MECHANISTIC STUDY OF TRANSFORMER IN-CONTEXT LEARNING WITH CATEGORICAL OUTPUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

We study in-context learning (ICL) with Transformers for categorical outputs y_i, a setting largely unexplored compared to research on real-valued y_i. While attention-only Transformers can, in principle, perform functional gradient descent (GD) inference for real-valued outputs, we show that categorical y_i introduce a non-linearity in GD that attention-only models cannot capture. This reveals a crucial role for the Transformer's multi-layered perceptron (MLP) layers, which we show are generally necessary for categorical ICL. However, we also analyze conditions under which attention-only models can, surprisingly, still perform well. Since training for categorical ICL requires substantial data, we propose a sparse Transformer parametrization linked to functional GD. This model trains far more efficiently with minimal performance degradation compared to an unconstrained Transformer. Our sparse design proves particularly valuable for data-limited applications, which we demonstrate through the ICL analysis of human surgical procedures.

1 INTRODUCTION

The remarkable success of Transformers in language generation and other tasks (Vaswani et al., 2017; Radford et al., 2019; Devlin et al., 2019; Brown et al., 2020; Touvron et al., 2023; DeepSeek, 2025) has spurred intensive research into understanding their in-context learning (ICL) capabilities. A key insight from recent theoretical work is that when performing ICL with real-valued outputs, Transformers can implement functional gradient descent in the Transformer forward pass, through their attention mechanisms alone (von Oswald et al., 2023; Ahn et al., 2023; Zhang et al., 2023; Cheng et al., 2024). This explains why attention-only models (without MLP layers) suffice for tasks like linear and kernel regression.

However, a fundamental question arises when extending ICL to categorical outputs: *do the same principles apply?* Categorical ICL introduces softmax nonlinearities that are absent in the real-valued case. Specifically, optimal inference requires computing expectations of the form $\mathbb{E}[w|f(x)] = \sum_c w_c \cdot \text{softmax}(W_e^\top f(x))_c$, where w_c are learned category embeddings that define the columns of W_e. These nonlinear computations suggest a role for the MLP layers in conventional Transformers.

This theoretical analysis suggests that the MLP layers in Transformers — previously deemed unnecessary for ICL — should become essential for categorical outputs. The MLPs could approximate the required nonlinear expectations through their universal approximation capabilities. Based on this reasoning, one would expect attention-only Transformers to perform poorly on categorical ICL tasks.

An Empirical Puzzle. To test this theoretical prediction, we conducted experiments on sophisticated synthetic and real-world data, comparing attention-only Transformers against full Transformers (with MLPs) on categorical ICL. Contrary to our expectations, under some (*but not all*) settings, attention-only models performed nearly as well as their MLP-equipped counterparts.

Our Investigation. We provide both theoretical analysis and extensive experiments to understand this puzzle, and through this enhance understanding of the role of MLP layers in Transformer ICL, when y_i is categorical. Through detailed mechanistic investigation, we show that while exact categorical ICL inference is indeed nonlinear, a simple linear approximation *in certain settings* can suffice in practice. We derive conditions under which this approximation holds, showing exam-

ples of where attention-alone works well, and where it does not. Furthermore, we leverage these insights to develop sparse parameter Transformer initialization strategies that reduce training data requirements significantly.

1.1 Summary of Contributions

1. **Theoretical analysis** of categorical ICL revealing the nonlinear computations that distinguish it from real-valued ICL, and the mechanistic role this suggests for MLP layers.
2. **Empirical discovery** that in some settings attention-only Transformers perform well on categorical ICL despite theoretical predictions, and analysis showing when and why linear approximations to nonlinear expectations suffice.
3. **Practical insights** enabling sparse parameter initialization that dramatically reduces training data requirements without performance loss, validated on both synthetic and real-world surgical procedure datasets.
4. **New application domain** demonstrating categorical ICL for multi-question image analysis in surgical contexts, extending ICL beyond single classification tasks.

1.2 Related Work

Mechanistic analyses of in-context learning. Recent theoretical work has shown that Transformers can implement functional gradient descent (GD) in the forward pass when outputs are real-valued. For example, von Oswald et al. (2023); Ahn et al. (2023); Zhang et al. (2023) demonstrated that attention-only Transformers are sufficient for tasks such as linear regression, where GD updates can be expressed linearly. Cheng et al. (2024) extended this analysis to nonlinear real-valued functions, further supporting the connection between self-attention and GD in function space. These studies, however, focus almost exclusively on real-valued outcomes, leaving the categorical setting largely unexplored.

Categorical in-context learning. The categorical case introduces a fundamental difference: the required updates involve nonlinear expectations over category embeddings under a softmax distribution. The most closely related prior work is Wang et al. (2025), who studied categorical ICL but proposed new cross-attention mechanisms rather than analyzing the capabilities of the standard Transformer architecture. Our work connects directly to this line of research by showing that – even without architectural modifications – Transformers can approximate categorical ICL effectively, and we explain why for some problems attention-only models are sufficient.

Sparsity and initialization. Our GD-based sparse parameterization is related to the literature on sparse neural networks, particularly the Lottery Ticket Hypothesis (Frankle & Carbin, 2019; Brix et al., 2020), which argues that subnetworks with carefully chosen initializations can match the performance of dense models. In contrast to empirical pruning approaches, our sparsity patterns are derived directly from functional GD analysis, yielding an interpretable initialization strategy with orders-of-magnitude lower data requirements. This also connects to work on solution multiplicity in neural networks (Draxler et al., 2018; Garipov et al., 2018; Lee et al., 2019), which shows that different parameterizations can yield similar functional behavior.

Applications of categorical ICL. Most investigations of Transformer ICL focused on mechanistic understanding have considered simulated data. We demonstrate the utility of categorical ICL in real-world surgical video understanding, a domain where labeled data is scarce and categorical outcomes (e.g., instruments, actions, targets) are natural. This is among the first real applications of this technology.

2 Setup: In-context learning with Transformers

For categorical outcomes $y_i \in \{1, \ldots, C\}$, an embedding vector $w_c \in \mathbb{R}^{d'}$ is learned for each category $c = 1, \ldots, C$; embedding-vector dimension d' is a design choice, like in language models (Vaswani et al., 2017). For use with a Transformer, the N labeled samples (x_i, y_i) are encoded as $z_i = (x_i, w_{y_i})$, for covariates $x_i \in \mathbb{R}^d$. Sample $N+1$, which is the query, is encoded $z_{N+1} = (x_{N+1}, 0_{d'})$, where $0_{d'}$ is a d'-dimensional vector of zeros (y_{N+1} is of course unavailable as Transformer input, and it is what we seek to infer). The encoding of observed categorical outcomes by their corresponding learned embedding vector is as in language models (Vaswani et al., 2017), and is a natural consequence of the analysis in Section 4.

The input to the Transformer is

$$Z_0 = \begin{bmatrix} z_1 & z_2 & \dots & z_N & z_{N+1} \\ 0_s & 0_s & \dots & 0_s & 0_s \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & \dots & x_N & x_{N+1} \\ w_{y_1} & w_{y_2} & \dots & w_{y_N} & 0_{d'} \\ 0_s & 0_s & \dots & 0_s & 0_s \end{bmatrix} \in \mathbb{R}^{(d+d'+s)\times(N+1)} \quad (1)$$

where 0_s is a vector of s zeros, constituting s-dimensional "scratch space" that the Transformer may use (if needed) to perform or store computations. The use of scratch space by Transformers has been discussed previously in Akyurek et al. (2023).

Following conventional Transformer design (Vaswani et al., 2017), the attention mechanism for head $h \in \{1, \dots, H\}$ is

$$\text{Attn}_h(Z) = V_h Z M \cdot A(K_h Z, Q_h Z) \quad (2)$$

where V_h, K_h and Q_h are each $(d + d' + s) \times (d + d' + s)$ real matrices, and the function $A : \mathbb{R}^{(d+d'+s)\times(N+1)} \times \mathbb{R}^{(d+d'+s)\times(N+1)} \to \mathbb{R}^{(N+1)\times(N+1)}$ represents attention, with component (n, m) of the output of $A(\cdot)$ representing the attention between column n of $K_h Z$ and column m of $Q_h Z$ (Vaswani et al., 2017). Following Cheng et al. (2024), the mask $M = \begin{bmatrix} I_{N\times N} & 0_{N\times 1} \\ 0_{1\times N} & 0 \end{bmatrix}$ imposes that only labeled (contextual) data are keys and values. We will consider kernel-based attention as in Cheng et al. (2024), as well as traditional softmax-based attention (Vaswani et al., 2017). For H heads, the composite of attention from all heads is $\sum_{h=1}^{H} P_h \cdot \text{Attn}_h(Z)$, where $P_h \in \mathbb{R}^{(d+d'+s)\times(d+d'+s)}$.

We consider attention *blocks*, as in the original Transformer (Vaswani et al., 2017), which include multi-layered perceptron (MLP) layers and associated skip connections:

$$Z_{\ell+1} = \tilde{Z}_{\ell+1} + \text{MLP}_\ell(\tilde{Z}_{\ell+1}) \ , \qquad \tilde{Z}_{\ell+1} = Z_\ell + \sum_{h=1}^{H} P_{h,\ell} \cdot \text{Attn}_{h,\ell}(Z_\ell) \quad (3)$$

where $\text{MLP}_\ell(\tilde{Z}_{\ell+1})$ acts separately on each column of $\tilde{Z}_{\ell+1}$. For an L-layer Transformer, the updates in (3) are performed sequentially from $\ell = 0, \dots, L$, and Z_{L+1} is output from the last (Lth) attention block. The predicted $\hat{f}(x_{N+1})$ associated with the query is within column $N + 1$ of Z_{L+1}.

As the final element of the Transformer, the query output is modeled via softmax as

$$p(Y_{N+1} = c | X_{N+1} = x_{N+1}, \mathcal{C}) = \frac{\exp[w_c^\top \hat{f}(x_{N+1})]}{\sum_{c'=1}^{C} \exp[w_{c'}^\top \hat{f}(x_{N+1})]} \quad (4)$$

The Transformer parameters are learned by seeking to minimize the cross-entropy loss based on the model $p(Y_{N+1}|X_{N+1} = x_{N+1}, \mathcal{C})$, using a training set of contextual data $(x_i^{(m)}, y_i^{(m)})$ for $i = 1, \dots, N + 1$, for $m = 1, \dots, M$ example sets.

3 Initial Empirical Observations of ICL for Categorical y_i

3.1 Experiments on synthetic data, train and test data aligned

We consider simulated data introduced in Wang et al. (2025); we choose these data because they represent a challenging in-context classification problem, and they allow us to connect to recent relevant work. We subsequently show results based on two real-data scenarios.

The data are generated $p(Y = c|f(x)) = \exp[w_c^T f(x)] / \sum_{c'=1}^{C} \exp[w_{c'}^T f(x)]$, for $C = 25$ and $w_c \in \mathbb{R}^5$, where w_c represents the category-dependent embedding vectors used for data synthesis (hidden from the Transformer). For data synthesis, w_c are generated (once) randomly, with each matrix component drawn i.i.d. from $\mathcal{N}(0, 1)$. After W_e is so drawn, different contextual datasets consider a distinct function $f^{(m)}(x)$, where m represents the context index. To constitute $f^{(m)}(x)$, 5 categories are selected uniformly at random from the dictionary of $C = 25$ categories. Let $c^{(m)}(1), \dots, c^{(m)}(5)$ denote these categories for context l. We further randomly generate 5 respective "anchor positions," $\tilde{x}(1), \dots, \tilde{x}(5)$, each drawn i.i.d. from $\mathcal{N}(0_d, I_d)$, where $d = 10$ (for covariates $x \in \mathbb{R}^{10}$). The function for context m is represented as $f^{(m)}(x) = \lambda \sum_{k=1}^{5} w_{c(k)} \kappa_{RBF}[x - \tilde{x}(k); \sigma_\ell]$, where the RBF kernel parameter σ_m for component m is selected such that $\kappa_{RBF}[x - \tilde{x}(m); \sigma_m] = \exp(-\sigma_m^2 \|x - \tilde{x}_m\|_2)$ equals 0.1 at the center of the other kernel to which it is closest (in a Euclidean distance sense). Parameter $\lambda = 10$, selected so as to have category $c(m)$ be clearly most probable in the region of $\tilde{x}(m)$. Each contextual block considers $N = 50$ samples.

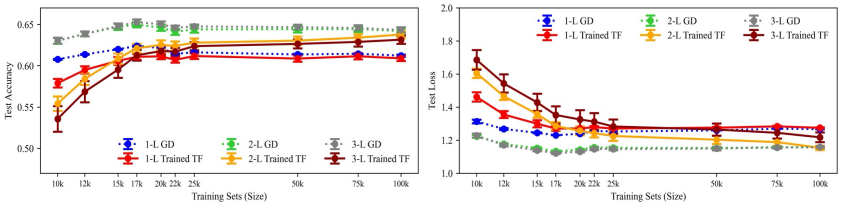

Figure 1: Classification accuracy (left) and cross-entropy loss (right) for the synthetic data, evaluated after training on separate *test* data, as a function of the number of contextual datasets M on which it was trained. The training data are $\{(x_i^{(m)}, y_i^{(m)})\}_{i=1,N+1}$, $m = 1, \ldots, M$, and the horizontal axis is M. Results are shown for the GD and Trained TF forms of the Transformer training, as a function of the number of layers (e.g., 3-L is a three-layer model). MLP layers follow each self-attention layer. Five random seeds were used to train each model, and the error bars (often small) reflect variance across these seeds.

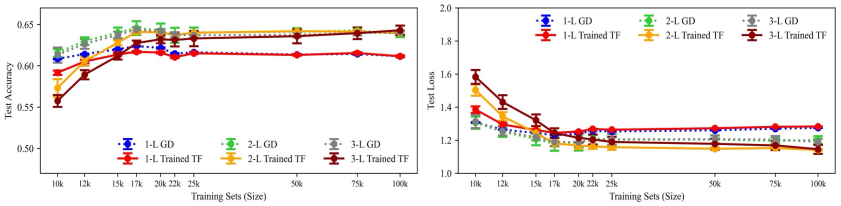

Figure 2: Results as in Figure 1, but here *no* MLP layers are present, and therefore the Transformer is attention-only.

3.2 Experiments on real data, mismatch of data used for training and testing

We also present our first experiment based on real data, for which there is a mismatch between the data used for training and testing. The covariates x_i for image i are features from a pre-trained (self-supervised) masked-autoencoder based vision Transformer (He et al., 2022; Dosovitskiy et al., 2020). The covariates are here $d = 768$, and we considered embedding vectors of dimension $d' = 5$. Each contextual block considered images from 5 label types, and the query was from one of these. A total of $N = 50$ contextual labeled samples are provided to the Transformer, plus a query, x_{N+1}.

The Transformer was trained using data from Caltech256 (Griffin et al., 2007), and here we show test results on data from TinyImageNet dataset (Le & Yang, 2015). Given the relatively large quantity of data needed to train the Trained TF versions of the Transformer, we present all results in Figure 3 for the sparse, GD-based versions of the Transformers, with and without MLP layers. In Figure 3 we also show results from the cross-attention (CA) model of Wang et al. (2025), which performs exact functional GD, but uses a model design that is inconsistent with the traditional Transformer. We also considered test experiments on the distinct DomainNet dataset (Peng et al., 2019), and those results are also shown in Appendix C (see Figure 7 there).

Concerning training on Caltech256 (Griffin et al., 2007), there are 256 object categories, with between 31–80 samples for each. The TinyImageNet (Le & Yang, 2015) dataset used for testing contains 200 classes with 500 samples per category. We also tested on DomainNet (Peng et al., 2019) (details in Appendix C), focusing on five visual domains: ClipArt (clipart illustrations), InfoGraph (infographic images), QuickDraw (hand-drawn sketches from Google's Quick Draw game), Real (photographs), and Sketch (artistic sketches), each spanning across 345 categories.

As in the experiment with synthetic data, the Transformer is given contextual data of size $N = 50$, with 10 samples from each of 5 classes (labels), selected uniformly at random (without replacement) from the train/test dataset. The query x_{N+1} is from among the five classes. The number of categories in this experiment is always $C = 5$, so the learned embedding vectors are near orthogonal (there is no fixed meaning in the labels between contexts).

3.3 Summary of Initial Empirical findings

In our experiments, we have implemented attention based on a radial basis function (RBF) kernel, and based on the traditional softmax-based attention widely used in Transformers. In all of our experiments, we find that these two types of attention mechanisms yield similar results. However, with unnormalized vectors, the ℓ_2 norm required in the RBF kernel is considerably more expensive computationally than the single inner product connected with softmax-based attention. In Figures 1-3 we present results based on softmax attention.

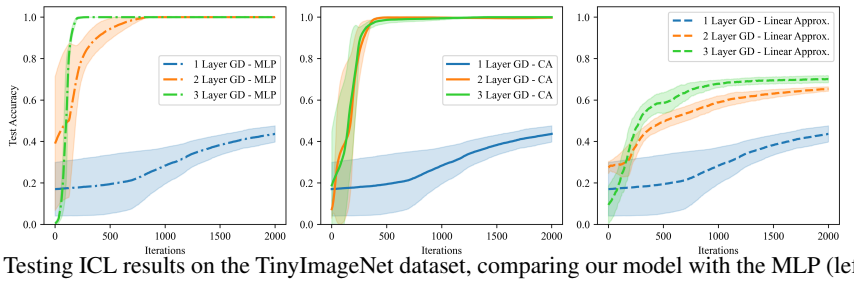

Figure 3: Testing ICL results on the TinyImageNet dataset, comparing our model with the MLP (left), and the CA model from Wang et al. (2025) (center), and our model with the linear approximation (right). For each of these Transformers sparse parameter training was performed guided by GD, and the models were trained on the Caltech256 dataset.

Following notation in the literature Cheng et al. (2024); von Oswald et al. (2023); Wang et al. (2025), when the Transformer parameters are trained from a random initialization of all parameters, without restrictions, the model is termed "Trained TF." A "GD" Transformer is one that is trained with constraints on its parameters, such that they are sparse in a way connected to functional GD analysis. We provide details on the GD-based implementation in the next section.

With categorical y_i, large quantities of training data are needed to learn accurate Trained TF models. The test results for the synthetic data are shown in Figures 1 and 2 as a function of the number of contextual datasets M employed for training, where $m = 1, \ldots, M$ and the mth training set is $\{(x_i^{(m)}, y_i^{(m)})\}$ for $i = 1, \ldots, N+1$ ($N = 50$ in these experiments). For the two- and three-layer models, Trained TF requires up to M =100,000 to match the results of GD trained on M =20,000. The need for large training sets for Trained TF was found in all our experiments, as was the relatively efficient training of GD.

Sparse Transformer design based on functional GD analysis performs well, and requires far less training data. In Figures 1 and 2, it is noted that GD trains far more effectively than Trained TF. A key reasons for this is that for the GD-based model the number of unknown parameters is substantially smaller than that of Trained TF (the parameter counts for all models, across all experiments, are summarized in Appendix A).

It is important to emphasize that the GD-based models are *not* exactly performing functional GD. However, as we discuss in the next sections, they perform inference that is closely related to (guided by) functional GD, and yields predictions that are very similar. In the next section we also explore details of the MLP-based and attention-only models, to explain why the latter is *sometimes* effective in this ICL setting, despite nonlinearities that are inherent to the setting of categorical y_i.

Results from the cross-attention (CA) model of Wang et al. (2025) agree well with our MLP-based Transformer. Considering the left and center subfigures in Figure 3, the highly similar predictive behavior of the Transformer with MLP layers, as compared to the CA model of Wang et al. (2025), indicates similar functional outputs of these two models (if not precise alignment in the underlying algorithm). As discussed further in Appendix B, this is believed to be connected to the MLP replacing the cross attention mechanistically. While we do not directly compare to Wang et al. (2025) in Figures 1 and 2, those data came from Wang et al. (2025), and the reader can verify that the results in Figures 1 and 2 align well with those published in Wang et al. (2025).

In *some* settings, attention-only Transformers are effective for ICL with categorical outcomes y_i. These initial findings present a compelling puzzle. Our synthetic data experiments (Figures 1-2) and our surgical data experiment that we will present later, in Section 6, demonstrate that attention-only models can perform remarkably well, challenging the theoretical need for MLPs. However, our experiments on Tiny ImageNet (Figure 3) show the opposite: MLPs provide a substantial performance gain. A key difference between these experiments is the nature of the data used for training the Transformer versus for testing it. In the experiments where attention-only models succeeded, the train and test data were drawn from similar distributions (synthetic-to-synthetic, surgery-to-surgery). In the experiment where MLPs were critical, the model was trained on natural images (Caltech256) and tested on a different domain (TinyImageNet and DomainNet). The additional experiments like Figure 3, shown in Appendix C, consider test data from DomainNet, which also represents a mismatch to the Caltech256 training data (see Figure 7). Those results also demonstrate poor performance of the attention-only Transformer.

Before concluding this section, we wish to benchmark the difficulty of the task in Figure 3. We considered k nearest neighbors of the query relative to the labeled contextual data, and it achieves about 40% accuracy on these data, as shown in Appendix C (see Table 5).

4 Mechanistic Investigation of Transformer Inference

4.1 Transformer inference through lens of functional GD

Like recent research on Transformer ICL (Cheng et al., 2024; von Oswald et al., 2023; Garg et al., 2022; Akyurek et al., 2023), we analyze how a Transformer *could* perform inference through the lens of functional gradient descent (GD). This entails examination of GD-based inference when y_i is categorical, and how aspects of such inference map to Transformer elements. While a Transformer could implement such inference, or its close analog, this does not mean that the Transformer *will* learn to do such inference in practice.

The in-context cross-entropy loss for the observed data (x_i, y_i) for $i = 1, \ldots, N$ is $\mathcal{L}(f) = -\frac{1}{N}\sum_{i=1}^{N} \log p(Y_i = y_i | f(x_i))$ using the softmax-based likelihood function in (4). If we assume that $f \in \mathcal{F}$ where $\mathcal{F}$ is a reproducing kernel Hilbert space (RKHS) (Schölkopf & Smola, 2002) with kernel $\kappa(x_i, x_j)$ (like done in Cheng et al. (2024) for real y_i), then functional gradient descent (GD) for $f(x_i)$ yields the updates:

$$f_{i,\ell+1} = f_{i,\ell} + \Delta f_{i,\ell}\ , \qquad \Delta f_{i,\ell} = \frac{\alpha}{N}\sum_{j=1}^{N}[w_{y_j} - \mathbb{E}(w|f_{j,\ell})]\kappa(x_i, x_j) \tag{5}$$

where $f_{i,\ell}$ represents $f(x_i)$ after $\ell > 0$ steps of functional GD, and α is the learning rate (in general, different for each functional GD step ℓ, and a different rate for each component of the latent function). The update in (5) is derived in Appendix E. The expectation

$$\mathbb{E}(w|f_{i,\ell}) = W_e \cdot \text{Softmax}(W_e^\top f_{i,\ell}) \tag{6}$$

where $W_e \in \mathbb{R}^{d' \times C}$ has columns defined by the C category embedding vectors. The expression $\mathbb{E}(w|f_{i,\ell})$ is the expectation over category embedding vectors, given $f_{i,\ell}$ and the softmax over categories in (4). The index ℓ is used for functional GD *steps*, because it will be connected to *layers* of the Transformer.

Recall from Section 2 the imposition of scratch space in the encoding of the vectors flowing through the Transformer; we now provide more details on how it is utilized in our model, for the Transformer forward pass to infer $f_{i,\ell}$. For position i at Transformer layer ℓ, we impose:

$$e_{i,\ell}^\top = \Big[\ \ x_i\ \ ,\ \ w_{y_i}\ \ ,\ \underbrace{f_{i,\ell}\ \ ,\ \ \mathbb{E}(w|f_{i,\ell})}_{\text{scratch space}}\ \Big] \tag{7}$$

As discussed below, we will connect GD steps to Transformer layers, motivating use of ℓ for GD steps. The position of $f_{i,\ell}$ in (7) is imposed by using those d' positions of $e_{N+1,L+1}$ (output from the Transformer) as the input to the softmax over categories (recall (4)) for prediction of the probability over categories at the output. In (7), it is meant that the d'-dimensional position for $\mathbb{E}(w|f_{i,\ell})$ is where such is placed in the scratch space, although in general this expectation is *not* computed exactly, and a trained Transformer need not be constrained to exactly perform GD-based inference.

With proper initialization, a Transformer can perform the first step of functional GD *exactly*, without MLP layers. As in previous Transformer-based ICL research (Cheng et al., 2024; von Oswald et al., 2023; Wang et al., 2025), we initialize $f_{i,0} = 0_{d'}$ for all i, and consequently in (7) this implies the initial expectation $\mathbb{E}(w|f_{i,0}) = \frac{1}{C}\sum_{c=1}^{C} w_c$, which we note is the same for all i. As shown in Appendices F and I, there are Transformer self-attention parameters such that the first update $\Delta f_{i,0}$ may be implemented *exactly*, for all $i = 1, \ldots, N+1$. Importantly, this first step of inference of the latent $f_{i,1}$ can be done with the first self-attention layer alone (no need for subsequent MLP).

However, if one is to perform further GD steps, the associated nonlinear expectation must correspondingly be updated. Specifically, for steps $\ell > 0$ one must update $\mathbb{E}(w|f_{i,\ell+1})$ based on $f_{i,\ell+1}$ from the preceding self-attention layer. As $\mathbb{E}(w|f_{i,\ell+1})$ is a nonlinear function of $f_{i,\ell+1}$, it is anticipated that nonlinear functions layers should follow each self-attention layer, to compute $\mathbb{E}(w|f_{i,\ell+1})$. This suggests a role for the Transformer MLP layers and the following overall Transformer interleaved process:

- Self-attention layer: Update $f_{i,\ell} \to f_{i,\ell+1}$, using existing approximation to $\mathbb{E}(w|f_{i,\ell})$.
- Subsequent MLP layer: Using the updated $f_{i,\ell+1}$, compute and store $\mathbb{E}(w|f_{i,\ell+1})$.

As detailed in Appendix F, an $H = 2$ head Transformer, with interleaved self-attention and MLP layers, *can* approximate aforementioned two interleaved steps of in-context learning.

4.2 Linear Approximation to expectation: Attention-only Transformer

In Appendix H we show that under a first-order Taylor expansion

$$\mathbb{E}(w|f_{i,\ell}) \approx \mathbb{E}(w|f_{i,\ell-1}) + W_e \cdot \text{softmax}(W_e^\top f_{i,\ell-1}) \cdot \tilde{W}_{e,\ell-1} \cdot \Delta f_{i,\ell-1} \tag{8}$$

recalling that the cth column of $W_e \in \mathbb{R}^{d' \times C}$ is the embedding vector w_c; the cth column of $\tilde{W}_e$ is $w_c - \mathbb{E}(w|f_{i,\ell-1})$.

With the *same* initialization as discussed above, $f_{i,0} = 0_{d'}$, $\text{softmax}(W_e^\top f_{i,0})$ is uniformly distributed for all i, and therefore we may approximate

$$\mathbb{E}(w|f_{i,1}) \approx \frac{1}{C}\Big[\sum_{c=1}^{C} w_c + W_e \tilde{W}_e^\top \Delta f_{i,0}\Big] \tag{9}$$

where $\Delta f_{i,0}$ is the output of the preceding (first) self-attention layer, and $\tilde{W}_e$ is *independent of* index i. As shown in Appendix I, the linear update in (9) can be implemented within the same self-attention layer used to compute $\Delta f_{i,0}$, and therefore there is not a need for an MLP layer for its computation (*within the linear approximation*).

There are two reasons that the approximation in (9) could work well: (1) $f_{i,0} = 0_{d'}$ is the same for all $i = 1, \ldots, N$, and therefore the *same* linear approximation holds for all i; (2) the linear approximation is performed about $f_{i,0} = 0_{d'}$, which is the *center* of the linear region of the softmax, for *all* $c = 1, \ldots, C$. Hence, this linear approximation is particularly well suited for updating $\mathbb{E}(w|f_{i,1})$, under the imposed $f_{i,0}$.

The first *two* steps of functional GD typically can be performed well with self-attention alone, *assuming a good match between the training and testing data*. As discussed in the previous subsection, with the initialization $f_{i,0} = 0_{d'}$, the first GD step can be done *exactly* with the first self-attention layer (note that setting $f_{i,0} = 0_{d'}$ is not special to this setting; it aligns with all prior ICL research (von Oswald et al., 2023; Cheng et al., 2024; Wang et al., 2025; Ahn et al., 2023; 2024)). Based on the above discussion, with that same first attention layer, one may accurately update $\mathbb{E}(w|f_{i,0}) \to \mathbb{E}(w|f_{i,1})$ within a linear approximation. The next (second) self-attention layer, leveraging the updated $\mathbb{E}(w|f_{i,1})$, can then perform the second step to compute $\Delta f_{i,1}$. Hence, it is anticipated that the first *two* steps of functional GD inference can be performed well with two self-attention layers, and *no* MLPs. Importantly, it is possible that this approximation could be sensitive to a match between the training and testing data, which the experiment in Figure 3 violates.

After the first two self-attention layers, the linear approximation is less appropriate. From (9), the linear approximation for $\mathbb{E}(w|f_{i,2})$ involves an expansion about $f_{i,1}$. There are two problems with this: (1) In general $f_{i,1}$ is *different* for each i, and therefore the same linear approximation *does not* hold for all i (but in an attention-only Transformer, we *have* to assume the same linear relationship for all i); (2) depending on $f_{i,1}$, the softmax function may no longer be in its linear regime, further undermining the linear approximation. Therefore, for $\mathbb{E}(w|f_{i,\ell})$ for $\ell \geq 2$, which are needed for GD steps 3 and beyond, a *nonlinear* representation of the expectation may be important. This implies that for layers three and beyond of the Transformer, MLP layers ideally should follow each self-attention layer.

4.3 Finer-grained examination of MLP & Linear Approximation

In the above discussion we have suggested that a role of the MLP, for categorical y_i, may be in computing the nonlinear expectation $\mathbb{E}(w|f_{i,\ell})$. We have also articulated domains in which a linear approximation may be adequate. To test this further, we trained a cross-attention Transformer from Wang et al. (2025), in which the expectation is performed exactly. We kept all parameters unchanged, but dropped in either an MLP or linear approximation to represent $\mathbb{E}(w|f_{i,\ell})$, instead of using the cross attention. As detailed in Appendix B and shown in Figure 6, both the MLP and the linear approximation were able to replicate the exact expectation with very low error. In Appendix B we also perform comparisons of our Transformer predictions, with and without MLP layers, to the only prior ICL model Wang et al. (2025) that considered categorical y_i (the latter is not a conventional Transformer, as it has cross-attention not MLP layers).

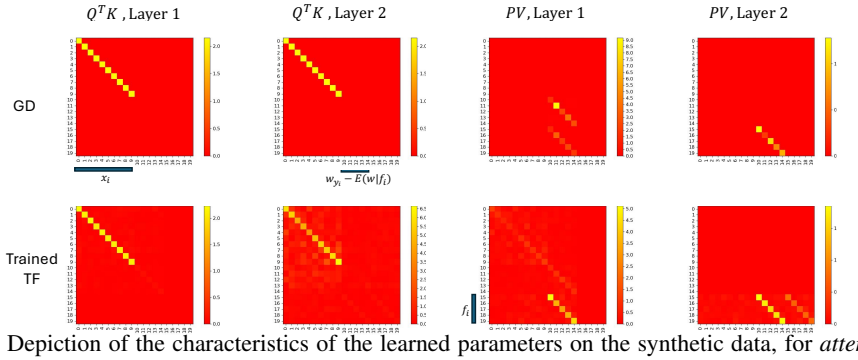

Figure 4: Depiction of the characteristics of the learned parameters on the synthetic data, for *attention-only* two-layer Transformers, where the top row is from GD and the bottom row is Trained TF. The Transformers employ softmax attention, so the key-query attention is defined by $Q^\top K$ (left two columns). The right two columns depict how the vectors are updated via attention, manifested by PV. We identify which part of the parameters are tied to x_i, $w_{y_i} - \mathbb{E}(w|f_{i,\ell})$ and $f_{i,\ell}$.

5 What Algorithm is "Trained TF" Learning, with Enough Data?

In Figures 1-2 we noted close agreement between Trained TF and GD predictive accuracy and cross-entropy loss (on test data), assuming that the training set for Trained TF was large enough. While these predictive results indicate that these models offer similar predictions, it does not mean that the two models are acting similarly mechanistically. We now examine the characteristics of the parameters learned for the Trained TF and GD models, to examine their similarity (or not), and hence to examine the degree to which these models implement similar algorithms at inference.

With softmax-based attention, using query and key matrices Q and K respectively, for vectors e_i and e_j at the respective Transformer positions, attention involves elements of the form $\exp[\lambda(Qe_j)^\top(Ke_i)] = \exp[\lambda(e_j^\top Q^\top K e_i)]$, and therefore attention is dependent on the matrix product $Q^\top K$. Similarly, from the review of the Transformer in Section 2, (for one attention head) the output of attention involves the matrix product PV. The Transformer does not depend on the specific form of Q, K, V and P (which are in general not identifiable), but it does depend on the aforementioned matrix products, which guide the algorithm the Transformer implements.

In Figure 4 we present these learned matrix products, for the GD-based and Trained TF models. Results are shown for a two-layer model. While the Trained TF and GD matrix products do not agree exactly, there is close agreement in general, indicating that these two Transformers are performing similar algorithms. For example, from the left two columns in Figure 4 it is seen that Trained TF (like GD) computes attention weights based on the covariates x_i. We see that the latent function $f_{i,\ell}$ is updated at the output of layers 1 and 2, for both GD and Trained TF, while the expectation is only updated in both cases at the output of layer 1.

These results are for the self-attention-alone form of the Transformer, chosen because it can be implemented with a single attention head (see Appendix I), and therefore the comparisons are less ambiguous. We have also performed comparisons when the MLP layers are present, and similar agreement in implied underlying algorithms is revealed. The presence of MLP layers is more complicated to compare, because it involves two attention heads (see Appendix F). We emphasize that the general level of agreement reflected in Figure 4 is only manifested when the Trained TF is trained with 100,000 or more training examples, which is consistent with Figures 1 and 2 that it is only with such large training sets that Trained TF performance approaches that of the GD-based Transformer.

In Table 4 of Appendix B we perform additional close inspection of the relationship between the Trained TF and GD-based models. Those experiments further the understanding that Trained TF (when trained on enough data) makes similar predictions to the GD-based counterpart.

6 Real-World Use of Transformer-Based ICL

We consider in-context answering of questions about images, where the questions have a set of categorical answers. The context-dependent probability of answers to question $q = 1, \ldots, Q$ is

$$p(Y_{N+1}^{(q)} = c | X_{N+1} = x_{N+1}, \mathcal{C}) = \frac{\exp[(w_c^{(q)})^\top \hat{f}(x_{N+1})]}{\sum_{c'=1}^{C} \exp[(w_{c'}^{(q)})^\top \hat{f}(x_{N+1})]} \tag{10}$$

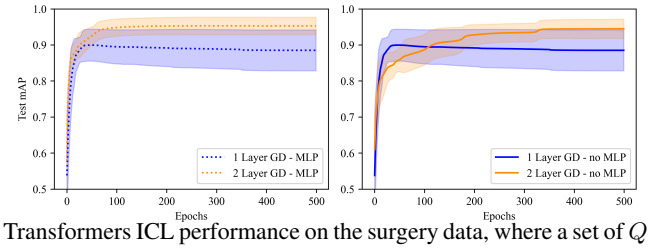

Figure 5: GD-based Transformers ICL performance on the surgery data, where a set of $Q = 5$ *binary* questions are answered, and each question connects to the presence of a surgical instrument and/or organ in the image. Left: MLP layers present, Right: no MLP layers (attention-only). Results are shown on test data, for which the items in the questions were *not* seen in the training images.

where $w_c^{(q)} \in \mathbb{R}^{d'}$ is an embedding vector for answer $c \in \{1, \ldots, C\}$, question $q \in \{1, \ldots, Q\}$.

For this setting, we show in Appendix G that from functional GD analysis:

$$f_{i,\ell+1} = f_{i,\ell} + \frac{\alpha}{N} \sum_{j=1}^{N} \frac{1}{Q} \sum_{q=1}^{Q} [w^{(q)}_{y_j^{(q)}} - \mathbb{E}(w^{(q)}|f_{j,\ell})] \kappa(x_i, x_j) \tag{11}$$

where the data for sample i is $(x_i, y_i^{(1)}, \ldots, y_i^{(Q)})$, where $y_i^{(q)} \in \{1, \ldots, C\}$ is the answer observed for question $q \in \{1, \ldots, Q\}$; for a context of size N, data of this form exists for $i = 1, \ldots, N$, and there is a query x_{N+1}. The GD-based sparse Transform design is detailed in Appendix G.

We leverage the CholecT45 dataset, a subset of the CholecT50 dataset Nwoye et al. (2023); Nwoye & Padoy (2022), and train the Transformer to detect action "triplets." CholecT45 consists of 45 video recordings of laparoscopic cholecystectomy, formalizing surgical activities in the form triplets <*instrument, verb, target*>. There are a total of 100 action triplet classes, composed from 6 instruments *(grasper, bipolar, hook, scissors, clipper, irrigator)*, 10 verbs *(grasp, retract, dissect, coagulate, clip, cut, aspirate, irrigate, pack, null)*, and 15 targets *(gallbladder, cystic-duct, cystic-artery, blood-vessel, fluid, abdominalwall or cavity, liver, omentum, peritoneum, gut, specimen-bag, null)*. In Appendix D we show example images from this dataset.

We consider the 25 most prevalent triplets in our empirical evaluations. We divide the dataset into two halves: a train/validation set and a test set. For train/validation, we consider surgical images with a randomly chosen 10 action triplets, and for testing, we consider the images with the rest of the 15 action triplets. We use three different random seeds and report the model performance in the form of mAP score (Nwoye & Padoy, 2022; Nwoye et al., 2022) by taking average over three seeds.

There is an important distinction between this experiment and that considered in Figure 3. In the latter, not only were the label classes seen at test different, the form of the images was different. By contrast, while here the triplets seen while training are different from those considered at test, for both training and testing the images are from surgeries of the same general type, so the form of the images is *not* mismatched. Example images from this dataset are shown in Appendix D.

In this experiment the feature extractor for x_i are the same masked autoencoder features as considered in Figure 3, of dimension $d = 768$. We considered embedding vectors of dimension $d' = 4$. Each contextual block considered images from $Q = 5$ action-triplet types, and the query was also from those 5 action-triplets (this is distinct from classification, because multiple triplets – up to 5 – can be in the same image). A total of $N = 50$ contextual labeled samples are provided to the Transformer, plus a query, x_{N+1}. Empirical results are shown on Figure 5, using GD with and without MLP layers. These two models perform similarly, achieving mAP > 0.9 with two layers. The MLP-based Transformer achieves a slightly higher mAP score.

7 Conclusions

We have considered Transformer-based ICL for data with categorical outcomes. We have highlighted nonlinearities that arise in this setting, that were absent in almost all prior work with real outcomes. We have performed a detailed mechanistic analysis of Transformer inference in this setting, showing that the Transformer learns to perform inference in a manner that has close (but not exact) connections to functional GD. In some experiments attention-only Transformers performed well, and we performed a detailed analysis of why this occurs. However, Transformers with MLP layers *always* performed well, and they are recommended. We demonstrated the utility of this technology on a real-world problem connected to human surgery.

REPRODUCIBILITY STATEMENT

All software from this paper will be released without constraints, upon publication. All datasets considered in the paper are in the public domain, and therefore all of our experiments are reproducible.

REFERENCES

K. Ahn, X. Cheng, H. Daneshmand, and S. Sra. Transformers learn to implement preconditioned gradient descent for in-context learning. *arXiv.2306.00297*, 2023.

K. Ahn, X. Cheng, M. Song, C. Yun, A. Jadbabaie, and S. Sra. Linear attention is (maybe) all you need (to understand transformer optimization), 2024.

E. Akyurek, D. Schuurmans, J. Andreas, T. Ma, and D. Zhou. What learning algorithm is in-context learning? investigations with linear models. *International Conference on Learning Representations*, 2023.

C. Brix, P. Bahar, and H. Ney. Successfully applying the stabilized lottery ticket hypothesis to the transformer architecture. In *Proc. Association for Computational Linguistics*, Online, July 2020. Association for Computational Linguistics.

T.B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D.M. Ziegler, J. Wu, Cl. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

X. Cheng, Y. Chen, and S. Sra. Transformers implement functional gradient descent to learn non-linear functions in context. *arXiv.2312.06528*, 2024.

DeepSeek. Deepseek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv:2501.12948v1*, 2025.

J. Devlin, M.-Wei C., K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv.1810.04805*, 2019.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv:2010.11929*, 2020.

F. Draxler, K. Veschgini, M. Salmhofer, and F. A. Hamprecht. Essentially no barriers in neural network energy landscape. In *International Conference on Machine Learning*, 2018.

J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

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

T. Garipov, P. Izmailov, D. Podoprikhin, D. P. Vetrov, and A. G. Wilson. Loss surfaces, mode connectivity, and fast ensembling of DNNs. In *Advances in Neural Information Processing Systems*, 2018.

G. Griffin, A. Holub, P. Perona, et al. Caltech-256 object category dataset. Technical report, Technical Report 7694, California Institute of Technology Pasadena, 2007.

K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

Y. Le and X. Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

N. Lee, T. Ajanthan, and P. H. S. Torr. SNIP: Singleshot network pruning based on connection sensitivity. In *International Conference on Learning Representations*, 2019.

C. Innocent Nwoye and N. Padoy. Data splits and metrics for method benchmarking on surgical action triplet datasets. *arXiv:2204.05235*, 2022.

C. Innocent Nwoye, D. Alapatt, T. Yu, A. Vardazaryan, F. Xia, Z. Zhao, T. Xia, F. Jia, Y. Yang, and H. Wang. Cholectriplet2021: A benchmark challenge for surgical action triplet recognition. *Medical Image Analysis*, 86:102803, 2023.

C.I. Nwoye, T. Yu, C. Gonzalez, B. Seeliger, P. Mascagni, D. Mutter, J. Marescaux, and N. Padoy. Rendezvous: Attention mechanisms for the recognition of surgical action triplets in endoscopic videos. *Medical Image Analysis*, 78:102433, 2022.

X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1406–1415, 2019.

A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019.

B. Schölkopf and A.J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. LLaMA: Open and efficient foundation language models. *arXiv.2302.13971*, 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *arXiv.1706.03762*, 2017.

J. von Oswald, E. Niklasson, E. Randazzo, J. Sacramento, A. Mordvintsev, A. Zhmoginov, and M. Vladymyrov. Transformers learn in-context by gradient descent. *arXiv.2212.07677*, 2023.

A. Wang, W. Convertino, X. Cheng, R. Henao, and L. Carin. On understanding attention-based in-context learning for categorical data. In *International Conference on Machine Learning*, 2025.

R. Zhang, S. Frei, and P.L. Bartlett. Trained transformers learn linear models in-context. *arXiv:2306.09927*, 2023.

Table of Contents for Appendix

A PARAMETER COUNTS FOR EXPERIMENTS

Table 1: Comparison of total number of parameters between GD and Trained TF, with or without MLP, on the synthetic dataset.

Layers	GD-Linear Approximation	TF-Linear Approxmation	GD-MLP	TF-MLP
1	156	1725	296	5290
2	186	3325	301	10290
3	216	4925	306	15290

Table 2: Comparison of total number of parameters between GD - MLP and GD - Linear Approximation, on the image dataset considered in Figure 3.

Layers	GD-Linear Approximation	GD-MLP
1	56	146
2	86	151
3	116	156

Table 3: Comparison of total number of parameters between GD - MLP and GD - Linear Approximation, on the surgery dataset.

Layers	GD-Linear Approximation	GD-MLP
1	105	1245
2	109	1249
3	113	1253

B FURTHER EXAMINATION OF THE ROLE OF MLP AND LINEAR APPROXIMATION

B.1 REPRESENTATION OF $\mathbb{E}(w|f_{i,\ell})$ BY MLP/LINEAR APPROXIMATION

We have postulated a role for the MLP units within a Transformer, when y_i is categorical, could be for computing the expectation $\mathbb{E}(w|f_{i,\ell})$. As a test for that, we consider the cross-attention-based Transformer of Wang et al. (2025), which effectively performs exact functional GD, with $\mathbb{E}(w|f_{i,\ell})$ computed exactly via cross attention. Such a Transformer was trained on the synthetic data of Section 3. After training, all model parameters were frozen, and the cross-attention was removed. In

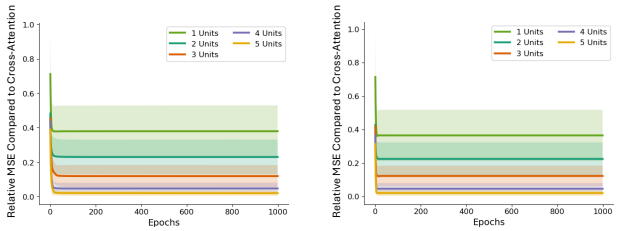

Figure 6: The cross-attention-based Transformer (Wang et al., 2025) was trained on the synthetic data of Section 3 and all parameters frozen. The cross attention was removed, and replaced by an MLP (left) or with a linear approximation (right). The parameters of the MLP/matrix were then trained, with all other parameters unchanged. Here is plotted the MSE error relative to the norm of the output of the cross-attention output.

Table 4: Comparison of the softmax probability over categories, at the output of Transformer ICL. At right, as a reference, we show the entropy of the softmax predictions, based on the cross-attention (CA) Transformer (Wang et al., 2025), that effectively performs exact functional GD. We also consider the *cross-entropy* between the output of Transformers considered here, relative to the softmax outputs of CA (Wang et al., 2025). The cross attention is shown between a GD-based form of the Transformer and with Trained TF (TF). Models of 1, 2 and 3 layers are considered, with the self-attention-only version of the ICL Transformer, and with MLP layers.

Model	**GD vs CA**	**TF vs CA**	**Entropy of CA**
Self-Attention Only (1 Layer)	1.377 ± 0.062	1.442 ± 0.089	1.238 ± 0.035
Self-Attention Only (2 Layers)	1.310 ± 0.043	1.282 ± 0.038	1.084 ± 0.031
Self-Attention Only (3 Layers)	1.244 ± 0.056	1.234 ± 0.052	1.028 ± 0.038
MLP layers present (1 Layer)	1.357 ± 0.033	1.442 ± 0.091	1.238 ± 0.035
MLP layers present (2 Layers)	1.268 ± 0.049	1.332 ± 0.038	1.084 ± 0.031
MLP layers present (3 Layers)	1.204 ± 0.051	1.271 ± 0.051	1.028 ± 0.038

place of the cross attention, we dropped in an MLP, and only trained the MLP parameters. Similarly, we dropped in a matrix approximation, and only trained the matrix. We wish to consider the degree to which the MLP/linear approximation recover the performance of the original cross-attention-based Transformer.

In Figure 6 we show results as a function of the number of hidden units in the MLP, which for the matrix approximation corresponds to the matrix rank. We observe that with five units (full rank matrix approximation), the MLP and linear approximation emulate the expectation accurately, in the relative MSE error compared to the output of the cross-attention diminishes quickly with increasing number of units in the MLP/linear models.

These results are for a two-layer model. As discussed in Section 4, this is the regime for which we expect the linear approximation to work well.

B.2 Comparison of predictions relative to Wang et al. (2025)

In Figures 1 and 2 we considered predictions and the cross-entropy loss on the synthetic data introduced in Wang et al. (2025). We here examine the similarity of predictions for the cross-attention-based ICL model of Wang et al. (2025) on these data relative to our model, with and without the MLP layers. Rather than comparing just the output predictions, which reflects top-1 predictions at the softmax output of the model, we here compare the full softmax-generated probability mass function (PMF). Specifically, we treat the predictions of the cross-attention (CA) model of Wang et al. (2025) as "ground truth," because they effectively correspond to exact functional GD. We calculate the cross-entropy between the PMF generated by the CA model of Wang et al. (2025) to the output PMF from our model, with and without MLP layers. Results are summarized in Table 4.

By considering the cross-entropy between the CA-based PMF and the PMFs of our model, we examine how closely our generated PMFs align with predictions from exact functional GD. Our GD models are *guided* by functional GD analysis, but the MLP layers are sufficiently flexible to possibly do better than functional GD inference.

Multiple issues can be examined by evaluating Table 4:

- Comparing results for GD vs CA and TF vs CA, we can evaluate the similarity of the GD and Trained TF models, respectively (comparing the GD vs CA column to the TF vs. CA column).
- For either the GD or Trained TF model, we can compare variation with and without the MLP layers (comparing rows 1-3 to rows 4-6).
- The entropy of the CA model serves as a lower bound for the cross entropy.

The results in Table 4 indicate that the Trained TF softmax output is (on average) further from exact functional GD (the CA model Wang et al. (2025)) than is the GD-based design of our Transformer. However, the differences are within the standard deviation of the experiments. These results suggest that while the Trained TF model (trained without constraints) yields inference predictions, here across all 25 elements of the softmax output, that are relatively close to exact functional GD. We also note that for these simulated data, the full softmax output from the Transformer with and without MLPs are similar, with differences within the standard deviation. However, we emphasize that one cannot expect the self-attention-alone models to be sufficient in general, as shown by Figures 3 and 7. The effectiveness of the linear approximation in (9) is key to whether attention-alone is sufficient. However, in Section 6 we show another experiment, with real-world data, for which the attention-only Transformer performs well.

C Comparison of GD with Self-attention alone, Cross Attention, and MLP layers, in the presence of data mismatch

In Figure 3 we presented results for Transformers trained on Caltech256, and tested on TinyImageNet. Those results, based on a GD-based sparse parameter implementation, for 1 to 3 layer models, showed good agreement between the Transformer with MLP layers and the cross-attention-based model of Wang et al. (2025). Importantly, they showed that the linear approximation did not work well in this setting, that corresponding to the attention-only form of the Transformer.

To further examine this case, for which there is mismatch between the training and testing data, we now present results when the model is again trained on Caltech256, but now tested on DomainNet. In particular, we consider the ClipArt, InfoGraph, QuickDraw, Real and Sketch image forms from the DomainNet dataset, with results summarized in Figure 7. These results demonstrate the same conclusions as Figure 3: (1) good alignment of predictions from the CA-based model of Wang et al. (2025) and our GD-based model in which the MLP layers are present. (2) Poor performance of the 2 and 3 layer versions of the Transformer that made a linear approximation to $\mathbb{E}(w|f_{i,\ell})$, and hence inappropriateness of the attention-alone Transformer.

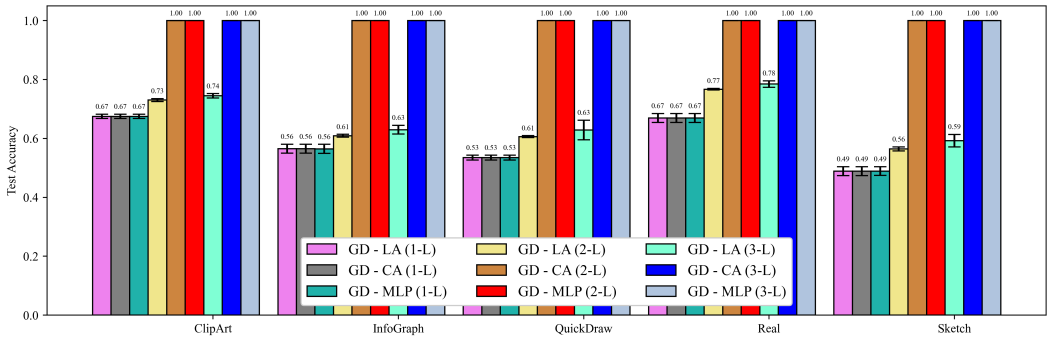

Figure 7: Performance of the GD-based Transformers in which the MLP layers are present, a linear approximation (LA) is employed for $\mathbb{E}(w|f_{i,\ell})$, and the cross-attention (CA) model of Wang et al. (2025). The Transformers were trained on Caltech256, and tested on DomainNet data. Error bars reflect standard deviation from multiple initializations of the Transformer parameters.

As a separate test of the difficulty of this ICL problem, we implemented a simple k-nearest neighbor (kNN) classifier on the contextual data. Specifically, given the features from the masked autoencoder

features, we simply calculated which k of the labeled contextual feature vectors were closest to the feature vector of the query x_{N+1}, and did a majority vote. Results are shown in Table 5, for different choices of k, revealing consistently about 40% accuracy, which is similar to the performance of the one-layer Transformers, as shown in Figure 3, but notably inferior to the deeper models, even that with linear approximation for $\mathbb{E}(w|f_{i,\ell})$.

Table 5: kNN Test Accuracy on the TinyImageNet dataset, using Masked Autoencoder (He et al., 2022) extractor (the same covariates used by the Transformers).

k (Neighbors)	Test Accuracy
1	0.4092
2	0.4092
3	0.4190
4	0.4232
5	0.4232
6	0.4250
7	0.4258
8	0.4142
9	0.4160
10	0.4130

D EXAMPLE IMAGES FROM THE SURGERY DATASET

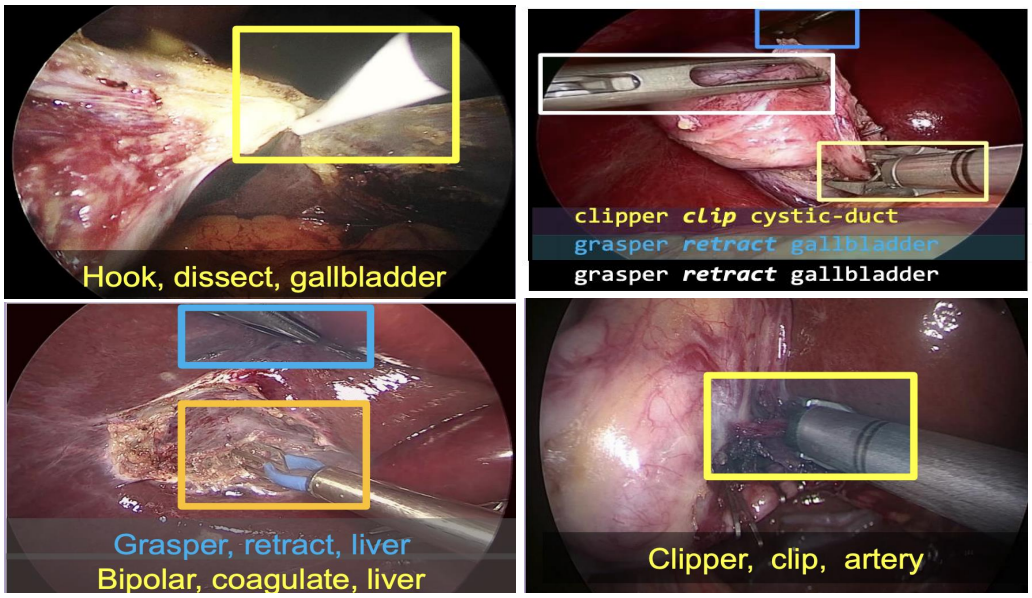

Figure 8: Example figures taken from CholecT45 dataset (Nwoye et al., 2023; Nwoye & Padoy, 2022), demonstrating surgical images with their associated action triplets. Bounding boxes are added manually for better understanding.

Example images from the surgery dataset are shown in Figure 8. The rectangles identify where in the image a "triplet" resides, and the three words associated with the triplet are also depicted. Note that the number if triplets in an image is greater than or equal to one, and the ICL algorithm is tasked with identifying the presence/absence of all triplets in a given image. Note that the triplets are diverse, and that our ICL Transformer is trained on one set of possible triplets, and tested on a distict set. While the *type* of triplets between training and testing data are distict, all images are connected to surgery, so there is not a mismatch on the form of the images.

E Derivation of the Functional GD Update Equation for Categorical y_i

Our derivation is based on the assumption that $f(x)$ resides in a reproducing kernel Hilbert space (RKHS) (Schölkopf & Smola, 2002), but the setup extends to softmax-based attention kernels as well (Wang et al., 2025). From the RKHS perspective, let $f(x) = A\psi(x) + b$, with $\psi(x)$ a *fixed* mapping of covariates x to a Hilbert space, and the parameters acting in that space are (A, b).

The cross-entropy cost function for inferring the parameters $A \in \mathbb{R}^{d' \times m}$ and $b \in \mathbb{R}^{d'}$, may be expressed as

$$\begin{aligned}\mathcal{L}(A,b) &= -\frac{1}{N}\sum_{i=1}^{N}\log\left[\frac{\exp[w_{y_i}^T(A\psi(x_i)+b)]}{\sum_{c=1}^{C}\exp[w_c^T(A\psi(x_i)+b)]}\right] \\ &= -\frac{1}{N}\sum_{i=1}^{N}[w_{y_i}^T A\psi(x_i) + w_{y_i}^T b - \log\sum_{c=1}^{C}\exp(w_c^T A\psi(x_i) + w_c^T b)]\,. \end{aligned} \tag{12}$$

Taking the partial derivative of $\mathcal{L}$ wrt b_j, component j of b:

$$\begin{aligned}\frac{\partial}{\partial b_j}\mathcal{L} &= -\frac{1}{N}\sum_{i=1}^{N}[w_{y_i}(j) - \frac{\sum_{c=1}^{C}\exp[w_c^T(A\psi(x_i)+b)]w_c(j)}{\sum_{c\prime=0}^{C}\exp[w_{c'}^T(A\psi(x_i)+b)]}] \\ &= -\frac{1}{N}\sum_{i=1}^{N}[w_{y_i}(j) - \frac{\sum_{c=1}^{C}\exp[w_c^T f_i]w_c(j)}{\sum_{c\prime=0}^{C}\exp[w_{c'}^T f_i]}]\,,\end{aligned}$$

where $w_{y_i}(j)$ is component j of $w_{y_i} \in \mathbb{R}^{d'}$, and $f_i = A\psi(x_i) + b$. Therefore

$$\begin{aligned}\nabla_b\mathcal{L} &= -\frac{1}{N}\sum_{i=1}^{N}\left[w_{y_i} - \frac{\sum_{c=1}^{C}\exp[w_c^T f_i]w_c}{\sum_{c\prime=0}^{C}\exp[w_{c'}^T f_i]}\right] \\ &= -\frac{1}{N}\sum_{i=1}^{N}\left[w_{y_i} - \mathbb{E}(w|f_i)\right]\,.\end{aligned} \tag{13}$$

We consequently have the GD update rule for b

$$b^{(l+1)} = b^{(l)} + \frac{\alpha}{N}\sum_{i=1}^{N}\left[w_{y_i} - \mathbb{E}(w|f_{i,\ell})\right], \tag{14}$$

where $l \geq 0$ is the GD step index, initialized at $l = 0$.

Similarly, let a_j represent the jth row of A. Taking the gradient of $\mathcal{L}$ wrt a_j:

$$\begin{aligned}\nabla_{a_j}\mathcal{L} &= -\frac{1}{N}\sum_{i=1}^{N}\Big[w_{y_i}(j)\psi(x_i) - \frac{\sum_{c=1}^{C}\exp[w_c^T(A\psi(x_i)+b)]w_c(j)\psi(x_i)}{\sum_{c\prime=0}^{C}\exp(w_{c'}^T(A\psi(x_i)+b))}\Big] \\ &= -\frac{1}{N}\sum_{i=1}^{N}\Big[w_{y_i}(j) - \frac{\sum_{c=1}^{C}\exp[w_c^T f_i]w_c(j)}{\sum_{c\prime=0}^{C}\exp(w_{c'}^T f_i)}\Big]\psi(x_i) \\ &= -\frac{1}{N}\sum_{i=1}^{N}\Big[w_{y_i}(j) - \mathbb{E}(w(j)|f_i)\Big]\psi(x_i)\,.\end{aligned} \tag{15}$$

The gradient update step for a_j is

$$\begin{aligned}a_j^{(l+1)} &= a_j^{(l)} - \alpha\nabla_{a_j}\mathcal{L} \\ &= a_j^{(l)} + \frac{\alpha}{N}\sum_{i=1}^{N}\Big[w_{y_i}(j) - \mathbb{E}(w(j)|f_i)\Big]\psi(x_i)\,.\end{aligned}$$

Using the GD update rules for b and $\{a_j\}_{j=1,d'}$, we have

$$f_j^{(l+1)} = \begin{pmatrix} (a_1^{(l+1)})^T \psi(x_j) + b_1^{(l+1)} \\ \vdots \\ (a_{d'}^{(l+1)})^T \psi(x_j) + b_{d'}^{(l+1)} \end{pmatrix}$$

$$= f_j^{(l)} + \frac{\alpha}{N} \sum_{i=1}^{N} \left[w_{y_i} - \mathbb{E}(w|f_{i,\ell}) \right] \kappa(x_i, x_j) + \alpha \sum_{i=1}^{N} \left[w_{y_i} - \mathbb{E}(w|f_{i,\ell}) \right] . \tag{16}$$

In the main body of the paper we omitted the term $\alpha \sum_{i=1}^{N} \left[w_{y_i} - \mathbb{E}(w|f_{i,\ell}) \right]$, which is connected to the bias update.

F Transformer Parameters for Multi-Step GD Via Self-Attention & MLP Layers

The input to the Transformer at layer l is

$$e_{i,\ell} = \begin{pmatrix} f_{i,\ell} \\ \mathbb{E}(w|f_{i,\ell}) \\ w_{y_i} \\ x_i \end{pmatrix} \tag{17}$$

Within $e_{i,\ell}$, the vector component $f_{i,\ell}$ is iteratively updated with increasing layer index l, with the update manifested by each self-attention layer. The expectation $\mathbb{E}(w|f_{i,\ell})$ is updated by each MLP layer. Vector components $f_{i,\ell}$ and $\mathbb{E}(w|f_{i,\ell})$ occupy what we term as *computational scratch space*. The covariates x_i and embedding vector w_{y_i} represent the encoding of the data (x_i, y_i), and the portion of $e_{i,\ell}$ occupied by (x_i, w_{y_i}) remains fixed at all Transformer layers.

Each attention block consists of a self-attention layer, composed of two attention heads; one of these attention heads implements $f_{i,\ell} \to f_{i,\ell+1}$ like above (for which $\mathbb{E}(w|f_{i,\ell})$ is needed), and the second attention head erases $\mathbb{E}(w|f_{i,\ell})$, preparing for its update by the subsequent MLP layer.

F.1 Self-attention layer

In matrix form, the input at layer l is

$$\begin{pmatrix} f_{1,\ell} & \dots & f_{N,\ell} & f_{N+1,\ell} \\ \mathbb{E}(w|f_{1,\ell}) & \dots & \mathbb{E}(w|f_{N,\ell}) & \mathbb{E}(w|f_{N+1,\ell}) \\ w_{y_1} & \dots & w_{y_N} & 0_{d'} \\ x_1 & \dots & x_N & x_{N+1} \end{pmatrix} \tag{18}$$

The update equation for $f_{i,\ell+1}$ is given by

$$f_{i,\ell+1} = f_{i,\ell} + \Delta f_{i,\ell} \tag{19}$$

where

$$\Delta f_{i,\ell} = \frac{\alpha}{N} \sum_{i=1}^{N} (w_{y_i} - \mathbb{E}(w|f_{i,\ell})) \kappa(x_i, x_j) \tag{20}$$

F.1.1 Self-Attention Head 1

We design $W_K^{(1)}$, $W_Q^{(1)}$, and $W_V^{(1)}$ such that

$$W_K^{(1)} e_{i,\ell} = (0_{d'}, 0_{d'}, 0_{d'}, x_i)^T \tag{21}$$

$$W_Q^{(1)} e_{j,\ell} = (0_{d'}, 0_{d'}, 0_{d'}, x_j)^T \tag{22}$$

$$W_V^{(1)} e_{i,\ell} = (\frac{\alpha}{N}[w_{y_i} - \mathbb{E}(w|f_{i,\ell})], 0_d, 0_{d'}, 0_{d'})^T \tag{23}$$

The output of this first attention head, at position $j \in \{1, \ldots, N+1\}$ is

$$(\frac{\alpha}{N}\sum_{i=1}^{N}(w_{y_i} - \mathbb{E}(w|f_{i,\ell}))\kappa(x_i, x_j), 0_d, 0_{d'}, 0_{d'})^T \tag{24}$$

The output of this first attention head at this first attention layer (before adding the skip connection) is

$$O^{(1)} = \begin{pmatrix} \Delta f_{1,\ell} & \ldots & \Delta f_{N,\ell} & \Delta f_{N+1,\ell} \\ 0_d & \ldots & 0_d & 0_d \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \end{pmatrix} \tag{25}$$

F.1.2 Self-Attention Head 2

With the second attention head we want to add $(0_d, -\mathbb{E}(w|f_j^{(l)}), 0_{d'}, 0_{d'})^T$ from position j, so we clear out the prior expectation. This will provide "scratch space" into which, with the next attention layer type, we will update the expectation, using $f_j^{(l+1)}$. To do this, we design $W_Q^{(2)}$ and $W_K^{(2)}$ such that

$$W_K^{(2)} e_{i,\ell} = \lambda(0_{d'}, 0_{d'}, 0_{d'}, x_i)^T \tag{26}$$

$$W_Q^{(2)} e_{j,\ell} = \lambda(0_{d'}, 0_{d'}, 0_{d'}, x_j)^T \tag{27}$$

where $\lambda \gg 1$. With an RBF kernel, for example (similar things will happen with softmax), if λ is very large,

$$\kappa(W_K^{(2)} e_{i,\ell}, W_Q^{(2)} e_{j,\ell}) = \delta_{i,j} \tag{28}$$

where $\delta_{i,j} = 1$ if $i = j$, and it's zero otherwise.

The value matrix is designed as

$$W_V^{(2)} e_{i,\ell} = (0_d, \mathbb{E}(w|f_{i,\ell}), 0_{d'}, 0_{d'})^T \tag{29}$$

The output of this head is

$$O^{(2)} = \begin{pmatrix} 0_d & \ldots & 0_d & 0_d \\ \mathbb{E}(w|f_{1,\ell}) & \ldots & \mathbb{E}(w|f_{N,\ell}) & \mathbb{E}(w|f_{N+1,\ell}) \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \end{pmatrix} \tag{30}$$

We then add $P^{(1)}O^{(1)} + P^{(2)}O^{(2)}$, with $P^{(1)}$ and $P^{(2)}$ designed so as to yield the cumulative output of the attention

$$O^{(\text{total})} = \begin{pmatrix} \Delta f_{1,\ell} & \ldots & \Delta f_{N,\ell} & \Delta f_{N+1,\ell} \\ -\mathbb{E}(w|f_{1,\ell}) & \ldots & -\mathbb{E}(w|f_{N,\ell}) & -\mathbb{E}(w|f_{N+1,\ell}) \\ 0_d & \ldots & 0_d & 0_d \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \end{pmatrix} \tag{31}$$

This is now added to the skip connection, yielding the total output of this attention layer as

$$T = \begin{pmatrix} f_1^{(l+1)} & \ldots & f_N^{(l+1)} & f_{N+1}^{(l+1)} \\ 0_{d'} & \ldots & 0_{d'} & 0_{d'} \\ w_{y_1} & \ldots & w_{y_N} & 0_{d'} \\ x_1 & \ldots & x_N & x_{N+1} \end{pmatrix} \tag{32}$$

With the first attention layer, with two heads, we update the functions, and we also erase the prior expectations. In the next attention layer, we update the expectations, and place them in the locations of the prior expectations.

F.2 Multi-Layer Perceptron (MLP) Layer

The vectors connected to T above will go into the next layer, which will be characterized by a MLP. Ideally, the MLP should implement the function

$$\mathbb{E}(w|f_{i,\ell+1}) = \sum_{c=1}^{C} w_c \Big[\frac{\exp(w_c^T f_{i,\ell+1})}{\sum\limits_{c'=1}^{C} \exp(w_{c'}^T f_{i,\ell+1})} \Big] \tag{33}$$

to be consistent with functional GD. Let $g_\gamma(f_{i,\ell+1})$ represent an MLP with parameters γ. The same MLP acts on each of the vectors at positions $i = 1, \ldots, N$, corresponding to the first N columns of T, from left. The components of that vector corresponding to $f_{i,\ell+1}$ are input to $g_\gamma(\cdot)$, and the output is a d'-dimensional vector. The output is placed in the position of the zeros in T.

At each layer of the Transformer, the form of the function in (33) is the same. Consequently, within the Transformer implementation, we tie the MLP parameters across all Transformer layer.

G Setup for multiple questions with categorical answers

Consider data of the form (x_i, y_i), where $x_i \in \mathbb{R}^d$ are covariates, and $y_i \in \{1, \ldots, C\}^Q$, with y_i representing answers to Q questions with C categorical answers. A special case is $C = 2$, corresponding to Q yes/no questions. Let $y_{i,m} \in \{1, \ldots, C\}$ represent the qth component of y_i, $q \in \{1, \ldots, Q\}$. We assume that the data are generated from the model

$$p(y_{i,q} = c|X = x_i) = \frac{\exp(f(x_i)^T w_c^{(q)})}{\sum_{c'=1}^{C} \exp(f(x_i)^T w_{c'}^{(q)})} \tag{34}$$

where $\{w_c^{(q)}\}_{q=1,Q}$ represent a set of fixed (learned) vectors, with each $w_c^{(q)} \in \mathbb{R}^{d'}$, and $f(x) \in \mathbb{R}^{d'}$ is a context-dependent latent function. This generalizes our prior setup, which only considered one categorical observation for each x_i, to now consider M such categorical observations.

Assume that we are given contextual data $\{(x_i, y_i)\}_{i=1,N}$, from which we wish to infer $f(x)$, and thereby predict y_{N+1} for a query x_{N+1}. Assuming that the M categorical observations are conditionally independent given $f(x)$, as reflected in (34), then the log-likelihood of the contextual data is

$$\mathcal{L} = \sum_{i=1}^{N} \sum_{q=1}^{Q} \log \Big[\frac{\exp(f(x_i)^T w_c^{(q)})}{\sum_{c'=1}^{C} \exp(f(x_i)^T w_{c'}^{(q)})} \Big] \tag{35}$$

$$= \sum_{i=1}^{N} \sum_{q=1}^{Q} \Big[f(x_i)^T w_c^{(q)} - \log \sum_{c'=1}^{C} \exp(f(x_i)^T w_{c'}^{(q)}) \Big] \tag{36}$$

We assume that $f(x) = A\psi(x)$, where $\psi(x) : \mathbb{R}^d \to \mathbb{R}^D$ is a (generally) nonlinear transformation of the covariates x to a D-dimensional feature space (which could be infinite dimensional), and $A \in \mathbb{R}^{d' \times D}$ is a latent matrix. The matrix A is context-dependent, while $\psi(x)$ is context-independent. We wish to perform context-dependent gradient ascent to infer A.

Gradient ascent applied to this setup yields the following update equation for the latent function:

$$f_{\ell+1}(x) = f_\ell(x) + \alpha \sum_{i=1}^{N} \sum_{q=1}^{Q} [w_{y_{i,q}}^{(q)} - \mathbb{E}(w^{(q)}|f_\ell(x_i))] \kappa(x, x_i) \tag{37}$$

where $\kappa(x_i, x_j) = \psi(x_i)^T \psi(x_j)$, and

$$\mathbb{E}(w^{(q)}|f_\ell(x_i)) = \frac{\sum_{c=1}^{C} w_c^{(q)} \exp(f_\ell(x_i)^T w_c^{(q)})}{\sum_{c'=1}^{C} \exp(f_\ell(x_i)^T w_{c'}^{(q)})} \tag{38}$$

One can rewrite (37) as

$$f_{\ell+1}(x) = f_\ell(x) + \alpha \sum_{i=1}^{N} \frac{1}{Q} \sum_{q=1}^{M} [w_{y_{i,q}}^{(q)} - \mathbb{E}(w^{(q)}|f_\ell(x_i))]\kappa(x, x_i) \tag{39}$$

$$= f_\ell(x) + \alpha \sum_{i=1}^{N} [\bar{w}_i - \mathbb{E}(w|f_\ell(x_i))]\kappa(x, x_i) \tag{40}$$

where

$$\bar{w}_i = \frac{1}{Q} \sum_{q=1}^{Q} w_{y_{i,q}}^{(q)}, \qquad \mathbb{E}(w|f_\ell(x_i)) = \frac{1}{Q} \sum_{q=1}^{Q} \mathbb{E}(w^{(q)}|f_\ell(x_i)) \tag{41}$$

where we see that $\mathbb{E}(w|f_\ell(x_i))$ is an average over expectations.

H Linearization of the Expectation

Consider

$$\mathbb{E}(w|f_{i,\ell}) = \mathbb{E}(w|f_{i,\ell-1} + \Delta f_{i,\ell-1}) \tag{42}$$

$$= \frac{\sum_{c=1}^{C} \exp[w_c^\top f_{i,\ell-1} + w_c^\top \Delta f_{i,\ell-1})]w_c}{\sum_{c'=1}^{C} \exp(w_{c'}^\top f_{i,\ell-1} + w_{c'}^\top \Delta f_{i,\ell-1})} \tag{43}$$

Assuming that $\Delta f_{i,\ell-1}$ makes a small change relative to $f_{i,\ell-1}$, which can be controlled by the learning rate, we may approximate $\mathbb{E}(w|f_{i,\ell})$ by its first-order (linear) Taylor expansion. We have

$$\nabla_f \frac{\exp(w_c^\top f)}{\sum_{c'=1}^{C} \exp(w_{c'}^\top f)} = w_c \frac{\exp(w_c^\top f)}{\sum_{c'=1}^{C} \exp(w_{c'}^\top f)} - \frac{\exp(w_c^\top f)}{[\sum_{c'=1}^{C} \exp(w_{c'}^\top f)]^2} \sum_{c'=1}^{C} w_{c'} \exp(w_{c'}^\top f)$$

$$= \frac{\exp(w_c^\top f)}{\sum_{c'=1}^{C} \exp(w_{c'}^\top f)} \Big[w_c - \mathbb{E}(w|f) \Big] \tag{44}$$

Therefore

$$\mathbb{E}(w|f_{i,\ell}) \approx \mathbb{E}(w|f_{i,\ell-1}) + \sum_{c=1}^{C} w_c \frac{\exp(w_c^\top f_{i,\ell-1})}{\sum_{c'=1}^{C} \exp(w_{c'}^\top f_{i,\ell-1})} [w_c - \mathbb{E}(w|f_{i,\ell-1})]^\top \Delta f_{i,\ell-1} \tag{45}$$

$$= \mathbb{E}(w|f_{i,\ell-1}) + W_e \cdot \text{softmax}(W_e^\top f_{i,\ell-1}) \cdot \tilde{W}_{e,\ell-1} \cdot \Delta f_{i,\ell-1} \tag{46}$$

where the cth column of $\tilde{W}_{e,\ell-1}$ corresponds to $w_c - \mathbb{E}(w|f_{i,\ell-1})$.

For $\ell = 1$, $f_{i,\ell-1} = 0_{d'}$, and therefore $\text{softmax}(W_e^\top f_{i,\ell-1})$ is a uniform C-dimensional probability mass function (PMF), and hence

$$\mathbb{E}(w|f_{i,1}) \approx \frac{1}{C} 1_C \Big[1 + W_e \tilde{W}_{e,0}^\top \Delta f_{i,0} \Big] \tag{47}$$

where the cth column of $\tilde{W}_{e,0}$ is $w_c - \frac{1}{C} 1_C$, where 1_C is a C-dimensional vector of all ones.

For $\ell = 1$, for each i we may approximate $\mathbb{E}(w|f_{i,1}) \approx \mathbb{E}(w|f_{i,0}) + M_1 \Delta f_{i,0}$ where $M_1 = \frac{1}{C} W_e \tilde{W}_e^\top$. Note that this matrix is independent of i. More generally, and with a weaker approximation, we use $\mathbb{E}(w|f_{i,\ell}) \approx \mathbb{E}(w|f_{i,\ell-1}) + M_\ell \Delta f_{i,\ell-1}$. For $\ell > 1$ this approximation is less appropriate, because the above first-order analysis indicates that the linear approximation is *dependent on* i, which we are ignoring. We expect the linear approximation to work best when the first two steps of functional GD reach near conference of GD-based inference.

I GD Parameters for Attention-Only Transformer

Under the linear approximation for $\mathbb{E}(w|f_{i,\ell})$, we can implement Transformer-based inference with a single attention head, as detailed below. Let $\bar{w} = \frac{1}{C} \sum_{c=1}^{C} w_c$, i.e., the average embedding vector

for all C categories. At the input layer, consider the encoding $(x_i, w_{y_i} - \bar{w}, 0_{d'})$, for $i = 1, \ldots, N$, where the $0_{d'}$ vector is positioned where f_i will be updated. Recall that $\bar{w} = \mathbb{E}(w|0_{d'})$, which is the zero-order (initial) approximation to the expectation. The query is encoded as $(x_{N+1}, 0_{d'}, 0_{d'})$.

The Transformer matrices at each layer are

$$W_Q = W_K = \begin{pmatrix} I_{d\times d} & 0_{d\times d'} & 0_{d\times d'} \\ 0_{d'\times d} & 0_{d'\times d'} & 0_{d'\times d'} \\ 0_{d'\times d} & 0_{d'\times d'} & 0_{d'\times d'} \end{pmatrix} \tag{48}$$

such that $W_Q e_{i,\ell} = W_K e_{i,\ell} = (x_i, 0_{d'}, 0_{d'})$ at each layer, where $e_{i,\ell} \in \mathbb{R}^{d+2d'}$ is the vector at position i, output from layer ℓ.

The W_V matrix can be expressed as

$$W_V = \frac{\alpha}{N} \begin{pmatrix} 0_{d\times d} & 0_{d\times d'} & 0_{d\times d'} \\ 0_{d'\times d} & 0_{d'\times d'} & 0_{d'\times d'} \\ 0_{d'\times d} & I_{d'\times d'} & 0_{d'\times d'} \end{pmatrix} \tag{49}$$

and position $i = 1, \ldots, N$ are used for keys and values, where all $i = 1, \ldots, N+1$ positions are used as queries. The output of attention from layer-1 is

$$\sum_{i=1}^{N} W_V e_{i,0} \kappa(W_K e_{i,0}, W_Q e_{j,0}) = \begin{pmatrix} 0_d \\ 0_{d'} \\ \frac{\alpha}{N} \sum_{i=1}^{N} [w_{y_i} - \bar{w}] \kappa(x_i, x_j) \end{pmatrix} \tag{50}$$

where it is understood that the above d'-dimensional vector v_j is positioned as $(0_d, 0_{d'}, v_j)$ in the coordinate system of the Transformer vectors.

Finally, there is an output projection matrix at each layer:

$$P_\ell = \begin{pmatrix} 0_{d\times d} & 0_{d\times d'} & 0_{d\times d'} \\ 0_{d'\times d} & 0_{d'\times d'} & -M_\ell \\ 0_{d'\times d} & 0_{d'\times d'} & I_{d'\times d'} \end{pmatrix} \tag{51}$$

which will update the expectation, to within a linear approximation, with the updated increment to the expectation appended to $\bar{w}$:

$$e_{j,0} + P_1 \sum_{i=1}^{N} W_V e_{i,0} \kappa(W_K e_{i,0}, W_Q e_{j,0}) = \begin{pmatrix} x_j \\ w_{y_j} - \bar{w} - M_1 \Delta f_{j,0} \\ \Delta f_{j,0} \end{pmatrix} \tag{52}$$

where $\Delta f_{j,0} = \frac{\alpha}{N} \sum_{i=1}^{N} [w_{y_i} - \bar{w}] \kappa(x_i, x_j)$.

The latent function $f_{i,\ell}$, for $i = 1, \ldots, N+1$, is updated in the last d' positions of $e_{i,\ell}$ at each layer ℓ, and this is sent into the softmax at the last layer (for position $i = N+1$).