R}^{T \times d}$ denote the input sequence of T tokens, where d is the embedding dimension.

$$\textbf{HFA}^h = \text{Softmax}\left(\prod_{a\in[A]}^{\odot} \mathbf{Q}^{(h,a)}(\mathbf{K}^{(h,a)})^\top\right) \prod_{a\in[A]}^{\odot} \mathbf{V}^{(h,a)}, \quad \forall h \in \{1,\dots,H\}, \tag{28}$$

$$\textbf{MHHFA}(\mathbf{X}) = \text{Concat}(\textbf{HFA}^1,\dots,\textbf{HFA}^H)\mathbf{W}^O, \tag{29}$$

where:

- $\mathbf{Q}^{(h,a)} = \mathbf{X}\mathbf{W}^{Q^{(h,a)}}$, $\mathbf{K}^{(h,a)} = \mathbf{X}\mathbf{W}^{K^{(h,a)}}$, and $\mathbf{V}^{(h,a)} = \mathbf{X}\mathbf{W}^{V^{(h,a)}}$ are the query, key, and value matrices for the h-th head a-th attention score, respectively.
- $d_h = \frac{d}{H}$ or $d_h = \frac{d}{HA}$, as matter of preference, is the dimension of each attention head.
- $\mathbf{W}^{Q^h}, \mathbf{W}^{K^h}, \mathbf{W}^{V^h} \in \mathbb{R}^{d \times d_h}$ and $\mathbf{W}^O \in \mathbb{R}^{d\times d}$ are learnable weight matrices.
- $\prod^{\odot}$ represents *Hadamard* product of matrices and $[A] = \{1,\dots,A\}$
- $\text{Softmax}(\cdot)$ is applied along the last dimension.

One can easily incorporate rotationary positional embedding into this module by applying the corresponding rotation matrices $\mathbf{R}$ as $\mathbf{R}\mathbf{Q}^{(h,a)}$ and $\mathbf{R}\mathbf{K}^{(h,a)}$ to keys and queries just as they are explained in (Su et al., 2023).

Theorem E.3 (Limitations of Two-Layer Multihead Linear Self-Attention)**.** *Two-layer multihead linear self-attention (2LMH) cannot represent factorized cross-feature interaction functions of the form*

$$\sum_j f^{(1)}(a_i,a_j)f^{(2)}(b_i,b_j), \quad \textit{or} \quad \sum_j f^{(1)}(a_i,b_j)f^{(2)}(b_i,a_j), \quad \textit{or similar variants,} \tag{30}$$

where the token embedding is defined as $\mathbf{x}_i = \text{concat}(a_i, b_i)$.

Proof. Consider a two-layer multihead linear self-attention (2LMH) model. Each layer consists of H attention heads, where the output of each head is computed as:

$$\text{head}_i^{(h)} = \sum_{j=1}^{L} \big(\mathbf{x}_i^\top \mathbf{C}^{(h)} \mathbf{x}_j\big)\mathbf{w}^{(h)}(\mathbf{x}_j), \quad \text{for } m = 1,\dots,H,$$

with $\mathbf{C}^{(h)} \in \mathbb{R}^{d\times d}$ as the attention matrix for head h, and $\mathbf{w}^{(h)}(\cdot)$ as a linear map. The outputs of the heads are concatenated and optionally projected using a weight matrix $\mathbf{W}^O$. For a single-layer multihead attention, the overall output for token i is a linear combination of bilinear terms of the form:

$$\text{Multihead}_i = \sum_{h=1}^{M}\sum_{j=1}^{L} \big(\mathbf{x}_i^\top \mathbf{C}^{(h)} \mathbf{x}_j\big)\mathbf{v}^{(h)}(\mathbf{x}_j).$$

In a two-layer model, the first layer computes:

$$\mathbf{h}_i = \text{Combine}\Big(\sum_{j=1}^{L} \big(\mathbf{x}_i^\top \mathbf{C}^{(h)} \mathbf{x}_j\big)\mathbf{v}^{(h)}(\mathbf{x}_j),\ h = 1,\dots,H_1\Big),$$

and passes $\{\mathbf{h}_i\}$ as input to the second layer. The second layer then computes:

$$\text{Head}_i^{(p)} = \sum_{k=1}^{L} \big(\mathbf{h}_i^\top \mathbf{C}^{(p)} \mathbf{h}_k\big)\mathbf{v}^{(p)}(\mathbf{h}_k), \quad \text{for } p = 1,\dots,H_2.$$

By definition, $\mathbf{h}_i$ is a linear combination of sums of bilinear terms in $\mathbf{x}_i$ and $\mathbf{x}_j$. Therefore, $\mathbf{h}_i$ remains a sum of bilinear expressions across $\{\mathbf{x}_i, \mathbf{x}_j\}$.

The second layer operates on $\mathbf{h}_i$ and computes terms of the form $\mathbf{h}_i^\top \mathbf{C}^{(p)} \mathbf{h}_k$. Since $\mathbf{h}_i$ itself is bilinear, this results in expressions that are *multi-bilinear* in $\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k$. However, it does not introduce multiplicative interactions between independent subsets of features (e.g., a_i, a_j vs. b_i, b_j).

Factorization is absent: The target function $\sum_j f^{(1)}(a_i, a_j) f^{(2)}(b_i, b_j)$ requires the output to be a product of two independent terms: one depending solely on (a_i, a_j) and the other on (b_i, b_j). Multihead attention combines bilinear terms additively, not multiplicatively, so it cannot achieve this factorization.

Suppose $f^{(1)}$ and $f^{(2)}$ are chosen such that $f^{(1)}(a_i, a_j)$ is orthogonal to $f^{(2)}(b_i, b_j)$. In this case, any additive combination of bilinear terms cannot approximate the product $f^{(1)}(a_i, a_j) f^{(2)}(b_i, b_j)$, regardless of how many layers or heads are used.

Even with two layers of multihead linear self-attention, the mechanism remains additive and cannot express functions requiring multiplicative factorization of independent feature interactions. This limitation highlights the inability of 2LMH to represent cross-feature factorized interactions such as $\sum_j f^{(1)}(a_i, a_j) f^{(2)}(b_i, b_j)$ or its variants. $\square$

F. HyperAttention

F.1. Defining HyperAttention

Generalizing the Definition 6.1 to order n,

$$\begin{aligned}
A_{ij_1j_2\ldots j_{n-1}} &= \sum_{\alpha\zeta_1\zeta_2\ldots\zeta_{n-1}}^{d} C_{\alpha\zeta_1\zeta_2\ldots\zeta_{n-1}} X_{i\alpha} X_{j_1\zeta_1} X_{j_2\zeta_2} \ldots X_{j_{n-1}\zeta_{n-1}} \\
V_{j_1j_2\ldots j_{n-1}\tau} &= \sum_{\xi_1\xi_2\ldots\xi_{n-1}}^{d} X_{j_1\xi_1} X_{j_2\xi_2} \ldots X_{j_{n-1}\xi_{n-1}} W^V_{\xi_1\xi_2\ldots\xi_{n-1}\tau} \\
\mathrm{HA}^{\mathrm{lin}}_{i\tau}(\mathbf{X}) &= \sum_{j_1j_2\ldots j_{n-1}}^{L} A_{ij_1j_2\ldots j_{n-1}} V_{j_1j_2\ldots j_{n-1}\tau},
\end{aligned}$$

where $\mathbf{C}$ and $\mathbf{W}^V \in \mathbb{R}^{d^n}$. Similar to how self-attention implements low rank approximation for data efficiency, i.e,

$$\mathbf{C} = \mathbf{W}^Q \left(\mathbf{W}^K\right)^\top,$$

where $\mathbf{W}^Q$ and $\mathbf{W}^K \in \mathbb{R}^{d\times R}$ are chosen low rank $R < d$, we can have low rank approximation of HyperAttention as

$$\begin{aligned}
C_{\alpha\zeta_1\zeta_2\ldots\zeta_{n-1}} &= \sum_{\sigma}^{R} W^Q_{\alpha\sigma} W^{K^1}_{\zeta_1\sigma} W^{K^2}_{\zeta_2\sigma} \ldots W^{K^{n-1}}_{\zeta_{n-1}\sigma} \\
W^V_{\xi_1\xi_2\ldots\xi_{n-1}\tau} &= \sum_{\sigma}^{R} W^{V^1}_{\xi_1\sigma} W^{V^2}_{\xi_2\sigma} \ldots W^{V^{n-1}}_{\xi_{n-1}\sigma} W^{V^n}_{\tau\sigma},
\end{aligned}$$

where each $\mathbf{W}^Q$, $\mathbf{W}^{K^i}$, $\mathbf{W}^{V^i} \in \mathbb{R}^{d\times R}$ and $R < d$. Finally adding the non-linearity the full definition becomes,

Definition F.1 (HyperAttention). Let $\mathbf{X} \in \mathbb{R}^{T\times d}$ be the input sequence of T tokens, where d is the embedding dimension. For n-th order attention, define:

$$\begin{aligned}
\mathbf{Q} &= \mathbf{X}\mathbf{W}^Q \in \mathbb{R}^{T\times R}, \\
\mathbf{K}^m &= \mathbf{X}\mathbf{W}^{K^m} \in \mathbb{R}^{T\times R}, \quad \forall m \in \{1, \ldots, n-1\}, \\
\mathbf{V}^m &= \mathbf{X}\mathbf{W}^{V^m} \in \mathbb{R}^{T\times R}, \quad \forall m \in \{1, \ldots, n-1\}, \\
\mathbf{V}^n &= \mathbf{W}^{V^n} \in \mathbb{R}^{d\times R}.
\end{aligned}$$

Then, for each token index i and output dimension τ,

$$A_{i,j_1,\ldots,j_{n-1}} = \mathrm{Softmax}_{(j_1,\ldots,j_{n-1})}\left(\sum_{\sigma=1}^{R} Q_{i,\sigma} K^1_{j_1,\sigma} K^2_{j_2,\sigma} \ldots K^{n-1}_{j_{n-1},\sigma}\right),$$

$$V_{j_1,\ldots,j_{n-1},\tau} = \sum_{\sigma=1}^{R} V^1_{j_1,\sigma} V^2_{j_2,\sigma} \ldots V^{n-1}_{j_{n-1},\sigma} V^n_{\tau,\sigma}.$$

The HyperAttention output is computed as:

$$\mathrm{HA}^{\mathrm{softmax}}_{i,\tau}(\mathbf{X}) = \sum_{j_1,\ldots,j_{n-1}=1}^{T} A_{i,j_1,\ldots,j_{n-1}} V_{j_1,\ldots,j_{n-1},\tau}.$$

For multihead HyperAttention, we compute multiple heads indexed by $h \in \{1,\ldots,H\}$, each with independent learnable weights $\mathbf{W}^{Q^h}$, $\mathbf{W}^{K^{h,m}}$, $\mathbf{W}^{V^{h,m}}$, and $\mathbf{W}^{V^{h,n}}$. The multihead HyperAttention output is given by:

$$\mathrm{HyperAttention}(\mathbf{X}) = \mathrm{Concat}\big(\mathrm{head}^1(\mathbf{X}),\ldots,\mathrm{head}^H(\mathbf{X})\big)\mathbf{W}^O,$$

where each head h is computed as:

$$\mathrm{head}^h(\mathbf{X}) = \mathrm{HA}^{\mathrm{smax},h}(\mathbf{X}) \in \mathbb{R}^{T\times d_h},$$

with $d_h = d/H$, and $\mathbf{W}^O \in \mathbb{R}^{d\times d}$ is a learnable projection matrix.

F.2. Representation Abilities of HyperAttention

Theorem F.2. *A single layer of order n linear HyperAttention, with embedding dimension $d = N$, can represent any function of the form*

$$\mathbf{F}_i = \sum_{j_1,j_2,\ldots,j_{n-1}\in[L]} f\left(s(i), s(j_1), s(j_2), \ldots, s(j_{n-1}\right) w_{s(j_1),s(j_2),s(j_{n-1})}$$

for all elements in the sequence, i.e, $i \in [L]$.

Proof. Seeing that $d = N$, the embeddings can be orthonormal. Consequently, the same arguments in Proof B follows. □

F.2.1. Skip-Trigram Bug

Let us now illustrate how higher-order dependencies may arise with a more practical example over *skip-trigrams*.

Next-Token Prediction A common strategy for training language models is *next-token prediction*: given a sequence of tokens $\big(s(0), s(1), \ldots, s(L-1)\big)$, the model learns to predict the next token. In our setting, the label for training is $s(L-1)$. For simplicity, we focus on the final row of the softmax self-attention output (corresponding to $s(L-1)$). Concretely, the model produces a probability distribution l over the vocabulary, defined as:

$$l \;=\; \frac{\sum_{j\in[L]} \exp\!\big(f\left(s(i), s(j)\right)\big)\, w_{s(j)}}{\sum_{j\in[L]} \exp\!\big(f\left(s(i), s(j)\right)\big)}.$$

During inference, the next token $\hat{y}$ is then sampled from this distribution.

As shown by (Elhage et al., 2021), self-attention can learn *skip-trigrams*. For example, in a sentence fragment such as "…keep…in [ ]," the model might predict the next token "mind," completing the phrase as "…keep…in mind." In the context of our work, we can interpret this phenomenon in two steps: (1) self-attention identifies that "in" is influenced by "keep" (i.e., $f(\mathrm{in}, \mathrm{keep})$ is large), and (2) it leverages that influence to generate the token "mind."

However, as shown in (Elhage et al., 2021), the same mechanism that raises the probabilities of the correct skip-trigrams "…keep…in mind" and "…keep…at bay" also inadvertently increases the probabilities of the erroneous skip-trigrams "…keep…in bay" and "…keep…at mind." This phenomenon, known as the "skip-trigram bug," arises from how attention influences these completions. From our interaction perspective, increasing the probability of "$\ldots$ keep $\ldots$ in mind" can be done in two ways (1) Increasing $f(\text{in}, \text{keep})$, which unfortunately also boosts the probability of "$\ldots$ keep $\ldots$ in bay." (2) Modifying w_{keep} to bias the model more strongly towards "mind," which reduces the probability of "$\ldots$ keep $\ldots$ at bay." Either approach makes it challenging for the model to consistently prefer the correct completions without also amplifying incorrect ones, thereby explaining the skip-trigram bug.

Hyper-Attention for Avoiding Skip-Trigram Bugs. The crux of the skip-trigram bug is that a *single* self-attention head (or pairwise interaction) tries to capture the entire phrase "$\ldots$ keep $\ldots$ in mind" by boosting $f(\text{in}, \text{keep})$ alone. This inadvertently increases the probability of other completions like "$\ldots$ keep $\ldots$ in bay" whenever w_{keep} also points toward "bay." However, many real-world contexts contain additional tokens that disambiguate the correct completion. For instance, the sentence "*...keep the deadline in [ ]*" strongly suggests "*mind*" over "*bay*".

In our framework of *hyper-attention*, one can introduce a *ternary* interaction term

$$f\big(\text{in}, \text{keep}, \text{ deadline}\big)$$

that focuses specifically on the triplet {"in", "keep", "deadline"}, allowing the model to favor "mind" without simultaneously boosting "bay." Concretely, if we let

$$f\big(s(i),\ s(j),\ s(k)\big) \quad \text{and} \quad w_{\,s(j),\,s(k)}$$

govern three-way effects (rather than just pairwise $f(\text{in}, \text{keep})$), the probability of "mind" can be increased via a *higher-order* interaction $f(\text{in}, \text{keep}, \text{the\_deadline})$ specifically tailored to that context. In doing so, we need not raise *all* completions of "$\ldots$ keep $\ldots$ in [ ]," and thus avoid inadvertently increasing "$\ldots$ keep $\ldots$ in bay."

Hence, by modeling *triplet* or *higher-order* interactions, hyper-attention more flexibly captures context-specific phrases like "*keep the deadline in mind*," while suppressing incorrect ones like "*keep the deadline in bay*," mitigating the skip-trigram bug highlighted.

F.3. Efficient Strategy to Mitigate $\mathcal{O}(L^3)$ Computation Complexity

Recall the definition of HyperAttention (Definition F.1). For simplicity we focus on third order HyperAttention but the same arguments generalizes to any order. Lets first look at linear version.

$$\begin{aligned}
A_{i,j_1,j_2} &= \sum_{\sigma=1}^{R} Q_{i,\sigma} K^1_{j_1,\sigma} K^2_{j_2,\sigma}, \\
V_{j_1,j_2,\tau} &= \sum_{\sigma=1}^{R} V^1_{j_1,\sigma} V^2_{j_2,\sigma} V^3_{\tau,\sigma}, \\
\text{HA}^{\text{lin}}_{i\tau}(\mathbf{X}) &= \sum_{j_1 j_2}^{L} A_{i j_1 j_2} V_{j_1 j_2 \tau}.
\end{aligned}$$

where we **denote** (i, j, k)-th entry of a tensor $\mathbf{T}$ as T_{ijk} and $\mathbf{C}, \mathbf{W}^V \in \mathbb{R}^{d\times d\times d}$. Here, each equation has $\mathcal{O}(L^3 R)$ computational complexity, so the total calculation has $\mathcal{O}(L^3 R)$ complexity. We can write the same expression as

$$\text{HA}^{\text{lin}}_{i\tau}(\mathbf{X}) = \sum_{j_1 j_2}^{L} \Big\{ \sum_{\sigma=1}^{R} Q_{i,\sigma} K^1_{j_1,\sigma} K^2_{j_2,\sigma} \Big\} \Big\{ \sum_{\sigma=1}^{R} V^1_{j_1,\sigma} V^2_{j_2,\sigma} V^3_{\tau,\sigma} \Big\} \tag{31}$$

Changing the order of summations (take the summations over js first), its computational complexity can be reduced to $\mathcal{O}(LR^2)$. Generally $R \ll L$ because R simply corresponds to attention head dimension. Thus, computational complexity reduces significantly.

As for the softmax or general nonlinear version, we use techniques similar to those in (Katharopoulos et al., 2020; Choromanski et al., 2022). For general nonlinear case Eq.31 can be written as,

$$\text{HA}^{\text{lin}}_{i\tau}(\mathbf{X}) = \frac{\sum_{j_1 j_2}^{L} \text{sim}\Big(\sum_{\sigma=1}^{R} Q_{i,\sigma} K^1_{j_1,\sigma} K^2_{j_2,\sigma}\Big) \Big\{\sum_{\sigma=1}^{R} V^1_{j_1,\sigma} V^2_{j_2,\sigma} V^3_{\tau,\sigma}\Big\}}{\sum_{j_1 j_2}^{L} \text{sim}\Big(\sum_{\sigma=1}^{R} Q_{i,\sigma} K^1_{j_1,\sigma} K^2_{j_2,\sigma}\Big)},$$

where $\mathrm{sim}(.)$ is just classical non-linearities (for softmax it is exponential function). (Alman & Song, 2023) show that this can be approximated with a function of the form

$$\mathrm{HA}^{\mathrm{lin}}_{i\tau}(\mathbf{X}) = \frac{\sum_{j_1 j_2}^{L}\left(\sum_{\sigma=1}^{R}\phi(Q_{i:})_\sigma \phi(K^1_{j_1:})_\sigma \phi(K^2_{j_2:})_\sigma\right)\left\{\sum_{\sigma=1}^{R} V^1_{j_1,\sigma}V^2_{j_2,\sigma}V^3_{\tau,\sigma}\right\}}{\sum_{j_1 j_2}^{L}\left(\sum_{\sigma=1}^{R}\phi(Q_{i:})_\sigma \phi(K^1_{j_1:})_\sigma \phi(K^2_{j_2:})_\sigma\right)},$$

if entries of the input matrices $\mathbf{Q}, \mathbf{K}$ are less than $o(\sqrt[3]{\log L})$. Consequently, the same summation order change trick applies.

G. Justification for Data Versatility Assumption

Assumption G.1 (Positive-Definite Covariance and Bounded Norm)**.** Let $\{\mathbf{a}_n\}_{n=1}^{B} \subset \mathbb{R}^N$ be i.i.d. random row vectors. Suppose there exist constants $\zeta_{\min} > 0$ and $M > 0$ such that:

(A1) **Positive-Definite Covariance:** The covariance matrix

$$\Sigma \;:=\; \mathrm{Cov}(\mathbf{a}_n) \;=\; \mathbb{E}\big[(\mathbf{a}_n - \mathbb{E}[\mathbf{a}_n])(\mathbf{a}_n - \mathbb{E}[\mathbf{a}_n])^\top\big] \quad \text{satisfies} \quad \Sigma \succeq \zeta_{\min}^2\,\mathbf{I}.$$

That is, $\lambda_{\min}(\Sigma) \geq \zeta_{\min}^2 > 0$.

(A2) **Bounded Norm:** The centered vectors satisfy

$$\|\mathbf{a}_n - \mathbb{E}[\mathbf{a}_n]\|_2 \;\leq\; M \quad \text{almost surely.}$$

Theorem G.2 (Full Column Rank with High Probability)**.** *Under Assumption G.1, let*

$$\mathbf{A} \;=\; \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \vdots \\ \mathbf{a}_B \end{bmatrix} \;\in\; \mathbb{R}^{B\times N}.$$

Then there exist positive constants $c, \kappa > 0$ (depending on M, $\zeta_{\min}$, and N) such that for all sufficiently large B,

$$\mathbb{P}\Big[\mathrm{rank}(\mathbf{A}) < N\Big] \;\leq\; e^{-\kappa B}.$$

Equivalently, $\mathbf{A}$ is full column rank with probability at least $1 - e^{-\kappa B}$.

Proof. **Step 1: Center the rows.** Define $\mathbf{x}_n := \mathbf{a}_n - \mathbb{E}[\mathbf{a}_n]$, so that $\mathbb{E}[\mathbf{x}_n] = \mathbf{0}$ and

$$\mathrm{Cov}(\mathbf{x}_n) \;=\; \mathbb{E}[\mathbf{x}_n\mathbf{x}_n^\top] \;=\; \Sigma.$$

Stack these centered rows into

$$\mathbf{X} \;=\; \begin{bmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_B^\top \end{bmatrix} \;\in\; \mathbb{R}^{B\times N}.$$

Since each row of $\mathbf{A}$ differs from $\mathbf{X}$ by a constant shift, $\mathrm{rank}(\mathbf{A}) = \mathrm{rank}(\mathbf{X})$. Hence it suffices to show $\mathbf{X}$ is full column rank with high probability.

Step 2: Expected Gram matrix lower bound. We have

$$\mathbf{X}^\top\mathbf{X} \;=\; \sum_{n=1}^{B} \mathbf{x}_n^\top\mathbf{x}_n \;=\; \sum_{n=1}^{B} \mathbf{x}_n\mathbf{x}_n^\top \quad \text{(summing rank-1 updates).}$$

Taking expectation,

$$\mathbb{E}[\mathbf{X}^\top \mathbf{X}] \;=\; \sum_{n=1}^{B} \mathbb{E}[\mathbf{x}_n \mathbf{x}_n^\top] \;=\; B\,\Sigma.$$

By Assumption G.1(A1), $\Sigma \succeq \zeta_{\min}^2 \mathbf{I}$, hence

$$\mathbb{E}[\mathbf{X}^\top \mathbf{X}] \;\succeq\; B\,\zeta_{\min}^2\,\mathbf{I}.$$

Step 3: Concentration via Matrix Bernstein. Define the centered matrix

$$\mathbf{Z}_n \;:=\; \mathbf{x}_n \mathbf{x}_n^\top - \Sigma.$$

Note $\mathbb{E}[\mathbf{Z}_n] = \mathbf{0}$. Summing,

$$\mathbf{X}^\top \mathbf{X} - \mathbb{E}[\mathbf{X}^\top \mathbf{X}] \;=\; \sum_{n=1}^{B} (\mathbf{x}_n \mathbf{x}_n^\top - \Sigma) \;=\; \sum_{n=1}^{B} \mathbf{Z}_n.$$

Each $\mathbf{Z}_n$ is bounded in operator norm since

$$\|\mathbf{x}_n \mathbf{x}_n^\top\|_{\mathrm{op}} \;=\; \|\mathbf{x}_n\|_2^2 \;\le\; M^2, \quad \|\Sigma\|_{\mathrm{op}} \;\le\; \|\Sigma\|_{\mathrm{F}} \;\text{(finite)}.$$

Thus for a constant $R \in \mathbb{R}$,

$$\|\mathbf{Z}_n\|_{\mathrm{op}} \le \|\mathbf{x}_n \mathbf{x}_n^\top\|_{\mathrm{op}} + \|\Sigma\|_{\mathrm{op}} = R.$$

In addition,

$$\|Z_n^2\|_{\mathrm{op}} \le \|Z_n\|_{\mathrm{op}}^2 \le R^2,$$
$$\left\| \sum_{n=1}^{B} \mathbb{E}\left[Z_n^2\right] \right\|_{\mathrm{op}} \le BR^2$$

Hence by a standard matrix Bernstein inequality (self-adjoint version) from (Tropp, 2014), there exist constant $\kappa > 0$ such that

$$\mathbb{P}\big[\|\mathbf{X}^\top \mathbf{X} - \mathbb{E}[\mathbf{X}^\top \mathbf{X}]\|_{\mathrm{op}} \;\ge\; \tfrac{1}{2} B\,\zeta_{\min}^2\big] = \mathbb{P}\left[\left\| \sum_{n=1}^{B} \mathbf{Z}_n \right\|_{\mathrm{op}} \;\ge\; \tfrac{1}{2} B\,\zeta_{\min}^2 \right] \;\le\; e^{-\kappa B}.$$

In other words, with high probability, $\mathbf{X}^\top \mathbf{X}$ stays within half its expected value in spectral norm.

Step 4: Weyl's inequality implies strict positivity. On this high-probability event,

$$\lambda_{\min}(\mathbf{X}^\top \mathbf{X}) \;\ge\; \lambda_{\min}\big(\mathbb{E}[\mathbf{X}^\top \mathbf{X}]\big) \;-\; \big\|\mathbf{X}^\top \mathbf{X} - \mathbb{E}[\mathbf{X}^\top \mathbf{X}]\big\|_{\mathrm{op}} \;\ge\; B\,\zeta_{\min}^2 \;-\; \tfrac{1}{2} B\,\zeta_{\min}^2 \;=\; \tfrac{1}{2} B\,\zeta_{\min}^2.$$

Hence $\mathbf{X}^\top \mathbf{X}$ is strictly positive-definite, implying $\mathrm{rank}(\mathbf{X}) = N$. Consequently, $\mathbf{A}$ is full column rank with probability at least $1 - e^{-\kappa B}$. □

Remark G.3 (Justification & Examples of Assumption G.1). In many natural data-generation processes, these assumptions hold:

(i) **Count Vectors from a Dictionary.** Suppose each sample $\mathcal{X}^{(n)}$ is a tuple of L elements drawn from a vocabulary $\mathcal{S}$ (of size N). The row vector $\mathbf{a}_n$ may represent counts $(s_\alpha^{(n)}, s_\beta^{(n)}, \dots)$ of how many times each element $\alpha, \beta, \dots$ appears. If we focus on the subset $\mathcal{B}_\mu$ of samples that *contain* μ, then $s_\mu^{(n)} \ge 1$, while the other $L-1$ slots of the sequence are drawn from $\mathcal{S} \setminus \{\mu\}$ according to some distribution.

(ii) **Positive-Definite Covariance.** When these $(L-1)$ "remaining" elements are distributed in a *non-degenerate* way (e.g., at least some variability in how the other vocabulary items appear), the resulting count vectors $\mathbf{a}_n$ will have a covariance Σ whose minimum eigenvalue is strictly positive. For instance, under a uniform choice of the $L-1$ positions among the $N-1$ possible elements, straightforward calculations show each coordinate has nonzero variance, and provided there are no perfect negative correlations, $\lambda_{\min}(\Sigma) > 0$.

(iii) **Bounded Norm.** Since $0 \leq s_\nu^{(n)} \leq L$ for each element $\nu \in \mathcal{S}$, the count vector $\mathbf{a}_n$ is trivially bounded by $\sqrt{N}\, L$ in Euclidean norm. Thus we can take $M = \sqrt{N}\, L$, satisfying Assumption G.1(A2).

(iv) **General Distributions.** Even more general scenarios (e.g., non-uniform sampling, correlated draws) satisfy the same assumptions, *provided* negative correlations are not too extreme to force Σ to have a zero eigenvalue. In practice, real-world data tends to have enough variability so that $\mathrm{Cov}(\mathbf{a}_n)$ is well-conditioned, meeting the requirement $\lambda_{\min}(\Sigma) \geq \zeta_{\min}^2$ for some $\zeta_{\min} > 0$.

H. Figures

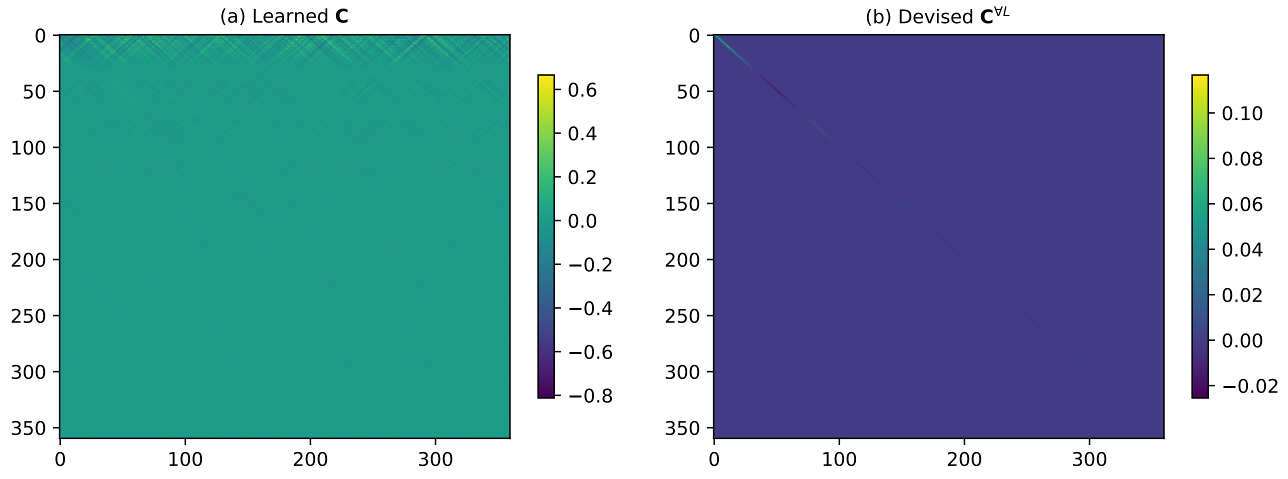

Figure 2. Comparison of learned vs. devised parameters for sinusoidal embedding: (a) Devised matrix $\mathbf{C}^{\forall L}$ showing the original devised structure, (b) Learned matrix $\mathbf{C}$ demonstrating the emergent but non-interpretable patterns. While visually distinct, both parameterizations lead to equivalent model behavior through different mathematical organizations.

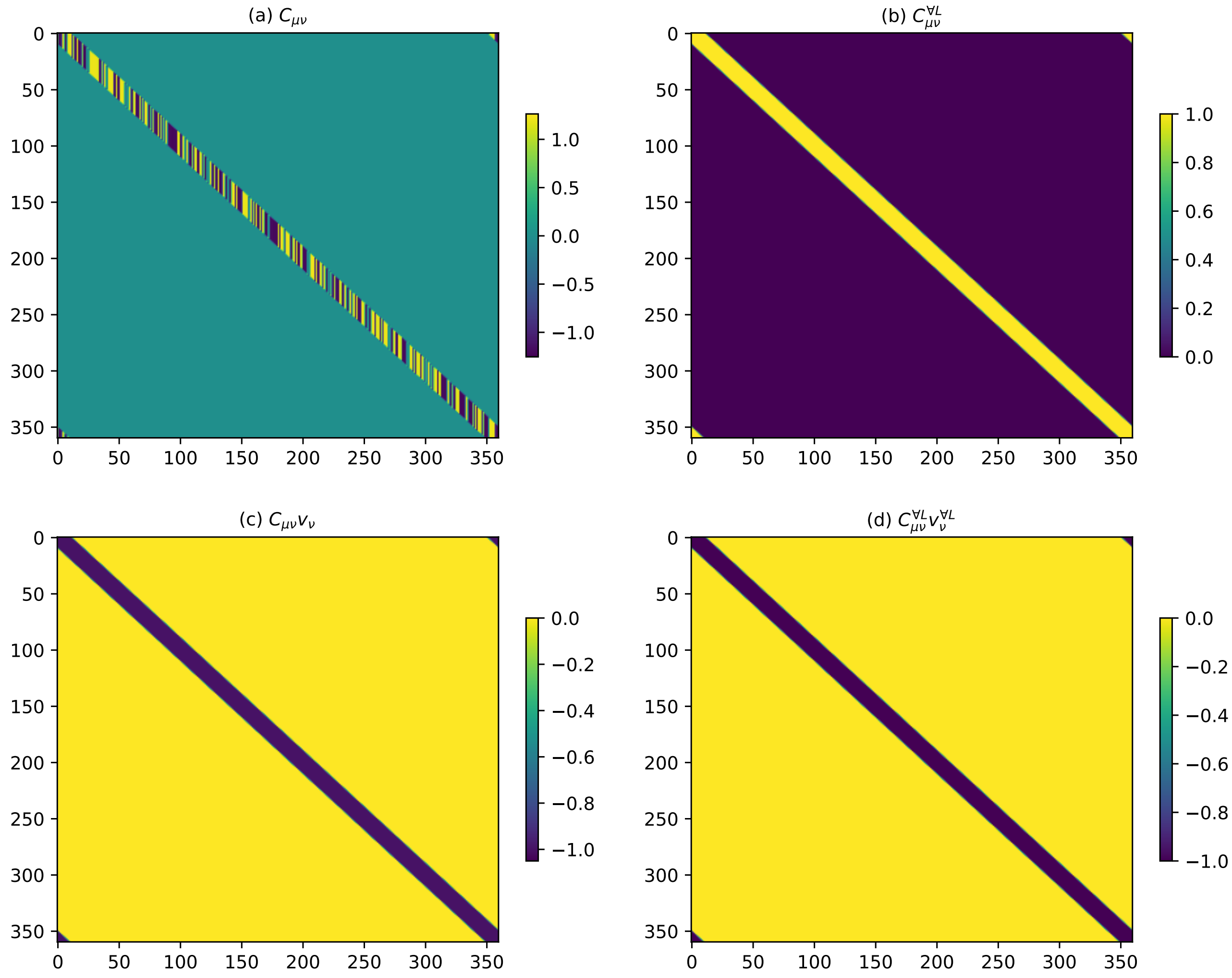

Figure 3. Equivalence of learned parameters with one-hot embedding in domain embedding base (here the parameters are already in the domain embedding base so we did not transfer them): (a) $C_{\mu\nu}$ learned parameters in domain embedding base, (b) $C^{\forall \mathrm{L}}$ devised parameters in domain embedding base, (c) $C_{\mu\nu}W_{\nu 0}$ the interesting matrix in domain embedding base, (d) Transformed $C^{\forall \mathrm{L}}_{\mu\nu}W^{\forall L}_{\nu 0}$ using original parameters. The main squared difference between (c) and (d) is $\mathcal{O}(10^{-5})$, demonstrating functional equivalence despite different parameter organizations.

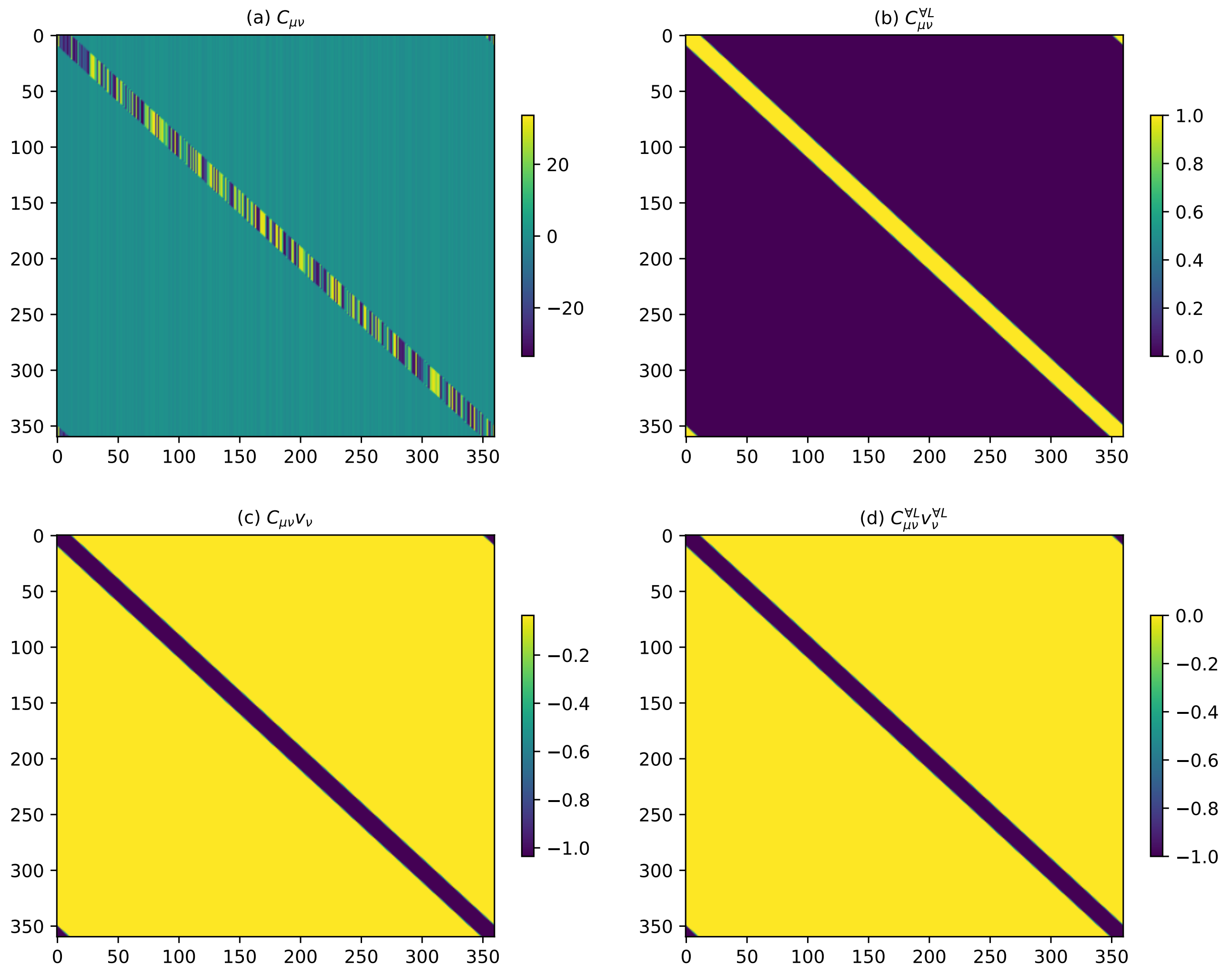

Figure 4. Equivalence of learned parameters with sinusoidal embedding in domain embedding base: (a) $C_{\mu\nu}$ learned parameters in domain embedding base, (b) $C^{\forall \mathrm{L}}$ devised parameters in domain embedding base, (c) $C_{\mu\nu}W_{\nu 0}$ the interesting matrix in domain embedding base, (d) Transformed $C^{\forall \mathrm{L}}_{\mu\nu}W^{\forall L}_{\nu 0}$ using original parameters. The main squared difference between (c) and (d) is $\mathcal{O}(10^{-5})$, demonstrating functional equivalence despite different parameter organizations. Additionally as a side note, comparing (b) with Fig. 2 (b), we observe the advantage of sinusoidal embeddings in terms of their parameter efficiency within the $\mathbf{C} = \mathbf{W}^{Q}\mathbf{W}^{K\top}$ matrix, particularly when relative positions are more important than absolute positions.