Scalable Set Encoding with Universal Mini-Batch Consistency and Unbiased Full Set Gradient Approximation

Jeffrey Willette [*1] **Seanie Lee** [*1] **Bruno Andreis** [1] **Kenji Kawaguchi** [2] **Juho Lee** [1] **Sung Ju Hwang** [1]

Abstract

Recent work on mini-batch consistency (MBC) for set functions has brought attention to the need for sequentially processing and aggregating chunks of a partitioned set while guaranteeing the same output for all partitions. However, existing constraints on MBC architectures lead to models with limited expressive power. Additionally, prior work has not addressed how to deal with large sets during training when the full set gradient is required. To address these issues, we propose a Universally MBC (UMBC) class of set functions which can be used in conjunction with arbitrary non-MBC components while still satisfying MBC, enabling a wider range of function classes to be used in MBC settings. Furthermore, we propose an efficient MBC training algorithm which gives an unbiased approximation of the full set gradient and has a *constant memory overhead for any set size for both train- and test-time*. We conduct extensive experiments including image completion, text classification, unsupervised clustering, and cancer detection on high-resolution images to verify the efficiency and efficacy of our scalable set encoding framework. Our code is available at github.com/jeffwillette/umbc

1. Introduction

For a variety of problems for which deep models can be applied, unordered sets naturally arise as an input. For example, a set of words in a document (Jurafsky & Martin, 2008) and sets of patches within an image for multiple instance learning (Quellec et al., 2017). Functions which encode sets are commonly known as set encoders, and most previously proposed set encoding functions (Zaheer et al., 2017; Lee et al., 2019) have implicitly assumed that the whole set can fit into memory and be accessed in a single chunk. However, this is not a realistic assumption if it is necessary to process large sets or streaming data. As shown in Figure 2a, Set Transformer (Lee et al., 2019) cannot properly handle streaming data and suffers performance degradation. Please see Figure 8 for more qualitative examples. Bruno et al. (2021) identified this problem, and introduced the mini-batch consistency (MBC) property which dictates that an MBC set encoding model must be able to sequentially process subsets from a partition of a set while guaranteeing the same output over any partitioning scheme, as illustrated in Figure 1. In order to satisfy the MBC property, they devised an attention-based MBC model, the Slot Set Encoder (SSE).

Although SSE satisfies MBC, there are several limitations. First, it has limited expressive power due to the constraints imposed on its architecture. Instead of the conventional softmax attention (Vaswani et al., 2017), the attention of SSE is restricted to using a sigmoid for attention without normalization over the rows of the attention matrix, which may be undesirable for applications requiring convex combinations of inputs. Moreover, the Hierarchical SSE is a composition of pure MBC functions and thus cannot utilize more expressive non-MBC models, such as those utilizing self-attention. Another crucial limitation of the SSE is its limited scalability during training. Training models with large sets requires computing gradients over the full set which can be computationally prohibitive. SSE proposes to randomly sample a small subset for gradient computation, which is a *biased estimator* of the full set gradient as we show in Appendix A.8.

To tackle these limitations of SSE, we propose *Universal MBC (UMBC) set functions* which enable utilizing a broader range of functions while still satisfying the MBC property. Firstly, we relax the restriction to the sigmoid on the activation functions for attention and show that cross-attention with a wider class of activation functions, including the softmax, is MBC. Moreover, we re-interpret UMBC's output as a set, which as we show in Figure 1 and Section 3, *universally* allows for the application of non-MBC set encoders

[*]Equal contribution [1]KAIST [2]National University of Singapore. Correspondence to: Jeffrey Willete <jwillette@kaist.ac.kr>, Seanie Lee <lsnfamily02@kaist.ac.kr>, Sung Ju Hwang <sjhwang82@kaist.ac.kr>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

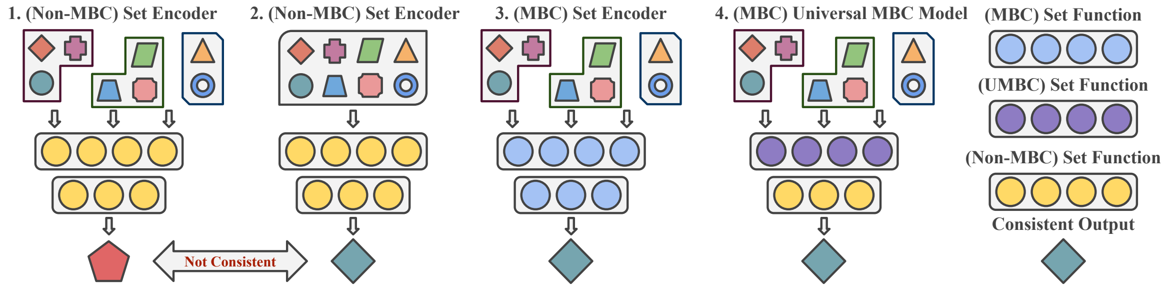
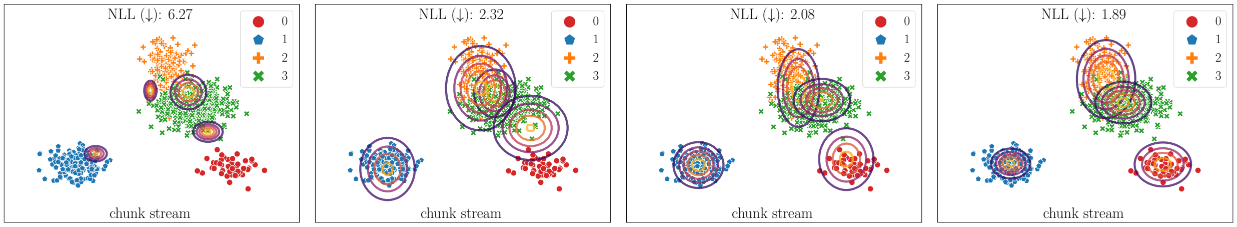


Figure 1. Non-MBC models (1, 2) produce inconsistent outputs when given different set partitions. MBC models (3) produce consistent outputs for any random partition with a specific architecture. UMBC composes both MBC/non-MBC components, expanding the possible set of MBC functions, and allowing for more expressive models.



(a) Set Transformer ✗ (b) Deepsets ✓ (c) Slot Set Encoder ✓ (d) UMBC+ST (Ours) ✓

Figure 2. **Streaming inputs**: A non-MBC model (a) suffers performance degradation in streaming settings. MBC models (b, c, d) can handle streaming inputs consistently. Creating an MBC composition of both MBC/non-MBC components (d) creates the strongest MBC model. A (✓) indicates MBC models while (✗) indicates non-MBC models.

when processing UMBC's output sets, resulting in more expressive functions while maintaining the MBC property. For a concrete example, UMBC used in conjunction with the (non-MBC) Set Transformer (ST) produces consistent output for any partition of a set as shown in Figure 3, and outperforms all other MBC models for clustering streaming data as illustrated in Figure 2.

Lastly, for training MBC models, we propose a novel and scalable algorithm to approximate full set gradient. Specifically, we obtain the full set representation by partitioning the set into subsets and aggregating the subset representations while only considering a portion of the subsets for gradient computation. We find this leads to a constant memory overhead for computing the gradient with a fixed size subset, and is an *unbiased estimator* of the full set gradient.

To verify the efficacy and efficiency of our proposed UMBC framework and full set gradient approximation algorithm, we perform extensive experiments on a variety of tasks including image completion, text classification, unsupervised clustering, and cancer detection on high-resolution images. Furthermore, we theoretically show that UMBC is a universal approximator of continuous permutation invariant functions under some mild assumptions and the proposed training algorithm minimizes the total loss of the full set version by making progress toward its stationary points. We summarize our contributions as follows:

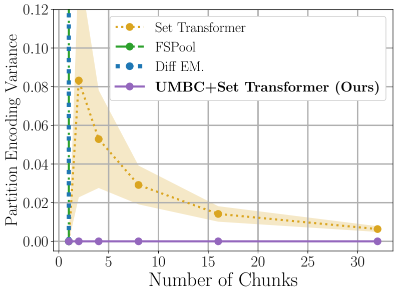


Figure 3. Variance between encodings of 100 different partitions of the same set. UMBC+ST satisfies MBC and thus has no variance.

- We propose a UMBC framework which allows for a broad class of activation functions, including softmax, for attention and also enables utilizing non-MBC functions in conjuction with UMBC while satisfying MBC, resulting in more expressive and less restrictive architectures.
- We propose an efficient training algorithm with a constant memory overhead for any set size by deriving an unbiased estimator of the full set gradient which empirically performs comparably to using the full set gradient.
- We theoretically show that UMBC is a universal approximator to continuous permutation invariant functions under mild assumptions and our algorithm minimizes the full set total loss by making progress toward its stationary points.

2. Related Work

Set Encoding. Deep learning for set structured data has been an active research topic since the introduction of DeepSets (Zaheer et al., 2017), which solidified the requirements of deep set functions, namely permutation equivariant feature extraction and permutation invariant set pooling. Zaheer et al. (2017) have shown that under certain conditions, functions which satisfy the aforementioned requirements

Table 1. Properties of set functions. UMBC models can use arbitrary component set functions and are therefore unconstrained.

Model	MBC	Cross-Attn.	Self-Attn.
DeepSets (Zaheer et al., 2017)	✓	✗	✗
SSE (Bruno et al., 2021)	✓	✓	✗
FSPool (Zhang et al., 2020)	✗	✗	✗
Diff EM (Kim, 2022)	✗	✗	✗
Set Transformer (Lee et al., 2019)	✗	✓	✓
(Ours) UMBC+Any Set Function	✓	✓	✓

act as universal approximators for functions of sets. Subsequently, the Set Transformer (Lee et al., 2019) applied attention (Vaswani et al., 2017) to sets, which has proven to be a powerful tool for set functions. Self-attentive set functions excel on tasks where independently processing set elements may fail to capture pairwise interactions between elements. Subsequent works which utilize pairwise set element interactions include optimal transport (Mialon et al., 2021) and expectation maximization (Kim, 2022). Other notable approaches to permutation invariant set pooling include featurewise sorting (Zhang et al., 2020), and canonical orderings of set elements (Murphy et al., 2019).

Mini-Batch Consistency (MBC). Every method mentioned in the preceding paragraph suffers from an architectural bias which limits them to seeing and processing the whole set in a single chunk. Bruno et al. (2021) identified this problem, and highlighted the necessity for MBC which guarantees that processing and aggregating each subset from a set partition results in the same representation as encoding the entire set at once (Definition 3.2). This is important in settings where the data may not fit into memory due to either large data or limited on-devices resources. In addition to identifying the MBC property, Bruno et al. (2021) also proposed the Slot Set Encoder (SSE) which utilizes cross attention between learnable 'slots' and set elements in conjunction with simple activation functions in order to achieve an MBC model. As shown in Table 1, however, SSE cannot utilize self-attention to model pairwise interactions of set elements due to the constraints imposed on its architecture, which makes it less expressive than the Set Transformer.

3. Method

In this section, we describe the problem we target and provide a formulation for UMBC models along with a derivation of our unbiased full set gradient approximation algorithm. All proofs of theorems are deferred to Appendix A.

3.1. Preliminaries

Let $\mathfrak{X}$ be a d_x-dimensional vector space over $\mathbb{R}$ and let $2^{\mathfrak{X}}$ be the power set of $\mathfrak{X}$. We focus on a collection of finite sets $\mathcal{X}$, which is a subset of $2^{\mathfrak{X}}$ such that $\sup_{X\in\mathcal{X}}|X| \in \mathbb{N}$. We want to construct a parametric function $f_\theta : \mathcal{X} \to \mathcal{Z}$ satisfying permutation invariance. Specifically, given a set $X_i = \{\mathbf{x}_{i,j}\}_{j=1}^{N_i} \in \mathcal{X}$, the output of the function $Z_i = f_\theta(X_i)$ is a fixed sized representation which is invariant to all permutations of the indices $\{1, \dots, N_i\}$. For supervised learning, we define a task specific decoder $g_\lambda : \mathcal{Z} \to \mathbb{R}^{d_y}$ and optimize parameters θ and λ to minimize the loss

$$L(\theta, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \ell((g_\lambda \circ f_\theta)(X_i), y_i) \tag{1}$$

on training data $((X_i, y_i))_{i=1}^{n}$, where y_i is a label for the input set X_i and ℓ denotes a loss function.

Definition 3.1 (Permutation Invariance)**.** Let $\mathfrak{S}_N$ be the set of all permutations of $\{1, \dots, N\}$, *i.e.* $\mathfrak{S}_N = \{\pi : [N] \to [N] \mid \pi \text{ is bijective}\}$ where $[N] := \{1, \dots, N\}$. A function $f_\theta : \mathcal{X} \to \mathcal{Z}$ is permutation invariant *iff* $f_\theta(\{\mathbf{x}_{\pi(1)}, \dots \mathbf{x}_{\pi(N)}\}) = f_\theta(\{\mathbf{x}_1, \dots, \mathbf{x}_N\})$ for all $X \in \mathcal{X}$ and for all permutation $\pi \in \mathfrak{S}_N$.

We further assume that the cardinality of a set X is sufficiently large, such that loading and processing the whole set at once is computationally prohibitive. For non-MBC models, a naïve approach to solve this problem would be to encode a small subset of the full set as an approximation, leading to a possibly suboptimal representation of the full set. Instead, Bruno et al. (2021) propose a *mini-batch consistent* (MBC) set encoder, the Slot Set Encoder (SSE), to piecewise process disjoint subsets of the full set and aggregate them to obtain a consistent full set representation.

Definition 3.2 (Mini-Batch Consistency)**.** We say a function f_θ is mini-batch consistent *iff* for any $X \in \mathcal{X}$, there is a function h such that for any partition $\zeta(X)$ of the set X,

$$f_\theta(X) = h\left(\{f_\theta(S) \in \mathcal{Z} \mid S \in \zeta(X)\}\right). \tag{2}$$

Models which satisfy the MBC property can partition a set into subsets, encode, and then aggregate the subset representations to achieve the exact same output as encoding the full set. Due to constraints on the architecture of the SSE, however, on certain tasks the SSE shows weaker performance than non-MBC set encoders such as Set Transformer (Lee et al., 2019) which utilizes self attention. To tackle this limitation, we propose Universal MBC (UMBC) set encoders which are both MBC and also allow for the use of arbitrary non-MBC set functions while still satisfying MBC property.

3.2. Universal Mini-Batch Consistent Set Encoder

In this section, we provide a formulation of our UMBC set encoder f_θ. Given an input set $X \in \mathcal{X}$, we represent it as a matrix $X = [\mathbf{x}_1 \cdots \mathbf{x}_N]^\top \in \mathbb{R}^{N\times d_x}$ whose rows are elements in the set, and independently process each element with $\phi : \mathbb{R}^{d_x} \to \mathbb{R}^{d_h}$ as $\Phi(X) = [\phi(\mathbf{x}_1) \cdots \phi(\mathbf{x}_N)]^\top$, where ϕ is a deep feature extractor. We then compute the un-normalized attention score between a set of learnable slots $\Sigma = [\mathbf{s}_1 \cdots \mathbf{s}_k]^\top \in \mathbb{R}^{k \times d_s}$ and $\Phi(X)$ as:

$$Q = \texttt{LN}(\Sigma W^Q), K(X) = \Phi(X)W^K, V(X) = \Phi(X)W^V$$
$$\hat{A} = \sigma\left(\sqrt{d^{-1}} \cdot QK(X)^\top\right) \in \mathbb{R}^{k \times N}, \tag{3}$$

where σ is an element-wise activation function with $\sigma(x) \gneq 0$ for all $x \in \mathbb{R}$, $\texttt{LN}$ denotes layer normalization (Ba et al., 2016), and $W^Q \in \mathbb{R}^{d_s \times d}, W^K \in \mathbb{R}^{d_h \times d}, W^V \in \mathbb{R}^{d_h \times d}$ are parameters which are part of θ. For simplicity, we omit biases for Q, K, and V. With the un-normalized attention score $\hat{A}$, we can define a map

$$\hat{f}_\theta : X \in \mathbb{R}^{N \times d_x} \mapsto \nu_p(\hat{A})V(X) \in \mathbb{R}^{k \times d} \tag{4}$$

for $p = 1, 2$, where $\nu_p : \mathbb{R}^{k \times d} \to \mathbb{R}^{k \times d}$ is defined by either $\nu_1(\hat{A})_{i,j} = \hat{A}_{i,j} / \sum_{i=1}^k \hat{A}_{i,j}$ which normalizes the columns or the identity mapping $\nu_2(\hat{A})_{i,j} = \hat{A}_{i,j}$. The choice of ν_p depends on the desired activation function σ. Alternatively, similar to slot attention (Locatello et al., 2020), we can make the function stochastic by sampling $\mathbf{s}_i \sim \mathcal{N}(\mu_i, \text{diag}(\texttt{softplus}(\mathbf{v}_i)))$ with reparameterization (Kingma & Welling, 2013) for $i = 1, \ldots, k$, where $\mu_i \in \mathbb{R}^{d_s}, \mathbf{v}_i \in \mathbb{R}^{d_s}$ are part of the parameters θ. If we sample $\mathbf{s}_1, \ldots, \mathbf{s}_k \overset{\text{iid}}{\sim} \mathcal{N}(\mu_1, \text{diag}(\texttt{softplus}(\mathbf{v}_1)))$ with a sigmoid for σ and ν_1 for normalization, and then apply a pooling function (sum, mean, min, or max) to the columns of $[\hat{f}_\theta(X)_1^\top \cdots \hat{f}_\theta(X)_k^\top]^\top \in \mathbb{R}^{k \times d}$, we achieve a function equivalent to the SSE, where $\hat{f}_\theta(X)_i$ is i-th row of $\hat{f}_\theta(X)$.

However, SSE has some drawbacks. First, since the attention score of $\nu_p(\hat{A})_{i,j}$ is independent to the other $N-1$ attention scores $\nu_p(\hat{A})_{i,l}$ for $l \neq j$, it is impossible for the rows of $\nu_p(\hat{A})$ to be convex coefficients as the softmax outputs in conventional attention (Vaswani et al., 2017). Notably, in some of our experiments, the constrained attention activation originally used in the SSE, which we call slot-sigmoid, significantly degrades generalization performance. Furthermore, stacking hierarchical SSE layers has been shown to harm performance (Bruno et al., 2021), which limits the power of the overall model.

To overcome these limitations of the SSE, we propose a Universal Mini-Batch Consistent (UMBC) set encoder f_θ by allowing the set function f_θ to also use arbitrary non-MBC functions. Firstly, we propose normalizing the attention matrix $\nu_p(\hat{A})$ over rows to consider dependency among different elements of the set in the attention operation:

$$\bar{f}_\theta : X \in \mathbb{R}^{N \times d_x} \mapsto \nu_p(\hat{A})\mathbf{1}_N \in \mathbb{R}^k \tag{5}$$
$$f_\theta : X \in \mathbb{R}^{N \times d_x} \mapsto \text{diag}\left(\bar{f}_\theta(X)\right)^{-1} \hat{f}_\theta(X) \in \mathbb{R}^{k \times d} \tag{6}$$

where $\mathbf{1}_N = (1, \ldots, 1) \in \mathbb{R}^N$. We prove that a UMBC set encoder f_θ is *permutation invariant, equivariant, and MBC.*

Theorem 3.3. *A UMBC function is permutation invariant.*

Any strictly positive elementwise function is a valid σ. For an instance, if we use the identity mapping ν_2 with $\sigma(\cdot) := \exp(\cdot)$, the attention matrix $\text{diag}(\bar{f}_\theta(X))^{-1}\nu_p(\hat{A})$ is equivalent to applying the softmax to each row of $\hat{A}$, which is hypothesized to break the MBC property by Bruno et al. (2021). However, we show that this does not break the MBC property in Appendix A.2. Intuitively, since

$$\hat{f}_\theta(X) = \sum_{S \in \zeta(X)} \hat{f}_\theta(S), \text{ and } \bar{f}_\theta(X) = \sum_{S \in \zeta(X)} \bar{f}_\theta(S) \tag{7}$$

holds for any partition $\zeta(X)$ of the set X, we can iteratively process each subset $S \in \zeta(X)$ and aggregate them without losing any information of $f_\theta(X)$, *i.e.*, f_θ is MBC even when normalizing over the N elements of the set. Note that the operation outlined above is mathematically equivalent to the softmax, but uses a non-standard implementation. We discuss the implementation and list 5 such valid attention activation functions which satisfy the MBC property in Appendix I.

Theorem 3.4. *Given the slots $\Sigma = [\mathbf{s}_1 \cdots \mathbf{s}_k]^\top \in \mathbb{R}^{k \times d_s}$, a UMBC set encoder is mini-batch consistent.*

Lastly, we may consider the output of a UMBC set encoder $f_\theta(X)$ as either a fixed vector or a set of k elements. Under the set interpretation, we may therefore apply subsequent functions on the set of cardinality k. To provide a valid input to subsequent set encoders, it is sufficient to view UMBC as a set to set function $\varphi(\Sigma; X, \theta) : \Sigma \mapsto f_\theta(X)$ for each set $X \in \mathcal{X}$, which is permutation equivariant w.r.t. the slots Σ.

Definition 3.5. A function $\varphi : \mathbb{R}^{k \times d_s} \to \mathbb{R}^{k \times d}$ is said to be permutation equivariant *iff* $\varphi([\Sigma_{\pi(1)}^\top \cdots \Sigma_{\pi(k)}^\top]^\top) = [\varphi(\Sigma)_{\pi(1)}^\top \cdots \varphi(\Sigma)_{\pi(k)}^\top]^\top$ for all $\Sigma \in \mathbb{R}^{k \times d_s}$ and for all $\pi \in \mathfrak{S}_k$, where $\mathfrak{S}_k = \{\pi : [k] \to [k] \mid \pi \text{ is bijective}\}$ contains all permutations of $\{1, \ldots, k\}$, and $\varphi(\Sigma)_i, \Sigma_i$ denote i-th row of $\varphi(\Sigma)$ and Σ, respectively.

Theorem 3.6. *For each input $X \in \mathcal{X}$, $\varphi(\Sigma; X, \theta) : \Sigma \mapsto f_\theta(X)$ is equivariant w.r.t. permutations of the slots Σ.*

A key insight is that we can leverage non-MBC set encoders such as Set Transformer after a UMBC layer to improve expressive power of an MBC model while still satisfying MBC (Definition 3.2). As a result, with some assumptions, a UMBC set encoder used in combination with any continuously sum decomposable (Zaheer et al., 2017) permutation invariant deep neural network is a universal approximator of the class of continuously sum decomposable functions.

Theorem 3.7. *Let $d_x = 1$ and restrict the domain $\mathcal{X}$ to $[0, 1]^M$. Suppose that the nonlinear activation function of ϕ has nonzero Taylor coefficients up to degree M. Then, UMBC used in conjunction with any continuously sum-decomposable permutation-invariant deep neural network with nonlinear activation functions that are* *not polynomials of finite degrees is a universal approximator of the class of functions $\mathcal{F} = \{f : [0,1]^M \to \mathbb{R} \mid f \text{ is continuous and permutation invariant}\}$.*

Although we use a non-MBC set encoder on top of UMBC, this *does not* violate the MBC property. Since we may obtain $f_\theta(X)$ by sequentially processing each subset of X and the resulting set with cardinality k is assumed small enough to load $f_\theta(X)$ in memory, we can directly provide the MBC output of UMBC to the non-MBC set encoder.

Corollary 3.8. *Let $\hat{g}_\omega : \mathbb{R}^{k\times d} \to \mathcal{Z}$ be a (non-MBC) set encoder and let $f_\theta : \mathcal{X} \to \mathbb{R}^{k\times d}$ be a UMBC set encoder. Then $\hat{g}_\omega \circ f_\theta$ is mini-batch consistent.*

For notational convenience, we write g_λ to indicate the composition $g_\lambda \circ \hat{g}_\omega$ of a set encoder $\hat{g}_\omega : \mathbb{R}^{k\times d} \to \mathcal{Z}$ and a decoder $g_\lambda : \mathcal{Z} \to \mathbb{R}^{d_y}$, throughout the paper. Similarly, the parameter λ denotes (ω, λ).

3.3. Stochastic Approximation of the Full Set Gradient

Although we can leverage SSE or UMBC at test-time by sequentially processing subsets to obtain the full set representation $f_\theta(X)$, at train-time it is infeasible to utilize the gradient of the loss (equation 1) w.r.t. the full set. Computation of the full set gradient with automatic differentiation requires storing the entire computation graph for all forward passes of each subset S from $\zeta(X)$ denoted as a partition of a set X, which incurs a prohibitive computational cost for large sets. As a simple approximation, Bruno et al. (2021) propose randomly sampling a single subset $S_{i,j} \in \zeta(X_i)$ and computing the gradient of the loss $\ell((g_\lambda \circ f_\theta)(S_{i,j}), y_i)$ based on a single subset at each iteration.

Remark 3.9. Let $\zeta(X_i)$ be a partition of set $X_i \in \mathcal{X}$ and $S_{i,j} \in \zeta(X_i)$ be a subset of X_i. Then the gradient of $1/n \sum_{i=1}^n \ell((g_\lambda \circ f_\theta)(S_{i,j}), y_i)$ is a *biased estimation* of the full set gradient and leads to a suboptimal solution in our experiments. Please see Appendix A.8 for further details.

In order to tackle this issue, we propose an *unbiased estimation* of the full set gradient which incurs a constant memory overhead. Firstly, we uniformly and independently sample a mini-batch $((\bar{X}_i, \bar{y}_i))_{i=1}^m$ from the training dataset $((X_i, y_i))_{i=1}^n$ for every iteration $t \in \mathbb{N}_+$. We denote this process by $((\bar{X}_i, \bar{y}_i))_{i=1}^m \sim D[((X_i, y_i))_{i=1}^n]$. Then, for each $\bar{X}_i$, we sample a mini-batch $\bar{\zeta}_t(\bar{X}_i) = \{\bar{S}_1, \ldots, \bar{S}_{|\bar{\zeta}_t(\bar{X}_i)|}\}$ from the partition $\zeta_t(\bar{X}_i) = \{S_1, \ldots, S_{|\zeta_t(\bar{X}_i)|}\}$ of $\bar{X}_i$, *i.e.*, all $\bar{S}_j$ are drawn independently and uniformly from $\zeta_t(\bar{X}_i)$. Denote this process by $\bar{\zeta}_t(\bar{X}_i) \sim D[\zeta_t(\bar{X}_i)]$. Instead of storing the computational graph of all forward passes of subsets in the partition $\zeta_t(\bar{X}_i)$ of a set $\bar{X}_i$, we apply `StopGrad` to all subsets $S \notin \bar{\zeta}_t(\bar{X}_i)$ as follows:

$$\hat{f}_\theta^{\bar{\zeta}_t,\zeta_t}(\bar{X}_i) = \sum_{S\in\bar{\zeta}_t(\bar{X}_i)} \hat{f}_\theta(S) + \texttt{StopGrad}\Big(\sum_{S\in\zeta_t(\bar{X}_i)\setminus\bar{\zeta}_t(\bar{X}_i)} \hat{f}_\theta(S)\Big) \quad (8)$$

$$\bar{f}_\theta^{\bar{\zeta}_t,\zeta_t}(\bar{X}_i) = \sum_{S\in\bar{\zeta}_t(\bar{X}_i)} \bar{f}_\theta(S) + \texttt{StopGrad}\Big(\sum_{S\in\zeta_t(\bar{X}_i)\setminus\bar{\zeta}_t(\bar{X}_i)} \bar{f}_\theta(S)\Big) \quad (9)$$

where, for any function $q : \theta \mapsto q(\theta)$, the symbol $\texttt{StopGrad}(q(\theta))$ denotes a constant with its value being $q(\theta)$, *i.e.*, $\partial\texttt{StopGrad}(q(\theta))/\partial\theta = 0$. For simplicity, we omit the superscript ζ_t if there is no ambiguity. Finally, we update both the parameter θ and λ of the respective encoder and decoder using the gradient of the following functions, respectively at $t \in \mathbb{N}_+$:

$$L_{t,1}(\theta,\lambda) = \frac{1}{m}\sum_{i=1}^m \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\ell(g_\lambda(f_\theta^{\bar{\zeta}_t}(\bar{X}_i)), y_i) \quad (10)$$

$$L_{t,2}(\theta,\lambda) = \frac{1}{m}\sum_{i=1}^m \frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\ell(g_\lambda(f_\theta^{\bar{\zeta}_t}(\bar{X}_i)), y_i). \quad (11)$$

We outline our proposed training method in Algorithm 1. Note that we can apply our algorithm to any set encoder for which a full set representation can be decomposed into a summation of subset representations as in equation 7 such as Deepsets with sum or mean pooling, or SSE which are in fact special cases of UMBC. Furthermore, we can apply the algorithm to any differentiable non-MBC set encoder if we simply place a UMBC layer before the non-MBC function. As a consequence of the $\texttt{StopGrad}()$ operation, if we set $|\bar{\zeta}_t(\bar{X}_i)| = 1$, our method incurs the same computation graph storage cost as randomly sampling a single subset. Moreover, $\partial L_{t,1}(\theta,\lambda)/\partial\theta$ and $\partial L_{t,2}(\theta,\lambda)/\partial\lambda$ are unbiased estimators of $\partial L(\theta,\lambda)/\partial\theta$ and $\partial L(\theta,\lambda)/\partial\lambda$, respectively.

Theorem 3.10. *For any $t \in \mathbb{N}_+$, $\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}$ and $\frac{\partial L_{t,2}(\theta,\lambda)}{\partial\lambda}$ are unbiased estimators of $\frac{\partial L(\theta,\lambda)}{\partial\theta}$ and $\frac{\partial L(\theta,\lambda)}{\partial\lambda}$ as follows:*

$$\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right] = \frac{\partial L(\theta,\lambda)}{\partial\theta} \quad (12)$$

$$\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,2}(\theta,\lambda)}{\partial\lambda}\right] = \frac{\partial L(\theta,\lambda)}{\partial\lambda}, \quad (13)$$

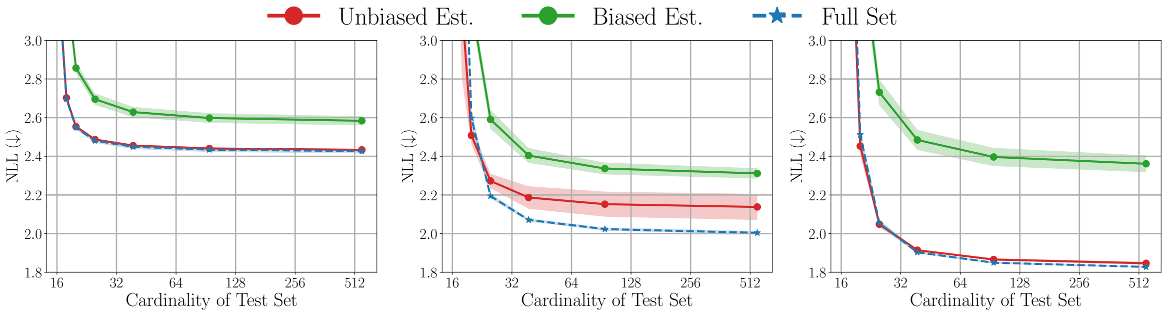
where the first expectation is taken for $((\bar{X}_i, \bar{y}_i))_{i=1}^m \sim D[((X_i, y_i))_{i=1}^n]$, and the second expectation is taken for $\bar{\zeta}_t(\bar{X}_i) \sim D[\zeta_t(\bar{X}_i)]$ for all $i \in \{1, \ldots, m\}$.

Under mild conditions, Theorem 3.10 implies that the updating θ with our method makes progress towards minimizing $L(\theta,\lambda)$. We formalize this statement in Appendix B.

4. Experiments

4.1. Amortized Clustering

We consider amortized clustering on a dataset generated from Mixture of K Gaussians (MoGs) (See Appendix C for



(a) DeepSets (b) SSE (c) UMBC + ST (Ours)

Figure 4. Performance of **(a)** DeepSets **(b)** SSE, and **(c)** UMBC with varying set sizes for amortized clustering on Mixtures of Gaussians. The unbiased estimate of the full set gradient outperforms the biased estimate, and is usually indistinguishable from the full set gradient.

Table 2. Clustering: NLL, mem. usage and wall clock time.

Model	MBC	NLL(↓)	Memory (Kb)	Time (Ms)
DeepSets	✓	2.43 ± 0.004	16	0.46 ± 0.07
SSE	✓	2.13 ± 0.067	61	0.83 ± 0.08
SSE (Hierarchical)	✓	2.38 ± 0.057	125	0.84 ± 0.06
FSPool	✗	3.52 ± 0.192	43	0.79 ± 0.08
Diff EM	✗	5.58 ± 0.966	476	7.11 ± 0.31
Set Transformer (ST)	✗	11.6 ± 2.180	225	2.26 ± 0.135
UMBC + FSPool	✓	2.01 ± 0.027	70	1.18 ± 0.10
UMBC + Diff EM	✓	2.13 ± 0.084	502	8.57 ± 0.33
UMBC + ST	✓	$\mathbf{1.84} \pm 0.008$	100	1.63 ± 0.11

dataset construction details). Given a set $X_i = \{\mathbf{x}_{i,j}\}_{j=1}^{N_i}$ sampled from a MoGs, the goal is to output the mixing coefficients, and Gaussian mean and variance, which minimizes the negative log-likelihood of the set as follows:

$$\{\pi_j(X_i), \mu_j(X_i), \Sigma_j(X_i)\}_{j=1}^{K} = f_\theta(X_i) \tag{14}$$

$$\ell(f_\theta(X_i)) = -\sum_{l=1}^{N_i} \log \sum_{j=1}^{K} \pi_j(X_i)\mathcal{N}(\mathbf{x}_{i,l}; \Theta_j(X_i)) \tag{15}$$

where $\Theta_j(X_i) = (\mu_j(X_i), \Sigma_j(X_i))$ denotes a mean vector and a diagonal covariance matrix for j-th Gaussian, and $\pi_j(X_i)$ is j-th mixing coefficient. Note that there is no label y_i since it is an unsupervised clustering problem. We optimize the parameters of the set encoder θ to minimize the loss over a batch, $L(\theta) = 1/n \sum_{i=1}^{n} \ell(f_\theta(X_i))$.

Setup. We evaluate training with the full set gradient vs. the unbiased estimation of the full set gradient. In this setting, for gradient computation, MBC models use a subset of 8 elements from a full set of 1024 elements. Non MBC models such as Set Transformer (ST), FSpool (Zhang et al., 2020), and Diff EM (Kim, 2022) are also trained with the set of 8 elements. We compare our UMBC model against Deepsets, SSE, Set Transformer, FSPool, and Diff EM. Note that at test-time all non-MBC models process every 8 element subset from the full set independently and aggregate the representations with mean pooling.

Results. In Figure 4, interestingly, the unbiased estimation of the full set gradient (**red**) is almost indistinguishable from the full set gradient (**blue**) for DeepSets and UMBC, while there is a significant gap for SSE. In all cases, the unbiased estimation of the full set gradient outperforms training with the biased gradient approximation with only the set of 8 elements per random sample (**green**), which is proposed by Bruno et al. (2021). Lastly, as shown in Table 2, we compare all models in terms of generalization performance (NLL), memory usage, and wall-clock time for processing a single subset. All non-MBC models show underperformance due to their violation of the MBC property. However, if we utilize 'UMBC+' compositions, the composition becomes MBC with significantly improved performance and little added overhead for memory and time complexity. In contrast, a composition of pure MBC functions, the Hierarchical SSE degrades the performance of SSE. Notably, UMBC with Set Transformer outperforms all other models whereas Set Transformer alone achieves the worst NLL. These results verify expressive power of UMBC in conjuction with non-MBC models.

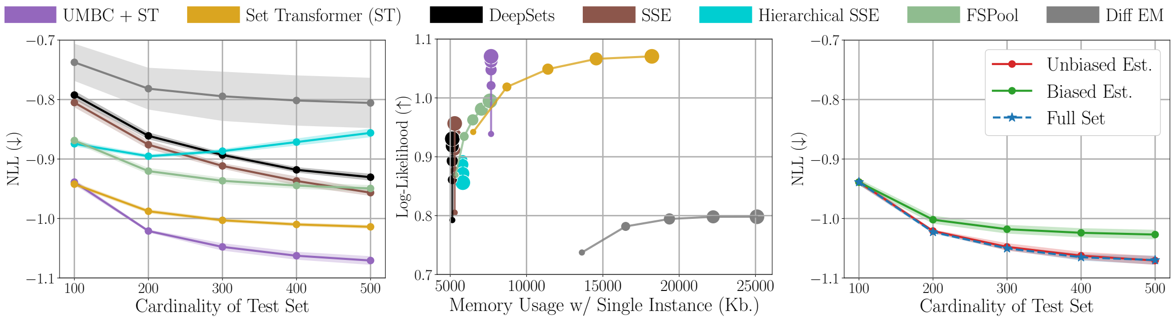
4.2. Image Completion

In this task, we are given a set of M RGB pixel values $y_i \in \mathbb{R}^3 \times \cdots \times \mathbb{R}^3$ of an image as well as the corresponding 2-dimensional coordinates $(\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,M})$ normalized to be in $[0,1]^2 \times \cdots \times [0,1]^2$. The goal of the task is to predict RGB pixel values of all M coordinates of the image. Specifically, given a context set $X_i = \{(\mathbf{x}_{i,c_j}, y_{i,c_j})\}_{j=1}^{N_i}$ processed by the set encoder, we obtain the set representation $f_\theta(X_i) \in \mathbb{R}^{k \times d}$. Then, a decoder $g_\lambda : \mathbb{R}^{k \times d} \times [0,1]^2 \times \cdots \times [0,1]^2 \to \mathbb{R}^6 \times \cdots \times \mathbb{R}^6$ which utilizes both the set representation and the target coordinates, learns to predict a mean and variance for each coordinate of the image as $((\hat{\mu}_{i,j,l})_{l=1}^3, (\hat{\sigma}_{i,j,l})_{l=1}^3)_{j=1}^M = g_\lambda(\mathbf{z}_i)$, where $\mathbf{z}_i = (f_\theta(X_i), \mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,M})$. Then we compute the negative log-likelihood of the label set $y_i = ((y_{i,j,l})_{l=1}^3)_{j=1}^M$:

$$\ell(g_\lambda(\mathbf{z}_i), y_i) = -\sum_{j=1}^{M} \sum_{l=1}^{3} \log \mathcal{N}(y_{i,j,l}; \hat{\mu}_{i,j,l}, \hat{\sigma}_{i,j,l}), \tag{16}$$

where $\mathcal{N}(\cdot; \mu, \sigma)$ is a univariate Gaussian probability density function. Finally, we optimize θ and λ to minimize the loss $L(\theta, \lambda) = 1/n \sum_{i=1}^{n} \ell(g_\lambda(\mathbf{z}_i), y_i)$.

Setup. In our experiments, we impose the restriction that a set encoder is only allowed to compute the gradient with 100 elements of a context set X_i during training and the model



(a) (b) (c)

Figure 5. **(a)** Comparison of different models with varying sizes of sets. **(b)** Memory usage for models to process *a single set* with different cardinalities denoted as the size of the circles. MBC models with our unbiased full set gradient approximation can process any set size with a **constant memory** overhead. **(c)** The effect of our algorithm compared to training with a small random subset and the full set.

can only process 100 elements of the context set at once at test time. We train the set encoders in a Conditional Neural Process (Garnelo et al., 2018) framework, using 32 × 32 images from the CelebA dataset (Liu et al., 2015). We vary the cardinality of the context set size $N_i \in \{100, \dots, 500\}$ and compare the negative log-likelihood (NLL) of each model. For baselines, we compare our UMBC set encoder against: Deepsets, SSE, Hierarchical SSE, Set Transformer (ST), FSPool, and Diff EM. For our UMBC, we use the softmax for σ in equation 3 and place the ST after the UMBC layer.

Results. First, as shown in Figure 5a, our **UMBC + ST** model outperforms all baselines, empirically verifying the expressive power of UMBC. SSE underperforms in terms of NLL due to its constrained architecture. Moreover, stacking hierarchical SSE layers degrades the performance of SSE for larger sets. Note that all MBC set encoders (**Deepsets**, **SSE**, **Hierarchical SSE** and **UMBC**) in Figure 5a are trained with our proposed unbiased gradient approximation in Algorithm 1. On the other hand, we train non-MBC models such as **Set Transformer (ST)**, **FSPool**, and **Diff EM** with a randomly sampled subset of 100 elements, and perform mean pooling over all subset representations at test-time to approximate an MBC model.

Additionally, Figure 5b shows GPU memory usage for each model while processing sets of varying cardinalities without a memory constraint. The marker size is proportional to the set cardinality. Notably, all four MBC models incur a *constant memory* overhead to process any set size, as we can apply `StopGrad` to most of the subset, and compute an unbiased gradient estimate with a fixed sized subset (100). However, memory overhead for all non-MBC models is a function of set size. Thus, Set Transformer uses more than twice the memory of UMBC to achieve a similar log-likelihood on a set of 500 elements.

Lastly, in Figure 5c, we show how our proposed unbiased training algorithm (**red**) improves the generalization performance of UMBC models compared to training with a limited subset of 100 elements (**green**). Notably, the performance of our algorithm is indistinguishable from that of training models with the full set gradient (**blue**). We present similar plots for Deepsets and SSE in Figures 11a and 11b. Across all models, our training algorithm significantly and consistently improves performance compared to training with random subsets of 100 elements, while requiring the same amount of memory. These empirical results verify both efficiency and effectiveness of our proposed method.

Table 3. Micro F1 score and memory usage on Quadro RTX8000 for long document classification with inverted EURLEX dataset.

Model	F1	MBC	Memory (MB)
Longformer	56.47 ±0.43	✗	25,185
ToBERT	67.11 ±0.87	✗	38,563
DeepSets w/ 100	59.97±0.59	✓	1,295
DeepSets w/ full	60.82±0.58	✓	7,317
SSE w/ 100	67.60±0.17	✓	1,319
SSE w/ full	67.91±0.33	✓	6,799
UMBC + BERT w/ 100	**70.48**±0.23	✓	4,909
UMBC + BERT w/ full	**70.23**±0.84	✓	11,497

4.3. Long Document Classification

In this task, we are given a long document $X_i = (\mathbf{x}_{i,1}, \dots, \mathbf{x}_{i,N_i})$ consisting of an average of 707.99 words. The goal of this task is to predict a binary multi-label $y_i = (y_{i,1}, \dots, y_{i,c}) \in \{0, 1\}^c$ of the document, where c is the number of classes. We ignore the order of words and consider the document as a multiset of words, *i.e.*, a set allowing duplicate elements. Specifically, given a training dataset $((X_i, y_i))_{i=1}^n$, we process each set X_i with the set encoder to obtain the set representation $f_\theta(X_i) \in \mathbb{R}^{k \times d}$. We then use a decoder $g_\lambda : \mathbb{R}^{k \times d} \to \mathbb{R}^c$ to output the probability of each class and compute the cross entropy loss:

$$\ell(\mathbf{z}_i, y_i) = -\sum_{j=1}^{c} \left(y_{i,j} \log \tilde{\sigma}(z_{i,j}) + (1 - y_{i,j}) \log(1 - \tilde{\sigma}(z_{i,j}))\right) \tag{17}$$

where $\mathbf{z}_i = (z_{i,1}, \dots, z_{i,c}) = (g_\lambda \circ f_\theta)(X_i)$ and $\tilde{\sigma}$ denotes the sigmoid function. Finally we optimize θ and λ to minimize the loss $L(\theta, \lambda) = 1/n \sum_{i=1}^n \ell(\mathbf{z}_i, y_i)$.

Setup. All models are trained on the inverted EURLEX dataset (Chalkidis et al., 2019) consisting of long legal documents divided into sections. The order of sections are inverted following prior work (Park et al., 2022). To predict

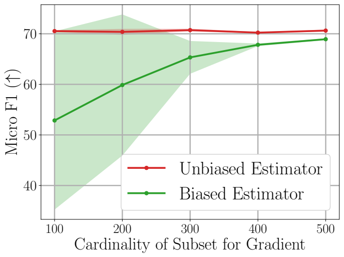


Figure 6. F1 score with varying the size of subset for gradient computation for our method and random sampling.

Table 4. Acc. and AUC of each model on the Camelyon16 dataset.

Model	MBC	Accuracy	AUROC	Accuracy	AUROC
		Pretrain		MBC Finetune	
DS-MIL	✗	$86.36_{\pm 0.88}$	$0.866_{\pm 0.00}$	-	-
AB-MIL	✗	$86.82_{\pm 0.00}$	$\underline{0.884_{\pm 0.01}}$	-	-
DeepSets	✓	$82.02_{\pm 0.65}$	$0.848_{\pm 0.01}$	$82.02_{\pm 0.65}$	$0.848_{\pm 0.01}$
SSE	✓	$74.73_{\pm 1.04}$	$0.748_{\pm 0.03}$	$74.57_{\pm 1.27}$	$0.755_{\pm 0.04}$
UMBC + ST	✓	$\underline{87.91_{\pm 1.41}}$	$0.874_{\pm 0.01}$	$\mathbf{88.84}_{\pm \mathbf{0.88}}$	$\mathbf{0.892}_{\pm \mathbf{0.01}}$

a label, we give the whole document to the model without any truncation. We compare the micro F1 of each model.

We compare UMBC to Deepsets, SSE, ToBERT (Pappagari et al., 2019), and Longformer (Beltagy et al., 2020). For Deepsets and SSE, we use the pre-trained word embedding from BERT (Devlin et al., 2019) without positional encoding and 2 layer fully connected (FC) networks for feature extractor ϕ. We use another 3 layer FC network for the decoder. For UMBC, we use the same feature extractor as SSE but instead use the pre-trained BERT as a decoder, with a randomly initialized linear classifier. We remove the positional encoding of BERT for UMBC to ignore word order. For all the MBC models, we train them both with full set denoted as "w/ full" and with our gradient approximation method on a subset of 100 elements denoted as "w/ 100".

Results. As shown in Table 3, our proposed UMBC outperforms all baselines including non-MBC models — Longformer and ToBERT which require excessive amounts of GPU memory for training models with long sequences. This result again verifies the expressive power of UMBC with BERT (a non-MBC model) for long document classification. Moreover, with significantly less GPU memory, all MBC models (Deepsets, SSE, and UMBC) trained with our unbiased gradient approximation using a subset of 100 elements, achieve similar performance to the models trained with full set. Lastly, in Figure 6, we plot the micro F1 score as a function of the cardinality of the subset used for gradient computation when training the UMBC model. Our proposed unbiased gradient approximation (**red**) shows consistent performance for all subset cardinalities. In contrast, training the model with a small random subset (**green**) is unstable, resulting in underperformance and higher F1 variance.

4.4. Multiple Instance Learning (MIL)

In MIL, we are given a 'bag' of instances with a corresponding bag label, but no labels for each instance within the bag. Labels should not depend on the order of the instances, *i.e.*, MIL can be recast as a set classification problem. Specifically, given a set $X_i = \{\mathbf{x}_{i,j}\}_{j=1}^{N_i}$, the goal is to predict its binary label $y_i \in \{0, 1\}$. For this task, we obtain two streams of set representations and compute the cross entropy loss from the decoder $g_\lambda : \mathbb{R}^{k \times d} \to \mathbb{R}$ output as:

$$\mathbf{z}_{i,1} = \max\{w^\top \phi(\mathbf{x}_{i,j}) + b\}_{j=1}^{N_i}, \quad \mathbf{z}_{i,2} = f_\theta(X_i) \quad (18)$$

$$\mathcal{L}_i = \frac{1}{2}\left(\ell(\mathbf{z}_{i,1}, y_i) + \ell(g_\lambda(\mathbf{z}_{i,2}), y_i)\right), \quad (19)$$

where $w \in \mathbb{R}^{d_h}$ and $b \in \mathbb{R}$ are parameters and ℓ is the cross entropy loss described in equation 17 with $c = 1$. We optimize all parameters θ, λ, w, and b to minimize the loss $1/n \sum_{i=1}^n \mathcal{L}_i$. At test time, we predict a label y_* for a set X_* as:

$$p_* = \frac{1}{2}\left(\tilde{\sigma}(\mathbf{z}_{*,1}) + \tilde{\sigma}(g_\lambda(\mathbf{z}_{*,2}))\right), \; y_* = \mathbb{1}\{p_* \geq \tau\}, \quad (20)$$

where $\mathbf{z}_{*,1} = \max\{w^\top \phi(\mathbf{x}) : \mathbf{x} \in X_*\}$, $\mathbf{z}_{*,2} = f_\theta(X_*)$, $\tilde{\sigma}$ is the sigmoid function, $\mathbb{1}$ is indicator function and $0 < \tau < 1$ is threshold tuned on the validation set.

Setup. We evaluate all models on the Camelyon16 Whole Slide Image cancer detection dataset (Bejnordi et al., 2017). Each instance consists of a high resolution image of tissue from a medical scan which is pre-processed into 256 × 256 patches of RGB pixels. After pre-processing, the average number of patches in a single set is over 9,300 (7.3GB), making each input roughly equivalent to processing *1% of ImageNet1k* (Deng et al., 2009). The largest input in the training set contains 32,382 patches (25.4 GB). We utilize a ResNet18 (He et al., 2016) which is pretrained on Camelyon16 (Li et al., 2021) via SimCLR (Chen et al., 2020) as a backbone feature extractor whose weights can be downloaded from this repository[1]. Our goal is to first pretrain MBC set encoders on the extracted features, and then use the unbiased estimation of the full set gradient to fine-tune the feature extractor on the full input sets. We evaluate the performance of UMBC against non-MBC MIL baselines: DS-MIL (Li et al., 2021) and AB-MIL (Ilse et al., 2018), as well as MBC baselines: DeepSets and SSE.

Results. As shown in Table 4, our UMBC model achieves the best accuracy and competitive AUROC score. Note that SSE shows the worst performance due to its constrained architecture, which even underperforms DeepSets in this task. These empirical results again verify the expressive power of our UMBC model. Moreover, we can further improve the performance of UMBC via fine-tuning the backbone network, ResNet18, which is only feasible as a consequence of our unbiased full set gradient approximation which incurs constant memory overhead. However, it is not possible for the non-MBC models to fine-tune with the ResNet18 since

[1] https://github.com/binli123/dsmil-wsi

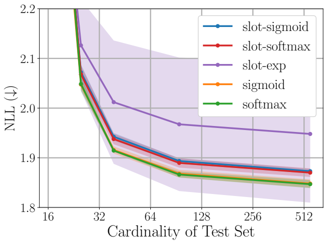


Figure 7. σ for amortized clustering

Activation	Acc.	AUC
slot-sigmoid	$83.72_{\pm 1.45}$	$0.846_{\pm 0.01}$
slot-exp	$83.26_{\pm 0.42}$	$0.850_{\pm 0.01}$
sigmoid	$81.71_{\pm 4.09}$	$0.847_{\pm 0.03}$
slot-softmax	$84.81_{\pm 0.04}$	$0.848_{\pm 0.01}$
softmax	$\mathbf{87.91}_{\pm \mathbf{1.41}}$	$\mathbf{0.874}_{\pm \mathbf{0.01}}$

Table 5. σ for MIL

it is computationally prohibitive to compute the gradient of the ResNet18 with sets consisting of tens of thousands of patches with 256×256 resolution.

4.5. Ablation Study

To validate effectiveness of activation functions σ for attention in equation 3, we train UMBC + Set Transformer with different activation functions listed in Table 15 for the amortized clustering and MIL pretraining tasks. As shown in Figure 7 and Table 5, softmax attention outperforms all the other activation functions whereas the slot-sigmoid used for attention in SSE underperforms. This experiment highlights the importance of choosing the proper activation function for attention, which is enabled by our UMBC framework.

5. Limitations and Future Work

One potential limitation of our method is a higher time complexity needed for our proposed unbiased gradient estimation during mini-batch training. If n represents the size of a single subset, and N represents the size of the whole set, then the naive sampling strategy of Bruno et al. (2021) has a time complexity of $\mathcal{O}(nk)$ while our full set gradient approximation has a complexity of $\mathcal{O}(Nk)$ during training since we must process the full set. However, we note some things we gain in exchange for this higher time complexity below.

Our unbiased gradient approximation can achieve higher performance than the biased sampling of a single subset as denoted in our experiments (specifically, see Figures 4, 5c and 6). Additionally, due to the stop gradient operation in Equation (9), UMBC achieves a constant memory overhead for any training set size. Our experiment in Figure 5b shows a constant memory overhead for SSE and DeepSets only because we apply our unbiased gradient approximation to those models in that experiment. The original models as they were presented in the original works do not have a constant memory overhead. As a result, our method can process huge sets during training, and practically any GPU size can be accommodated by adjusting the size of the gradient set. For example, the average set in the experiment on Camelyon16 contains 9329 patches (7.3 GB), while the largest input in the training set contains 32,382 patches (25.46 GB). Even though the set sizes are large, models can be trained on all inputs using a single 12GB GPU due to the constant memory overhead.

Another potential limitation is that UMBC (and SSE) use a cross attention layer with parameterized slots in order to achieve an MBC model. However, the fixed parameters, which are independent to the input set, can be seen as a type of bottleneck in the attention layer which is not present in traditional self-attention. Therefore we look forward to seeing future work which may find ways to make the slot parameters depend on the input set, which may increase overall model expressivity.

6. Conclusion

In order to overcome the limited expressive power and training scalability of existing MBC set functions, such as DeepSets and SSE, we have proposed Universal MBC set functions that allow mixing both MBC and non-MBC components to leverage a broader range of architectures which increases model expressivity while universally maintaining the MBC property. Additionally, we generalized MBC attention activation functions, showing that many functions, including the softmax, are MBC. Furthermore, for training scalability, we have proposed an unbiased approximation to the full set gradient with a constant memory overhead for processing a set of any size. Lastly, we have performed extensive experiments to verify the efficiency and efficacy of our scalable set encoding framework, and theoretically shown that UMBC is a universal approximator of continuous permutation invariant functions and converges to stationary points of the total loss with the full set.

Acknowledgements

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST)), the Engineering Research Center Program through the National Research Foundation of Korea (NRF) funded by the Korean Government MSIT (NRF-2018R1A5A1059921), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2021-0-02068, Artificial Intelligence Innovation Hub), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2022-0-00184, Development and Study of AI Technologies to Inexpensively Conform to Evolving Policy on Ethics), Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00713), and Samsung Electronics (IO201214-08145-01)

References

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Bejnordi, B. E., Veta, M., Van Diest, P. J., Van Ginneken, B., Karssemeijer, N., Litjens, G., Van Der Laak, J. A., Hermsen, M., Manson, Q. F., Balkenhol, M., et al. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *Jama*, 318(22):2199–2210, 2017.

Beltagy, I., Peters, M. E., and Cohan, A. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

Bruno, A., Willette, J., Lee, J., and Hwang, S. J. Mini-batch consistent slot set encoder for scalable set encoding. *Advances in Neural Information Processing Systems*, 34: 21365–21374, 2021.

Chalkidis, I., Fergadiotis, E., Malakasiotis, P., and Androutsopoulos, I. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 6314–6322. Association for Computational Linguistics, 2019.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pp. 1597–1607. PMLR, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pp. 4171–4186, 2019.

Fehrman, B., Gess, B., and Jentzen, A. Convergence rates for the stochastic gradient descent method for non-convex objective functions. *Journal of Machine Learning Research*, 21:136, 2020.

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. A. Conditional neural processes. In *International Conference on Machine Learning*, pp. 1704–1713. PMLR, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Ilse, M., Tomczak, J., and Welling, M. Attention-based deep multiple instance learning. In *International conference on machine learning*, pp. 2127–2136. PMLR, 2018.

Jurafsky, D. and Martin, J. H. Speech and language processing: An introduction to speech recognition, computational linguistics and natural language processing. *Upper Saddle River, NJ: Prentice Hall*, 2008.

Kawaguchi, K., Zhang, L., and Deng, Z. Understanding dynamics of nonlinear representation learning and its application. *Neural Computation*, 34(4):991–1018, 2022.

Kim, M. Differentiable expectation-maximization for set representation learning. In *International Conference on Learning Representations*, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Lee, J. D., Simchowitz, M., Jordan, M. I., and Recht, B. Gradient descent only converges to minimizers. In *Conference on learning theory*, pp. 1246–1257. PMLR, 2016.

Li, B., Li, Y., and Eliceiri, K. W. Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 14318–14328, 2021.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Mertikopoulos, P., Hallak, N., Kavis, A., and Cevher, V. On the almost sure convergence of stochastic gradient descent in non-convex problems. *Advances in Neural Information Processing Systems*, 33:1117–1128, 2020.

Mialon, G., Chen, D., d'Aspremont, A., and Mairal, J. A trainable optimal transport embedding for feature aggregation and its relationship to attention. In *International Conference on Learning Representations*, 2021.

Murphy, R. L., Srinivasan, B., Rao, V., and Ribeiro, B. Janossy pooling: Learning deep permutation-invariant functions for variable-size inputs. In *International Conference on Learning Representations*, 2019.

Pappagari, R., Zelasko, P., Villalba, J., Carmiel, Y., and Dehak, N. Hierarchical transformers for long document classification. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 838–844. IEEE, 2019.

Park, H., Vyas, Y., and Shah, K. Efficient classification of long documents using transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 702–709, 2022.

Pinkus, A. Approximation theory of the MLP model in neural networks. *Acta numerica*, 8:143–195, 1999.

Quellec, G., Cazuguel, G., Cochener, B., and Lamard, M. Multiple-instance learning for medical image and video analysis. *IEEE reviews in biomedical engineering*, 10: 213–234, 2017.

Rolnick, D. and Tegmark, M. The power of deeper networks for expressing natural functions. In *International Conference on Learning Representations*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wagstaff, E., Fuchs, F. B., Engelcke, M., Osborne, M. A., and Posner, I. Universal approximation of functions on sets. *Journal of Machine Learning Research*, 23(151): 1–56, 2022.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Zhang, Y., Hare, J., and Prügel-Bennett, A. Fspool: Learning set representations with featurewise sort pooling. In *International Conference on Learning Representations*, 2020.

A. Proofs

A.1. Proof of Theorem 3.3

Proof. Let $\mathfrak{S}_N$ be a set of all permutations on $\{1, \dots, N\}$ and let $\pi \in \mathfrak{S}_N$ be given. For a given set $X = [\mathbf{x}_1 \cdots \mathbf{x}_N]^\top \in \mathbb{R}^{N \times d_x}$, and permutation π, we can construct a permutation matrix $P \in \mathbb{R}^{N \times N}$ such that

$$PX = \begin{bmatrix} -\mathbf{x}_{\pi(1)}^\top - \\ \vdots \\ -\mathbf{x}_{\pi(N)}^\top - \end{bmatrix}.$$

Since we apply the feature extractor ϕ independently to each element of the set X,

$$\Phi(PX) = \begin{bmatrix} -\phi(\mathbf{x}_{\pi(1)})^\top - \\ \vdots \\ -\phi(\mathbf{x}_{\pi(N)})^\top - \end{bmatrix} = P\Phi(X).$$

With an elementwise, strictly positive activation function σ,

$$\begin{aligned} \sigma(\sqrt{d^{-1}} \cdot Q(\Phi(PX)W^K)^\top) &= \sigma(\sqrt{d^{-1}} \cdot Q(P\Phi(X)W^K)^\top) \\ &= \sigma(\sqrt{d^{-1}} \cdot Q(\Phi(X)W^K)^\top P^\top) \\ &= \sigma(\sqrt{d^{-1}} \cdot QK(X)^\top)P^\top \\ &= \hat{A}P^\top. \end{aligned}$$

For $p = 1$, the un-normalized attention score with the permutation π is

$$\nu_1(\hat{A}P^\top) = \begin{bmatrix} \hat{A}_{1,\pi(1)} / \sum_{i=1}^k \hat{A}_{i,\pi(1)} & \cdots & \hat{A}_{1,\pi(N)} / \sum_{i=1}^k \hat{A}_{i,\pi(N)} \\ \vdots & \ddots & \vdots \\ \hat{A}_{k,\pi(1)} / \sum_{i=1}^k \hat{A}_{i,\pi(1)} & \cdots & \hat{A}_{k,\pi(N)} / \sum_{i=1}^k \hat{A}_{i,\pi(N)} \end{bmatrix} = \nu_1(\hat{A})P^\top.$$

Since ν_2 is the identity mapping, $\nu_p(\hat{A}P^\top) = \nu_p(\hat{A})P^\top$ for $p = 1, 2$.

Now, we consider the matrix multiplication of

$$\nu_p(\hat{A})P^\top = \begin{bmatrix} \nu_p(\hat{A})_{1,\pi(1)} & \cdots & \nu_p(\hat{A})_{1,\pi(N)} \\ \vdots & \ddots & \vdots \\ \nu_p(\hat{A})_{k,\pi(1)} & \cdots & \nu_p(\hat{A})_{k,\pi(N)} \end{bmatrix} \text{ and } P\Phi(X)W^V = \begin{bmatrix} \phi(\mathbf{x}_{\pi(1)})^\top W^V \\ \vdots \\ \phi(\mathbf{x}_{\pi(N)})^\top W^V \end{bmatrix}.$$

Since

$$\begin{aligned} \begin{bmatrix} \nu_p(\hat{A})_{1,\pi(1)} & \cdots & \nu_p(\hat{A})_{1,\pi(N)} \\ \vdots & \ddots & \vdots \\ \nu_p(\hat{A})_{k,\pi(1)} & \cdots & \nu_p(\hat{A})_{k,\pi(N)} \end{bmatrix} \begin{bmatrix} \phi(\mathbf{x}_{\pi(1)})^\top W^V \\ \vdots \\ \phi(\mathbf{x}_{\pi(N)})^\top W^V \end{bmatrix} &= \begin{bmatrix} \sum_{j=1}^N \nu_p(\hat{A})_{1,\pi(j)} \phi(\mathbf{x}_{\pi(j)})^\top W^V \\ \vdots \\ \sum_{j=1}^N \nu_p(\hat{A})_{k,\pi(j)} \phi(\mathbf{x}_{\pi(j)})^\top W^V \end{bmatrix} \\ &= \begin{bmatrix} \sum_{j=1}^N \nu_p(\hat{A})_{1,j} \phi(\mathbf{x}_j)^\top W^V \\ \vdots \\ \sum_{j=1}^N \nu_p(\hat{A})_{k,j} \phi(\mathbf{x}_j^\top) W^V \end{bmatrix} \\ &= \nu_p(\hat{A})V(X), \end{aligned}$$

$\hat{f}_\theta$ is permutation invariant. Since $\bar{f}_\theta(X)_i = \sum_{j=1}^N \nu_p(\hat{A})_{i,j} = \sum_{j=1}^N \nu_p(\hat{A})_{i,\pi(j)}$, diag $\left(\bar{f}_\theta(X)\right)^{-1}$ is also invariant with respect to the permutation of input X, which leads to the conclusion that $f_\theta(PX) = \text{diag}\left(\bar{f}_\theta(PX)\right)^{-1} \hat{f}_\theta(PX) = \text{diag}(\bar{f}_\theta(X))^{-1}\hat{f}_\theta(X)$.
$\therefore$ f_θ is permutation invariant. □

A.2. Proof for Theorem 3.4

Proof. Let input set $X \in \mathbb{R}^{N \times d_x}$ be given and let $\zeta(X) = \{S_1, \ldots, S_l\}$ be a partition of X with $|S_i| = N_i$, *i.e.*, $X = \bigcup_{i=1}^{l} S_i$ and $S_i \cap S_j = \emptyset$ for $i \neq j$. Since a Universal MBC set encoder is permutation invariant, without loss of generality we can assume that,

$$K(X) = \begin{bmatrix} K(X)_1 \\ \vdots \\ K(X)_l \end{bmatrix}, \quad V(X) = \begin{bmatrix} V(X)_1 \\ \vdots \\ V(X)_l \end{bmatrix} \tag{21}$$

where $K(X)_i = \Phi(S_i)W^K \in \mathbb{R}^{N_i \times d}$ and $V(X)_i = \Phi(S_i)W^V \in \mathbb{R}^{N_i \times d}$ for $i = 1 \ldots, l$. Then we can express the matrix $\nu_p(\hat{A})$ as follows:

$$\nu_p(\hat{A}) = \left[\nu_p(\hat{A}^{(1)}) \cdots \nu_p(\hat{A}^{(l)})\right], \tag{22}$$

where $\hat{A}^{(i)} = \sigma(\sqrt{d^{-1}} \cdot QK(X)_i^\top) \in \mathbb{R}^{k \times N_i}$ for $i = 1 \ldots, l$ since $\nu_p(\hat{A})_{i,j}$ is independent to $\nu_p(\hat{A})_{i,q}$ for all $q \neq j$.

Since

$$\hat{f}_\theta(X) = \left[\nu_p(\hat{A}^{(1)}) \cdots \nu_p(\hat{A}^{(l)})\right] \begin{bmatrix} V(X)_1 \\ \vdots \\ V(X)_l \end{bmatrix},$$

the following equality holds

$$\begin{aligned} \hat{f}_\theta(X) &= \left[\nu_p(\hat{A}^{(1)}) \cdots \nu_p(\hat{A}^{(l)})\right] \begin{bmatrix} V(X)_1 \\ \vdots \\ V(X)_l \end{bmatrix} \\ &= \sum_{i=1}^{l} \nu_p(\hat{A}^{(i)}) V(X)_i \\ &= \sum_{i=1}^{l} \hat{f}_\theta(S_i). \end{aligned} \tag{23}$$

Thus, $\hat{f}_\theta(X)$ is mini-batch consistent.

Since

$$\bar{f}_\theta(X)_i = \sum_{q=1}^{l} \sum_{j=1}^{N_i} \nu_p(\hat{A}^{(q)})_{i,j},$$

we can decompose $\bar{f}_\theta(X)$ into a summation of $\bar{f}_\theta(S_i)$ as

$$\begin{aligned} \bar{f}_\theta(X) &= \sum_{i=1}^{l} \left(\sum_{j=1}^{N_i} \nu_p(\hat{A}^{(i)})_{1,j}, \ldots, \sum_{j=1}^{N_i} \nu_p(\hat{A}^{(i)})_{k,j} \right)^\top \\ &= \sum_{i=1}^{l} \nu_p(\hat{A}^{(i)}) \mathbf{1}_{N_i} \\ &= \sum_{i=1}^{l} \bar{f}_\theta(S_i), \end{aligned} \tag{24}$$

where $\mathbf{1}_{N_i} = (1, \ldots, 1) \in \mathbb{R}^{N_i}$. It implies that $\bar{f}_\theta(X)$ is mini-batch consistent (MBC). Combining equation 23 and equation 24

$$f_\theta(X) = \operatorname{diag}\left(\sum_{i=1}^{l} \bar{f}_\theta(S_i)\right)^{-1} \left(\sum_{i=1}^{l} \hat{f}_\theta(S_i)\right).$$

Now, we define a function,

$$h(\{f_\theta(S_1), \ldots, f_\theta(S_l)\}) \coloneqq h_1(\{\bar{f}_\theta(S_1), \ldots, \bar{f}_\theta(S_l)\}) \cdot h_2(\{\hat{f}_\theta(S_1), \ldots, \hat{f}_\theta(S_l)\}),$$

where

$$h_1(\{\bar{f}_\theta(S_1), \ldots \bar{f}_\theta(S_l)\} \coloneqq \operatorname{diag}\left(\sum_{i=1}^{l} \bar{f}_\theta(S_i)\right)^{-1}$$

$$h_2(\{\hat{f}_\theta(S_1), \ldots, \hat{f}_\theta(S_l)\}) \coloneqq \sum_{i=1}^{l} \hat{f}_\theta(S_i).$$

Then $f_\theta(X) = h(\{f_\theta(S_1), \ldots, f_\theta(S_l)\})$. Since $\zeta(X)$ is arbitrary, f_θ is mini-batch consistent. □

A.3. Proof of Corollary 3.8

Proof. Let $\hat{g}_\omega : \mathbb{R}^{k\times d} \to \mathcal{Z}$ be an arbitrary set encoder and let $f_\theta : \mathcal{X} \to \mathbb{R}^{k\times d}$ be a UMBC set encoder. Given a set $X \in \mathcal{X}$ and a partition $\zeta(X)$, we get

$$f_\theta(X) = \operatorname{diag}\left(\sum_{S\in\zeta(X)} \bar{f}_\theta(S)\right)^{-1}\left(\sum_{S\in\zeta(X)} \hat{f}_\theta(S)\right) \in \mathbb{R}^{k\times d},$$

as shown in section A.2. We assume that k is small enough so that we can load $f_\theta(X)$ in memory after we compute $f_\theta(X)$. As a consequence, we can directly evaluate $\hat{g}_\omega(f_\theta(X))$ without partitioning $f_\theta(X)$ into smaller subsets $\{S_1, \ldots, S_l\}$ and aggregating $\hat{g}_\omega(S_i)$.
$\therefore \hat{g}_\omega \circ f_\theta$ is mini-batch consistent. □

A.4. Proof of Theorem 3.6

Proof. Let $\pi \in \mathfrak{S}_k$ be a permutation on $\{1 \ldots, k\}$ and let $\mathfrak{S}_k$ be a set of all permutations on $\{1, \ldots, k\}$. Define a matrix

$$Z = \begin{bmatrix} -\mathbf{z}_1^\top- \\ \vdots \\ -\mathbf{z}_k^\top- \end{bmatrix} \coloneqq \varphi(\Sigma; X, \theta) \in \mathbb{R}^{k\times d}$$

with the input set $X \in \mathbb{R}^{N\times d_x}$ and the given slots $\Sigma = [\mathbf{s}_1 \cdots \mathbf{s}_k]^\top \in \mathbb{R}^{k\times d_s}$. Then we can identify a permutation matrix $P \in \mathbb{R}^{k\times k}$ such that,

$$P\Sigma = \begin{bmatrix} -\mathbf{s}_{\pi(1)}^\top- \\ \vdots \\ -\mathbf{s}_{\pi(k)}^\top- \end{bmatrix}.$$

Since the query Q with the permutation P is $P\Sigma W^Q = PQ$, the un-normalized attention score with the permutation matrix P is

$$\sigma(\sqrt{d^{-1}} \cdot\ PQK(X)^\top) = P\sigma(\sqrt{d^{-1}} \cdot\ QK(X)^\top) = P\hat{A}.$$

Since the normalization matrix $\bar{f}_\theta(X)$ is a function of the slots Σ, we define a new normalization matrix by permuting the slots Σ with the given permutation matrix P as

$$\operatorname{diag}\left(\bar{f}_\theta(X); P\Sigma\right)^{-1} = \begin{bmatrix} \frac{1}{c_1} & & & \\ & \frac{1}{c_2} & & \\ & & \ddots & \\ & & & \frac{1}{c_k} \end{bmatrix}$$

where $c_i = \sum_{j=1}^{N} \nu_p(P\hat{A})_{i,j}$ for $i = 1, \ldots, k$. Note that

$$\begin{aligned}
\nu_1(P\hat{A}) &= \begin{bmatrix} \hat{A}_{\pi(1),1}/\sum_{i=1}^k \hat{A}_{\pi(i),1} & \cdots & \hat{A}_{\pi(1),N}/\sum_{i=1}^k \hat{A}_{\pi(i),N} \\ \vdots & \ddots & \vdots \\ \hat{A}_{\pi(k),1}/\sum_{i=1}^k \hat{A}_{\pi(i),1} & \cdots & \hat{A}_{\pi(k),N}/\sum_{i=1}^k \hat{A}_{\pi(i),N} \end{bmatrix} \\
&= P\hat{A} \cdot \mathrm{diag}\left(\sum_{i=1}^k \hat{A}_{i,1}, \ldots, \sum_{i=1}^k \hat{A}_{i,N}\right)^{-1} \\
&= P\nu_1(\hat{A})
\end{aligned}$$

and $\nu_2(P\hat{A}) = P\nu_2(\hat{A})$.

Then we get $c_i = \bar{f}_\theta(X)_{\pi(i)}$ since

$$\begin{aligned}
c_i &= \sum_{j=1}^N \nu_p(P\hat{A})_{i,j} \\
&= \sum_{j=1}^N (P\nu_p(\hat{A}))_{i,j} \\
&= \sum_{j=1}^N \sum_{l=1}^k P_{i,l}\nu_p(\hat{A})_{l,j} \\
&= \sum_{l=1}^k P_{i,l} \left(\sum_{j=1}^N \nu_p(\hat{A})_{l,j}\right) \\
&= \sum_{l=1}^k P_{i,l}\bar{f}_\theta(X)_l \\
&= \bar{f}_\theta(X)_{\pi(i)}.
\end{aligned}$$

The last equality holds since i-th row of the permutation matrix P has a single non-zero entry which is 1. Thus,

$$\begin{bmatrix} c_1 & & & \\ & c_2 & & \\ & & \ddots & \\ & & & c_k \end{bmatrix} = \begin{bmatrix} \bar{f}_\theta(X)_{\pi(1)} & & & \\ & \bar{f}_\theta(X)_{\pi(2)} & & \\ & & \ddots & \\ & & & \bar{f}_\theta(X)_{\pi(k)} \end{bmatrix} = P\mathrm{diag}(\bar{f}_\theta(X)),$$

which implies that

$$\begin{aligned}
\mathrm{diag}\left(\bar{f}_\theta(X); P\Sigma\right)^{-1} &= \begin{bmatrix} \frac{1}{c_1} & & & \\ & \frac{1}{c_2} & & \\ & & \ddots & \\ & & & \frac{1}{c_k} \end{bmatrix} \\
&= \begin{bmatrix} \bar{f}_\theta(X)_{\pi(1)} & & & \\ & \bar{f}_\theta(X)_{\pi(2)} & & \\ & & \ddots & \\ & & & \bar{f}_\theta(X)_{\pi(k)} \end{bmatrix}^{-1}.
\end{aligned}$$

Finally, combining all the pieces, we get

$$
\begin{aligned}
&\text{diag}\left(\bar{f}_\theta(X); P\Sigma\right)^{-1} \nu_p(P\hat{A})V(X) \\
&= \text{diag}\left(\bar{f}_\theta(X); P\Sigma\right)^{-1} P\nu_p(\hat{A})V(X) \\
&= \begin{bmatrix} \bar{f}_\theta(X)_{\pi(1)} & & & \\ & \bar{f}_\theta(X)_{\pi(2)} & & \\ & & \ddots & \\ & & & \bar{f}_\theta(X)_{\pi(k)} \end{bmatrix}^{-1} \begin{bmatrix} (\nu_p(\hat{A})V(X))_{\pi(1),1} & \cdots & (\nu_p(\hat{A})V(X))_{\pi(1),d} \\ \vdots & \ddots & \vdots \\ (\nu_p(\hat{A})V(X))_{\pi(k),1} & \cdots & (\nu_p(\hat{A})V(X))_{\pi(k),d} \end{bmatrix} \\
&= \begin{bmatrix} (\nu_p(\hat{A})V(X))_{\pi(1),1}/\bar{f}_\theta(X)_{\pi(1)} & \cdots & (\nu_p(\hat{A})V(X))_{\pi(1),d}/\bar{f}_\theta(X)_{\pi(1)} \\ \vdots & \ddots & \vdots \\ (\nu_p(\hat{A})V(X))_{\pi(k),1}/\bar{f}_\theta(X)_{\pi(k)} & \cdots & (\nu_p(\hat{A})V(X))_{\pi(k),d}/\bar{f}_\theta(X)_{\pi(k)} \end{bmatrix} \\
&= P \begin{bmatrix} \bar{f}_\theta(X)_1 & & & \\ & \bar{f}_\theta(X)_2 & & \\ & & \ddots & \\ & & & \bar{f}_\theta(X)_k \end{bmatrix}^{-1} \nu_p(\hat{A})V(X) \\
&= P\text{diag}\left(\bar{f}_\theta(X)\right)^{-1} \hat{f}_\theta(X) \\
&= P\varphi(\Sigma; X, \theta) \\
&= PZ \\
&= \begin{bmatrix} -\mathbf{z}_{\pi(1)}^\top- \\ \vdots \\ -\mathbf{z}_{\pi(k)}^\top- \end{bmatrix}.
\end{aligned}
$$

$\therefore \varphi(\Sigma; X, \theta)$ is permutation equivariant. □

A.5. Proof of Theorem 3.7

Proof. Following the previous proofs (Zaheer et al., 2017; Wagstaff et al., 2022) for *uncountable set* $\mathfrak{X}$, we assume a set size is fixed. In other words, we restrict the domain $\mathcal{X} \subset 2^{\mathfrak{X}}$ to $[0,1]^M$. Let $f \in \mathcal{F}$ be given. By using the proof of Theorem 13 from Wagstaff et al. (2022) (with a more detailed proof in Zaheer et al., 2017), the function f is continuously sum-decomposable via $\mathbb{R}^{M+1}$ as:

$$f(X) = \rho\left(\Psi(X)\right)$$

for all $X = \{x_1, \ldots, x_M\} \in [0,1]^M$, where $\Psi : [0,1]^M \to \mathcal{Q} \subseteq \mathbb{R}^{M+1}$ is invertible and defined by

$$\Psi(X) = \left(\sum_{x\in X} \psi_0(x), \ldots, \sum_{x\in X} \psi_M(x)\right), \quad \psi_q(x) = x^q, \quad \text{for } q = 0, \ldots, M,$$

and $\rho : \mathbb{R}^{M+1} \to \mathbb{R}$ is continuous and defined by

$$\rho = f \circ \Psi^{-1}.$$

We want to show that UMBC with some continuously decomposable permutation invariant deep neural network can approximate the function f by showing that ρ and Ψ are approximated by components of the UMBC model. Let $h : [0,1]^M \to \mathbb{R}$ be a continuously decomposable permutation invariant deep neural network defined by

$$h(Z) = \kappa\left(\sum_{\mathbf{z}\in Z} \xi(\mathbf{z})\right),$$

where $Z = \{\mathbf{z}_i \in \mathbb{R}^{M+1}\}_{i=1}^k$, $\kappa : \mathbb{R}^{M+1} \to \mathbb{R}$ is a deep neural network, and $\xi : \mathbb{R}^{M+1} \to \mathbb{R}^{M+1}$ is defined by

$$\xi(\mathbf{z}) = \frac{1}{k} \cdot \hat{\xi}(M \cdot \mathbf{z})$$

with some deep neural network $\hat{\xi} : \mathbb{R}^{M+1} \to \mathbb{R}^{M+1}$. First, we want to show that Deepsets with average pooling is a special case of UMBC. Set the slots Σ as the zero matrix $\mathbf{0} \in \mathbb{R}^{k \times d_s}$, $d = d_h = M+1$, and $W^V = I_{M+1}$. Then by using Lemma 1 from Lee et al. (2019), f_θ becomes average pooling, *i.e.*,

$$f_\theta(X) = \begin{bmatrix} \frac{1}{M}\sum_{i=1}^{M}\phi(x_i) \\ \vdots \\ \frac{1}{M}\sum_{i=1}^{M}\phi(x_i) \end{bmatrix} \in \mathbb{R}^{k\times(M+1)}.$$

Then the composition of UMBC and h becomes continuously sum-decomposable function as follows:

$$\begin{aligned}(h \circ f_\theta)(X) &= \kappa\left(\sum_{j=1}^{k}\xi(f_\theta(X)_j)\right) \\ &= \kappa\left(\sum_{j=1}^{k}\frac{1}{k}\hat{\xi}\left(\frac{M}{M}\sum_{i=1}^{M}\phi(x_i)\right)\right) \\ &= (\kappa \circ \hat{\xi})\left(\sum_{i=1}^{M}\phi(x_i)\right)\end{aligned}$$

where $f_\theta(X)_j$ is j-th row of $f_\theta(X)$. By defining $\hat{\rho} \coloneqq \kappa \circ \hat{\xi}$ and $\hat{\Psi}(X) \coloneqq \sum_{x\in X}\phi(x) \in \hat{\mathcal{Q}} \subseteq \mathbb{R}^{M+1}$,

$$\begin{aligned}\sup_X \|f(X) - (h\circ f_\theta)(X)\|_2 &= \sup_X \left\|(\rho\circ\Psi)(X) - (\hat{\rho}\circ\hat{\Psi})(X) + (\rho\circ\hat{\Psi})(X) - (\rho\circ\hat{\Psi})(X)\right\|_2 \\ &\leq \sup_X \left\|(\rho\circ\Psi)(X) - (\rho\circ\hat{\Psi})(X)\right\|_2 + \sup_X \left\|(\hat{\rho}\circ\hat{\Psi})(X) - (\rho\circ\hat{\Psi})(X)\right\|_2 \\ &\leq \sup_X \left\|\rho(\hat{\Psi}(X)) - \rho(\Psi(X))\right\|_2 + \sup_{z\in\hat{\mathcal{Q}}}\|\rho(z) - \hat{\rho}(z)\|_2.\end{aligned}$$

Since $\hat{\Psi} : [0,1]^M \to \hat{\mathcal{Q}} \subseteq \mathbb{R}^{M+1}$ is continuous and $[0,1]^M$ is compact, $\hat{\mathcal{Q}}$ is compact. Since $\hat{\mathcal{Q}}$ is compact and ρ is continuous, and the nonlinearity of $\hat{\rho}$ is not a polynomial of finite degree, Theorem 3.1 of (Pinkus, 1999) implies the following (as a network with one hidden layer can be approximated by a network of greater depth by using the same construction for the first layer and approximating the identity function with later layers): for any $\epsilon' > 0$, there exists $\tau' \geq 1$ and parameters of $\hat{\rho}$ such that if the width of $\hat{\rho}$ is at least τ', then $\sup_{z\in\hat{\mathcal{Q}}}\|\rho(z) - \hat{\rho}(z)\| < \epsilon'$. Combining these, we have that

$$\sup_X \|f(X) - (h\circ f_\theta)(X)\|_2 < \sup_X \left\|\rho(\hat{\Psi}(X)) - \rho(\Psi(X))\right\|_2 + \epsilon'_{(\tau')}$$

where $\epsilon'_{(\tau')}$ depends on the width τ' of $\hat{\rho}$. Since the nonlinearity of ϕ has nonzero Taylor coefficients up to degree M, the proof of Theorem 3.4 of (Rolnick & Tegmark, 2018) implies the following: there exists $\tau \geq 1$ such that if the width of $\phi = (\phi_0, \phi_1, \ldots, \phi_M)$ is at least τ, for every $\delta > 0$, there exists parameters of ϕ such that $\sup_x|\phi_q(x) - \psi_q(x)| < \frac{\delta}{2M(M+1)}$

for $q \in \{0, 1, \ldots, M\}$. Let us fix the width of ϕ to be at least τ. By the triangle inequality,

$$\begin{aligned}\left\|\hat{\Psi}(X) - \Psi(X)\right\|_2 &= \left\|\sum_{x \in X} (\phi_0(x) - \psi_0(x), \ldots, \phi_M(x) - \psi_M(x))\right\|_2 \\ &\leq \sum_{x \in X} \|(\phi_0(x) - \psi_0(x), \ldots, \phi_M(x) - \psi_M(x))\|_2 \\ &\leq \sum_{x \in X} \sum_{q=0}^{M} |\phi_q(x) - \psi_q(x)| \\ &\leq |X| \sum_{q=0}^{M} \sup_x |\phi_q(x) - \psi_q(x)| \\ &< M(M+1) \frac{\delta}{2M(M+1)} \\ &= \frac{\delta}{2}\end{aligned}$$

for all $X \in [0,1]^M$. It implies that for every $\delta > 0$ there exists the parameters of ϕ such that

$$\sup_X \left\|\hat{\Psi}(X) - \Psi(X)\right\|_2 \leq \frac{\delta}{2} < \delta.$$

Since $\Psi : [0,1]^M \to \mathcal{Q} \subseteq \mathbb{R}^{M+1}$ is continuous and $[0,1]^M$ is compact, $\mathcal{Q}$ is compact. Since $\mathcal{Q}$ and $\hat{\mathcal{Q}}$ are compact, $\tilde{\mathcal{Q}} := \mathcal{Q} \cup \hat{\mathcal{Q}}$ is compact. Define $\tilde{\rho} : \tilde{\mathcal{Q}} \to \mathbb{R}$ by $\tilde{\rho}(z) = \rho(z)$ for all $z \in \tilde{\mathcal{Q}}$. Replacing ρ with $\tilde{\rho}$,

$$\sup_X \|f(X) - (h \circ f_\theta)(X)\| < \sup_X \|\tilde{\rho}(\hat{\Psi}(X)) - \tilde{\rho}(\Psi(X))\| + \epsilon'_{(\tau')}.$$

Since $\tilde{\mathcal{Q}}$ is compact and $\tilde{\rho}$ is continuous on $\tilde{\mathcal{Q}}$, $\tilde{\rho}$ is uniformly continuous. Thus, for any $\epsilon > 0$ there is a δ_0 such that

$$\text{for any } z_1, z_2 \in \tilde{\mathcal{Q}} \text{ with } \|z_1 - z_2\|_2 < \delta_0 \Rightarrow \|\tilde{\rho}(z_1) - \tilde{\rho}(z_2)\|_2 < \epsilon.$$

Since $\sup_X \|\hat{\Psi}(X) - \Psi(X)\|_2 < \delta$ with an arbitrary small $\delta > 0$, we can take a small δ such that $\delta < \delta_0$, *i.e.* $\|\hat{\Psi}(X) - \Psi(X)\|_2 < \delta_0$ for all $X \in [0,1]^M$. Then for all $X \in [0,1]^M$,

$$\left\|\tilde{\rho}(\hat{\Psi}(X)) - \tilde{\rho}(\Psi(X))\right\|_2 < \epsilon.$$

It implies that

$$\left\|\tilde{\rho}(\hat{\Psi}(X)) - \tilde{\rho}(\Psi(X))\right\|_2 \leq \sup_X \left\|\tilde{\rho}(\hat{\Psi}(X)) - \tilde{\rho}(\Psi(X))\right\|_2 \leq \epsilon.$$

Thus, we get

$$\sup_X \|f(X) - (h \circ f_\theta)(X)\|_2 \leq \epsilon + \epsilon'_{(\tau')},$$

where $\epsilon'_{(\tau')}$ depends on the width τ' of $\hat{\rho}$, while $\epsilon > 0$ is arbitrarily small with a fixed width of ϕ (due to the universal approximation result with a bounded width of Rolnick & Tegmark, 2018). Let $\epsilon_0 > 0$ be given. Then, we set τ' to be sufficiently large to ensure that $\epsilon'_{(\tau')} < \epsilon_0/2$ and set $\epsilon < \epsilon_0/2$, obtaining

$$\sup_X \|f(X) - (h \circ f_\theta)(X)\|_2 < \epsilon_0.$$

Since $\epsilon_0 > 0$ was arbitrary, this proves the following desired result: (a formal restatement of this theorem) suppose that the nonlinear activation function of ϕ has nonzero Taylor coefficients up to degree M. Let h be a continuously-decomposable permutation-invariant deep neural network with the nonlinear activation functions that are not polynomials of finite degrees. Then, there exists $\tau \geq 1$ such that if the width of ϕ is at least τ, then for any $\epsilon_0 > 0$, there exists $\tau' \geq 1$ for which the following statement holds: if the width of h is at least τ', then there exist trainable parameters of f_θ and h satisfying

$$\sup_X \|f(X) - (h \circ f_\theta)(X)\|_2 < \epsilon_0.$$

□

A.6. Proof for Theorem 3.10

$f_\theta(X) \in \mathbb{R}^{k\times d}$ is defined by

$$f_\theta(X)_i = \frac{1}{\bar f_\theta(X)_i}\hat f_\theta(X)_i \in \mathbb{R}^d$$

for all $i = 1, \dots, k$, where $\hat f_\theta(X)_i \in \mathbb{R}^d$ is i-th row of $\hat f_\theta(X)$ defined in equation 4 and $\bar f_\theta(X)_i \in \mathbb{R}$ is i-th component of $\bar f_\theta(X)$ which is defined in equation 5.

Proof. From the mini-batch consistency and definition of $\hat f_\theta$ and $\bar f_\theta$, we have that for any partition procedure ζ,

$$\hat f_\theta(X) = \sum_{S\in\zeta(X)} \hat f_\theta(S) \text{ and } \bar f_\theta(X) = \sum_{S\in\zeta(X)} \bar f_\theta(S).$$

By using this and defining $\ell_i(q) := \ell(q, y_i) \in \mathbb{R}$ and $\mathbf{z}_j^{(i)} := f_\theta(X_i)_j \in \mathbb{R}^d$, where $f_\theta(X_i)_j$ is j-th row of $f_\theta(X_i)$, the chain rule along with the linearity of the derivative operator yields that for any partition procedure ζ,

$$\begin{aligned}
\frac{\partial L(\theta,\lambda)}{\partial\theta} &= \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{\partial(\ell_i\circ g_\lambda)(\mathbf{z}_j^{(i)})}{\partial\mathbf{z}_j^{(i)}}\frac{\partial f_\theta(X_i)_j}{\partial\theta} \\
&= \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{\partial(\ell_i\circ g_\lambda)(\mathbf{z}_j^{(i)})}{\partial\mathbf{z}_j^{(i)}}\left(\frac{1}{\bar f_\theta(X_i)_j}\frac{\partial\hat f_\theta(X_i)_j}{\partial\theta} - \hat f_\theta(X)_j\frac{1}{\bar f_\theta(X_i)_j^2}\frac{\partial\bar f_\theta(X_i)_j}{\partial\theta}\right) \qquad (25) \\
&= \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{k}\frac{\partial(\ell_i\circ g_\lambda)(\mathbf{z}_j^{(i)})}{\partial\mathbf{z}_j^{(i)}}\left(\frac{1}{\bar f_\theta(X_i)_j}\sum_{S\in\zeta(X_i)}\frac{\partial\hat f_\theta(S)_j}{\partial\theta} - \frac{1}{\bar f_\theta(X_i)_j^2}\hat f_\theta(X_i)_j\sum_{S\in\zeta(X_i)}\frac{\partial\bar f_\theta(S)_j}{\partial\theta}\right).
\end{aligned}$$

Similarly, by defining $\bar\ell_i(q) := \ell(q, \bar y_i) \in \mathbb{R}$ and $\bar{\mathbf{z}}_j^{(i)} := \hat f_\theta(\bar X_i)_j \in \mathbb{R}^d$,

$$\begin{aligned}
\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta} &= \frac{1}{m}\sum_{i=1}^{m}\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{j=1}^{k}\frac{\partial(\bar\ell_i\circ g_\lambda)(\bar{\mathbf{z}}_j^{(i)})}{\partial\bar{\mathbf{z}}_j^{(i)}}\frac{\partial f_\theta^{\bar\zeta_t}(\bar X_i)_j}{\partial\theta} \qquad (26) \\
&= \frac{1}{m}\sum_{i=1}^{m}\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{j=1}^{k}\frac{\partial(\bar\ell_i\circ g_\lambda)(\bar{\mathbf{z}}_j^{(i)})}{\partial\bar{\mathbf{z}}_j^{(i)}}\left(\frac{1}{\bar f_\theta(\bar X_i)_j}\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\hat f_\theta(\bar S)_j}{\partial\theta} - \frac{1}{\bar f_\theta(\bar X_i)_j^2}\hat f_\theta(\bar X_i)_j\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\bar f_\theta(\bar S)_j}{\partial\theta}\right).
\end{aligned}$$

Let $t \in \mathbb{N}_+$ be fixed. By the linearity of expectation, we have that

$$\mathbb{E}_{((\bar X_i,\bar y_i))_{i=1}^m}\mathbb{E}_{(\bar\zeta_t(\bar X_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right] = \mathbb{E}_{((\bar X_i,\bar y_i))_{i=1}^m}\left[\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{k}\frac{\partial(\bar\ell_i\circ g_\lambda)(\bar{\mathbf{z}}_j^{(i)})}{\partial\bar{\mathbf{z}}_j^{(i)}}\mathrm{Q}_{\mathrm{ij}}\right] \qquad (27)$$

where

$$\mathrm{Q}_{ij} = \frac{1}{\bar f_\theta(\bar X_i)_j}\mathbb{E}_{\bar\zeta_t(\bar X_i)}\left[\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\hat f_\theta(\bar S)_j}{\partial\theta}\right] - \frac{1}{\bar f_\theta(\bar X_i)_j^2}\hat f_\theta(\bar X_i)_j\mathbb{E}_{\bar\zeta_t(\bar X_i)}\left[\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\bar f_\theta(\bar S)_j}{\partial\theta}\right].$$

Below, we further analyze the following factors in the right-hand side of this equation:

$$\mathbb{E}_{\bar\zeta_t(\bar X_i)}\left[\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\hat f_\theta(\bar S)_j}{\partial\theta}\right] \text{ and } \mathbb{E}_{\bar\zeta_t(\bar X_i)}\left[\frac{|\zeta_t(\bar X_i)|}{|\bar\zeta_t(\bar X_i)|}\sum_{\bar S\in\bar\zeta_t(\bar X_i)}\frac{\partial\bar f_\theta(\bar S)_j}{\partial\theta}\right].$$

Denote the elements of $\bar{\zeta}_t(\bar{X}_i)$ as $\{\bar{S}_1, \bar{S}_2, \dots, \bar{S}_{|\bar{\zeta}_t(\bar{X}_i)|}\} = \bar{\zeta}_t(\bar{X}_i)$. Then,

$$\begin{aligned}\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{\bar{S}\in\bar{\zeta}_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(\bar{S})_j}{\partial\theta}\right] &= \mathbb{E}_{\bar{S}_1,\bar{S}_2,\dots,\bar{S}_{|\bar{\zeta}_t(\bar{X}_i)|}}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{l=1}^{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\hat{f}_\theta(\bar{S}_l)_j}{\partial\theta}\right] \\ &= \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{l=1}^{|\bar{\zeta}_t(\bar{X}_i)|}\mathbb{E}_{\bar{S}_l}\left[\frac{\partial\hat{f}_\theta(\bar{S}_l)_j}{\partial\theta}\right].\end{aligned}$$

Since $\bar{S}_l$ is drawn independently and uniformly from the elements of $\zeta_t(\bar{X}_i)$, we have that

$$\mathbb{E}_{\bar{S}_l}\left[\frac{\partial\hat{f}_\theta(\bar{S}_l)_j}{\partial\theta}\right] = \frac{1}{|\zeta_t(\bar{X}_i)|}\sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta}.$$

Substituting this into the right-hand side of the preceding equation, we have that

$$\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{\bar{S}\in\bar{\zeta}_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(\bar{S})_j}{\partial\theta}\right] = \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{l=1}^{|\bar{\zeta}_t(\bar{X}_i)|}\frac{1}{|\zeta_t(\bar{X}_i)|}\sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta} = \sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta}.$$

Similarly,

$$\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{\bar{S}\in\bar{\zeta}_t(\bar{X}_i)}\frac{\partial\bar{f}_\theta(\bar{S})_j}{\partial\theta}\right] = \sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta}.$$

Substituting these into Q_{ij},

$$\mathrm{Q}_{ij} = \frac{1}{\bar{f}_\theta(\bar{X}_i)_j}\sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta} - \frac{1}{\bar{f}_\theta(\bar{X}_i)_j^2}\hat{f}_\theta(X)_j\sum_{S\in\zeta_t(\bar{X}_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta}.$$

By using this in equation 27 and defining $B(\bar{X}_i, \bar{y}_i)_j = \frac{\partial(\bar{\ell}_i\circ g)(\bar{\mathbf{z}}_j^{(i)})}{\partial\bar{\mathbf{z}}_j^{(i)}}\mathrm{Q}_{ij}$, we have that

$$\begin{aligned}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right] = \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\left[\frac{1}{m}\sum_{i=1}^m\sum_{j=1}^k B(\bar{X}_i,\bar{y}_i)_j\right] &= \frac{1}{m}\sum_{i=1}^m\sum_{j=1}^k\mathbb{E}_{(\bar{X}_i,\bar{y}_i)}[B(\bar{X}_i,\bar{y}_i)_j] \\ &= \sum_{j=1}^k\frac{1}{m}\sum_{i=1}^m\frac{1}{n}\sum_{l=1}^n B(X_l,y_l)_j \\ &= \frac{1}{n}\sum_{i=1}^n\sum_{j=1}^k B(X_i,y_i)_j.\end{aligned}$$

Thus, expanding the definition of $B(X_i, y_i)_j$,

$$\begin{aligned}&\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right] \\ &= \frac{1}{n}\sum_{i=1}^n\sum_{j=1}^k\frac{\partial(\ell_i\circ g_\lambda)(\mathbf{z}_j^{(i)})}{\partial\mathbf{z}_j^{(i)}}\left(\frac{1}{\bar{f}_\theta(X_i)_j}\sum_{S\in\zeta_t(X_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta} - \frac{1}{\bar{f}_\theta(X_i)_j^2}\hat{f}_\theta(X)_j\sum_{S\in\zeta_t(X_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta}\right).\end{aligned}$$

Here, since $\sum_{S\in\zeta_t(X_i)}\frac{\partial\hat{f}_\theta(S)}{\partial\theta} = \sum_{S\in\zeta(X_i)}\frac{\partial\hat{f}_\theta(S)}{\partial\theta}$ and $\sum_{S\in\zeta_t(X_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta} = \sum_{S\in\zeta(X_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta}$ for any partition procedure ζ from the mini-batch consistency, we have that

$$\begin{aligned}&\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right] \\ &= \frac{1}{n}\sum_{i=1}^n\sum_{j=1}^k\frac{\partial(\ell_i\circ g_\lambda)(\mathbf{z}_j^{(i)})}{\partial\mathbf{z}_j^{(i)}}\left(\frac{1}{\bar{f}_\theta(X_i)_j}\sum_{S\in\zeta(X_i)}\frac{\partial\hat{f}_\theta(S)_j}{\partial\theta} - \frac{1}{\bar{f}_\theta(X_i)_j^2}\hat{f}_\theta(X)_j\sum_{S\in\zeta(X_i)}\frac{\partial\bar{f}_\theta(S)_j}{\partial\theta}\right).\end{aligned} \tag{28}$$

By comparing equation equation 25 and equation equation 28, we conclude that

$$\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta}\right]=\frac{\partial L(\theta,\lambda)}{\partial\theta}.$$

Since t was arbitrary, this holds for any $t\in\mathbb{N}_+$.

Now we want to show equation 13 holds for any $t\in\mathbb{N}_+$. Since $\bar{f}_\theta^{\bar{\zeta}_t}(\bar{X}_i)=\bar{f}_\theta(\bar{X}_i)$ and $\hat{f}_\theta^{\bar{\zeta}_t}(\bar{X}_i)=\hat{f}_\theta(\bar{X}_i)$ for all $i\in[m]$,

$$\begin{aligned}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,2}(\theta,\lambda)}{\partial\lambda}\right]&=\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{1}{m}\sum_{i=1}^m\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\lambda}\right]\\&=\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{1}{m}\sum_{i=1}^m\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right].\end{aligned}\tag{29}$$

Since we independently and uniformly sample $\bar{S}_1,\bar{S}_2,\ldots,\bar{S}_{|\bar{\zeta}_t(\bar{X}_i)|}$ from $\zeta_t(\bar{X}_i)$ and $\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}$ is constant with respect to the sampling,

$$\begin{aligned}\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right]&=\mathbb{E}_{\bar{S}_1,\bar{S}_2,\ldots,\bar{S}_{|\bar{\zeta}_t(\bar{X}_i)|}}\left[\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right]\\&=\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{j=1}^{|\bar{\zeta}_t(\bar{X}_i)|}\mathbb{E}_{\bar{S}_j}\left[\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right]\\&=\frac{1}{|\bar{\zeta}_t(\bar{X}_i)|}\sum_{j=1}^{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\\&=\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}.\end{aligned}\tag{30}$$

Since we sample a mini-batch $((\bar{X}_i,\bar{y}_i))_{i=1}^m$ independently and uniformly from the whole training set $((X_i,y_i))_{i=1}^n$, if we apply equation 30 to equation 29, we get

$$\begin{aligned}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\left[\frac{1}{m}\sum_{i=1}^m\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right]&=\frac{1}{m}\sum_{i=1}^m\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\left[\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta(\bar{X}_i))}{\partial\lambda}\right]\\&=\frac{1}{m}\sum_{i=1}^m\left(\frac{1}{n}\sum_{j=1}^n\frac{\partial(\ell_j\circ g_\lambda)(f_\theta(X_j))}{\partial\lambda}\right)\\&=\frac{1}{n}\sum_{j=1}^n\frac{\partial(\ell_j\circ g_\lambda)(f_\theta(X_j))}{\partial\lambda}\\&=\frac{\partial L(\theta,\lambda)}{\partial\lambda}\end{aligned}\tag{31}$$

Therefore, we conclude that

$$\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,2}(\theta,\lambda)}{\partial\lambda}\right]=\frac{\partial L(\theta,\lambda)}{\partial\lambda}.$$

Since t was arbitrary, this holds for any $t\in\mathbb{N}_+$. □

A.7. Unbiased Estimation of Full Set Gradient for MIL

Corollary A.1. *For multiple instance learning, given a training set $((X_i, y_i))_{i=1}^n$, we define the loss as follows:*

$$\begin{aligned} \mathbf{z}_{i,1} &= \max\{w^\top \phi(\mathbf{x}_{i,j}) + b\}_{j=1}^{N_i}, \quad \mathbf{z}_{i,2} = f_\theta(X_i), \\ \mathcal{L}_i &= \frac{1}{2}\left(\ell(\mathbf{z}_{i,1}, y_i) + \ell(g_\lambda(\mathbf{z}_{i,2}), y_i)\right), \\ L(\theta, \lambda, w, b) &= \frac{1}{n}\sum_{i=1}^n \mathcal{L}_i. \end{aligned}$$

For every iteration $t \in \mathbb{N}_+$ we sample a mini-batch of training data $((\bar{X}_i, \bar{y}_i))_{i=1}^m \sim D[((X_i, y_i))_{i=1}^n]$ and sample random partition $(\zeta_t(\bar{X}_i))_{i=1}^m$. Then we sample a mini-batch of subsets $\bar{\zeta}_t(\bar{X}_i) \sim D([\zeta_t(\bar{X}_i)]$. Let $\psi := (w, b) \in \mathbb{R}^{d_h+1}$ and define a function $h_\psi : \mathcal{X} \to \mathbb{R}$ by $h_\psi(X) := \max\{w^\top\phi(\mathbf{x}) + b : \mathbf{x} \in X\}$. Similar to equation 9 we define $h_\psi^{\bar{\zeta}_t}$ as,

$$\begin{aligned} h_\psi^{\bar{\zeta}_t}(\bar{X}_i) &:= h_\psi^{\bar{\zeta}_t, \zeta_t}(\bar{X}_i) \\ &:= \max\{h_\psi(S), \texttt{StopGrad}(h_\psi(S')) \mid S \in \bar{\zeta}_t(\bar{X}_i), S' \in \zeta_t(\bar{X}_i) \setminus \bar{\zeta}_t(\bar{X}_i)\}. \end{aligned}$$

Then we update the parameters θ, λ and ψ using the gradient of the following functions as

$$L_{t,1}(\theta, \lambda, \psi) = \frac{1}{2m}\sum_{i=1}^m \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\left(\ell(h^{\bar{\zeta}_t}(\bar{X}_i), \bar{y}_i) + \ell(g_\lambda(f_\theta^{\bar{\zeta}_t}(\bar{X}_i), \bar{y}_i))\right)$$

$$L_{t,2}(\theta, \lambda) = \frac{1}{2m}\sum_{i=1}^m \ell(g_\lambda(f_\theta^{\bar{\zeta}_t}(\bar{X}_i), \bar{y}_i))$$

$$\begin{aligned} \theta_{t+1} &= \theta_t - \eta_t \frac{\partial L_{t,1}(\theta_t, \lambda_t, \psi_t)}{\partial \theta_t} \\ \psi_{t+1} &= \psi_t - \eta_t \frac{\partial L_{t,1}(\theta_t, \lambda_t, \psi_t)}{\partial \psi_t} \\ \lambda_{t+1} &= \lambda_t - \eta_t \frac{\partial L_{t,2}(\theta_t, \lambda_t)}{\partial \lambda_t}, \end{aligned}$$

where $\eta_t > 0$ is a learning rate. If we assume that there exists a unique maximum value $\max\{w^\top\phi(\mathbf{x}) + b : \mathbf{x} \in X_i\} \in \mathbb{R}$ for each $i \in \{1, \ldots, n\}$ and sample a single subset from $\zeta_t(\bar{X}_i)$ for $i \in \{1, \ldots, m\}$, i.e. *$|\bar{\zeta}_t(\bar{X}_i)| = 1$, then the following holds:*

$$\mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta, \lambda, \psi)}{\partial \theta}\right] = \frac{\partial L(\theta, \lambda, \psi)}{\partial \theta} \tag{32}$$

$$\mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta, \lambda, \psi)}{\partial \psi}\right] = \frac{\partial L(\theta, \lambda, \psi)}{\partial \psi} \tag{33}$$

$$\mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,2}(\theta, \lambda)}{\partial \lambda}\right] = \frac{\partial L(\theta, \lambda, \psi)}{\partial \lambda} \tag{34}$$

Proof. It is enough to show the equation 32 and 33 hold since we have already proved equation 34, which does not depend on ψ, in Theorem 3.10. By defining $\ell_i(q) := \ell(q, y_i) \in \mathbb{R}$, $u_{i,j} = w^\top\phi(\mathbf{x}_{i,j}) + b$ where $\mathbf{x}_{i,j} \in X_i$ for $j \in \{1, \ldots, N_i\}$, and $u_{i,M} = \max\{u_{i,j}\}_{j=1}^{N_i}$,

$$\begin{aligned} \frac{\partial L(\theta, \lambda, \psi)}{\partial \psi} &= \frac{1}{n}\sum_{i=1}^n \frac{1}{2}\frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial h_\psi(X_i)}{\partial \psi} \\ &= \frac{1}{2n}\sum_{i=1}^n \frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial \max\{h_\psi(S) \mid S \in \zeta(X_i)\}}{\partial \psi} \\ &= \frac{1}{2n}\sum_{i=1}^n \frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial u_{i,M}}{\partial \psi} \end{aligned}$$

for any partition $\zeta(X_i)$ for all $i \in \{1, \dots, m\}$. Similarly, we define $\bar{\ell}_i(q) \coloneqq \ell(q, \bar{y}_i) \in \mathbb{R}$, $\bar{u}_{i,j} \coloneqq w^\top \phi(\bar{\mathbf{x}}_{i,j}) + b$ where $\bar{\mathbf{x}}_{i,j} \in \bar{X}_i$ for $j \in \{1, \dots, N_i\}$, and $\bar{u}_{i,M} = \max\{\bar{u}_{i,j}\}_{j=1}^{N_i}$. Let $t \in \mathbb{N}_+$ be fixed and define,

$$\bar{B}_{t,i} \coloneqq \{h_\psi(S) : S \in \bar{\zeta}_t(\bar{X}_i)\}.$$

With linearity of expectation and properties of the max operation, we have that

$$\begin{aligned}
&\mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m} \left[\frac{\partial L_{t,1}(\theta, \lambda, \psi)}{\partial \psi} \right] \\
&= \frac{1}{2m} \sum_{i=1}^m \mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m} \left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \mathbb{1}\{\bar{u}_{i,M} \in \bar{B}_{t,i}\} \frac{\partial \bar{u}_{i,M}}{\partial \psi} \right]
\end{aligned} \tag{35}$$

Note that we partition the set $\bar{X}_i$ and there is a unique maximum value. Thus, only one subset $S \in \zeta_t(\bar{X}_i)$ includes the element leading to the maximum value $\bar{u}_{i,M}$. If we do not choose such a subset S, the gradient in equation 35 becomes zero. Since we sample uniformly a single subset $\bar{S}$ from $\zeta_t(\bar{X}_i)$, *i.e.* $\bar{\zeta}_t(\bar{X}_i) = \{\bar{S}\}$, we get

$$\begin{aligned}
\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)} \left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \mathbb{1}\{\bar{u}_{i,M} \in \bar{B}_{t,i}\} \frac{\partial \bar{u}_{i,M}}{\partial \psi} \right] &= \mathbb{E}_{\bar{S}} \left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \mathbb{1}\{\bar{u}_{i,M} = h_\psi(\bar{S})\} \frac{\partial \bar{u}_{i,M}}{\partial \psi} \right] \\
&= \frac{|\zeta_t(\bar{X}_i)|}{1} \frac{1}{|\zeta_t(\bar{X}_i)|} \frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \frac{\partial \bar{u}_{i,M}}{\partial \psi} \\
&= \frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \frac{\partial \bar{u}_{i,M}}{\partial \psi}
\end{aligned} \tag{36}$$

If we apply the right hand side of equation 36 to equation 35, we obtain

$$\begin{aligned}
\mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m} \left[\frac{\partial L_{t,1}(\theta, \lambda, \psi)}{\partial \psi} \right] &= \frac{1}{2m} \sum_{i=1}^m \mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \left[\frac{\partial \bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)} \frac{\partial \bar{u}_{i,M}}{\partial \psi} \right] \\
&= \frac{1}{2m} \sum_{i=1}^m \frac{1}{n} \sum_{j=1}^n \frac{\partial \ell_j(h_\psi(X_j))}{\partial h_\psi(X_j)} \frac{\partial u_{j,M}}{\partial \psi} \\
&= \frac{1}{2n} \sum_{i=1}^n \frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)} \frac{\partial u_{i,M}}{\partial \psi} \\
&= \frac{\partial L(\theta, \lambda, \psi)}{\partial \psi}.
\end{aligned}$$

With the chain rule, we get,

$$\frac{\partial L(\theta, \lambda, \psi)}{\partial \theta} = \frac{1}{2n} \sum_{i=1}^n \left(\frac{\partial \ell_i(h_\psi(X_i))}{\partial \theta} + \frac{\partial (\ell_i \circ g_\lambda)(f_\theta(X_i))}{\partial \theta} \right).$$

Since we have already shown that

$$\frac{1}{2n} \sum_{i=1}^n \frac{\partial (\ell_i \circ g_\lambda)(f_\theta(X_i))}{\partial \theta} = \mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m} \left[\frac{1}{2m} \sum_{i=1}^m \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \frac{\partial (\bar{\ell}_i \circ f_\theta^{\bar{\zeta}_t})(\bar{X}_i)}{\partial \theta} \right] \tag{37}$$

in Theorem 3.10, it suffices to show that

$$\frac{1}{2n} \sum_{i=1}^n \frac{\partial \ell_i(h_\psi(X_i))}{\partial \theta} = \mathbb{E}_{((\bar{X}_i, \bar{y}_i))_{i=1}^m} \mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m} \left[\frac{1}{2m} \sum_{i=1}^m \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \frac{\partial \bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial \theta} \right]. \tag{38}$$

For the left hand side of equation 38, we have that

$$\begin{aligned}\frac{1}{2n}\sum_{i=1}^{n}\frac{\partial \ell_i(h_\psi(X_i))}{\partial\theta} &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial \max\{h_\psi(S)\mid S\in\zeta(X_i)\}}{\partial\theta}\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial u_{i,M}}{\partial\theta}\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}\frac{\partial (w^\top\phi(\mathbf{x}_{i,M})+b)}{\partial\theta}\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial \ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}w^\top\frac{\partial \phi(\mathbf{x}_{i,M})}{\partial\theta}.\end{aligned} \tag{39}$$

For the right hand side of equation 38, with linearity of expectation, we obtain

$$\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{1}{2m}\sum_{i=1}^{m}\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}\right] = \frac{1}{2m}\sum_{i=1}^{m}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}\right]. \tag{40}$$

Now we expand the summand in the right hand side,

$$\begin{aligned}\mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}\right] &= \mathbb{E}_{\bar{\zeta}_t(\bar{X}_i)}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)}\mathbb{1}\{\bar{u}_{i,M}\in\bar{B}_{t,i}\}\frac{\partial\bar{u}_{i,M}}{\partial\theta}\right]\\ &= \mathbb{E}_{\bar{S}}\left[\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)}\mathbb{1}\{\bar{u}_{i,M}=h_\psi(\bar{S})\}\frac{\partial\bar{u}_{i,M}}{\partial\theta}\right]\\ &= \frac{|\zeta_t(\bar{X}_i)|}{1}\frac{1}{|\zeta_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)}\frac{\partial\bar{u}_{i,M}}{\partial\theta}\\ &= \frac{\partial\bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)}w^\top\frac{\partial\phi(\bar{\mathbf{x}}_{i,M})}{\partial\theta}.\end{aligned} \tag{41}$$

Now we apply the right hand side of equation 41 to equation 40. Then we get,

$$\begin{aligned}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{1}{2m}\sum_{i=1}^{m}\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\frac{\partial\bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}\right] &= \frac{1}{2m}\sum_{i=1}^{m}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\left[\frac{\partial\bar{\ell}_i(h_\psi(\bar{X}_i))}{\partial h_\psi(\bar{X}_i)}w^\top\frac{\partial\phi(\bar{\mathbf{x}}_{i,M})}{\partial\theta}\right]\\ &= \frac{1}{2m}\sum_{l=1}^{m}\frac{1}{n}\sum_{i=1}^{n}\frac{\partial\ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}w^\top\frac{\partial\phi(\mathbf{x}_{i,M})}{\partial\theta}\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial\ell_i(h_\psi(X_i))}{\partial h_\psi(X_i)}w^\top\frac{\partial\phi(\mathbf{x}_{i,M})}{\partial\theta}\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial\ell_i(h_\psi(X_i))}{\partial\theta}.\end{aligned} \tag{42}$$

Finally combining equation 37 and equation 42, we arrive at the the conclusion:

$$\begin{aligned}&\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{\partial L_{t,1}(\theta,\lambda,\psi)}{\partial\theta}\right]\\ &= \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}\left[\frac{1}{2m}\sum_{i=1}^{m}\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}\left(\frac{\partial\bar{\ell}_i(h_\psi^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}+\frac{\partial(\bar{\ell}_i\circ g_\lambda)(f_\theta^{\bar{\zeta}_t}(\bar{X}_i))}{\partial\theta}\right)\right]\\ &= \frac{1}{2n}\sum_{i=1}^{n}\frac{\partial\ell_i(h_\psi(X_i))}{\partial\theta}+\frac{1}{2n}\sum_{i=1}^{n}\frac{\partial(\ell_i\circ g_\lambda)(f_\theta(X_i))}{\partial\theta}\\ &= \frac{\partial L(\theta,\lambda,\psi)}{\partial\theta}.\end{aligned}$$

□

A.8. SSE's Training Method Is a Biased Approximation to the Full Set Gradient

In this section, we show that sampling a single subset, and computing the gradient as an approximation to the gradient of $L(\theta, \lambda)$, which is proposed by Bruno et al. (2021), is a biased estimation of full set gradient. Since $\hat{f}_\theta$ with an attention activation function comprised of ν_1 and a sigmoid for σ is equivalent to a Slot Set Encoder, and is a special case of UMBC, we focus on the gradient of $\hat{f}_\theta$. Specifically, at every iteration $t \in \mathbb{N}_+$, we sample a mini-batch $((\bar{X}_i, \bar{y}_i))_{i=1}^m$ from the training dataset $((X_i, y_i))_{i=1}^n$. We choose a partition $\zeta_t(\bar{X}_i)$ for each $\bar{X}_i$ and sample a single subset $\bar{S}_i$ from the partition $\zeta_t(\bar{X}_i)$. If we compute the gradient of the loss as

$$\frac{\partial}{\partial \lambda}\left(\frac{1}{m}\sum_{i=1}^m (\bar{\ell}_i \circ g_\lambda)(\hat{f}_\theta(\bar{S}_i))\right), \tag{43}$$

then it is a *biased estimation* of $\frac{\partial L(\theta,\lambda)}{\partial \lambda}$, where $\bar{\ell}_i(\cdot)$ is defined by $\bar{\ell}_i(q) \coloneqq \ell(q, \bar{y}_i)$.

Proof. The gradient of $L(\theta, \lambda)$ with respect to the parameter λ is

$$\begin{aligned}\frac{\partial L(\theta,\lambda)}{\partial \lambda} &= \frac{1}{n}\sum_{i=1}^n \frac{\partial (\ell_i \circ g_\lambda)(\hat{f}_\theta(X_i))}{\partial \lambda} \\ &= \frac{1}{n}\sum_{i=1}^n \frac{\partial (\ell_i \circ g_\lambda)\left(\sum_{S \in \zeta(X_i)} \hat{f}_\theta(S)\right)}{\partial \lambda}\end{aligned} \tag{44}$$

for a partition $\zeta_t(X_i)$ of the set X_i, where $\ell_i(\cdot)$ is defined by $\ell_i(q) \coloneqq \ell(q, y_i)$. However, the expectation of equation 43 is not equal to the full set gradient in equation 44:

$$\begin{aligned}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{\bar{S}_i}\left[\frac{1}{m}\sum_{i=1}^m \frac{\partial(\bar{\ell}_i \circ g_\lambda)(\hat{f}_\theta(\bar{S}_i))}{\partial \lambda}\right] &= \frac{1}{m}\sum_{i=1}^m \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{\bar{S}_i}\left[\frac{\partial(\bar{\ell}_i \circ g_\lambda)(\hat{f}_\theta(\bar{S}_i))}{\partial \lambda}\right] \\ &= \frac{1}{n}\sum_{i=1}^n \frac{1}{|\zeta_t(X_i)|}\sum_{S \in \zeta_t(X_i)} \frac{\partial(\ell_i \circ g_\lambda)(\hat{f}_\theta(S))}{\partial \lambda} \\ &\neq \frac{1}{n}\sum_{i=1}^n \frac{\partial(\ell_i \circ g_\lambda)\left(\sum_{S\in\zeta_t(X_i)} \hat{f}_\theta(S)\right)}{\partial \lambda} \\ &= \frac{1}{n}\sum_{i=1}^n \frac{\partial(\ell_i \circ g_\lambda)(\hat{f}_\theta(X_i))}{\partial \lambda}\end{aligned} \tag{45}$$

To see why this is the case, we analyze the case of real valued function $g_\lambda : \mathbb{R}^d \to \mathbb{R}$ with $\lambda \in \mathbb{R}^d$ and a squared loss function

$$\begin{aligned}\ell(z_i, y_i) &\coloneqq \frac{1}{2}(z_i - y_i)^2 \\ z_i &\coloneqq \lambda^\top \hat{f}_\theta(X_i) \coloneqq g_\lambda(\hat{f}_\theta(X_i)) \in \mathbb{R}.\end{aligned}$$

Since $\hat{f}_\theta$ is sum decomposable, *i.e.* $\hat{f}_\theta(X_i) = \sum_{j=1}^{N_i} \hat{f}_\theta(\mathbf{x}_{i,j})$ where $X_i = \{\mathbf{x}_{i,1}, \ldots, \mathbf{x}_{i,N_i}\}$, the full set gradient from equation 44 becomes,

$$
\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\frac{\partial}{\partial\lambda}\left(\frac{1}{2}(z_i-y_i)^2\right) &= \frac{1}{n}\sum_{i=1}^{n}(z_i-y_i)\frac{\partial z_i}{\partial\lambda} \\
&= \frac{1}{n}\sum_{i=1}^{n}(z_i-y_i)\hat{f}_\theta(X_i) \\
&= \frac{1}{n}\sum_{i=1}^{n}(z_i-y_i)\left(\sum_{l=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,l})\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\lambda^\top\left(\sum_{j=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,j})\right)-y_i\right)\left(\sum_{l=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,l})\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\left(\sum_{j=1}^{N_i}\lambda^\top\hat{f}_\theta(\mathbf{x}_{i,j})\right)-y_i\right)\left(\sum_{l=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,l})\right). \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{N_i}\lambda^\top\hat{f}_\theta(\mathbf{x}_{i,j})-\frac{y_i}{N_i}\right)\left(\sum_{l=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,l})\right).
\end{aligned}
\tag{46}
$$

Assume that $\zeta_t(X_i)=\{\{\mathbf{x}_{i,1}\},\ldots,\{\mathbf{x}_{i,N_i}\}\}$ and we sample a single subset $\bar{S}_i$ from the partition $\zeta_t(\bar{X}_i)$ for all $i\in\{1,\ldots,m\}$ and $t\in\mathbb{N}_+$. Then gradient of the subsampling a single subset from equation 45 becomes,

$$
\begin{aligned}
\frac{1}{n}\sum_{i=1}^{n}\frac{1}{N_i}\sum_{j=1}^{N_i}\frac{1}{2}\frac{\partial\left(\lambda^\top\hat{f}_\theta(\mathbf{x}_{i,j})-y_i\right)^2}{\partial\lambda} &= \frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{N_i}\left(\frac{\lambda^\top\hat{f}_\theta(\mathbf{x}_{i,j})-y_i}{N_i}\right)\hat{f}_\theta(\mathbf{x}_{i,j}) \\
&\neq \frac{1}{n}\sum_{i=1}^{n}\left(\sum_{j=1}^{N_i}\lambda^\top\hat{f}_\theta(\mathbf{x}_{i,j})-\frac{y_i}{N_i}\right)\left(\sum_{l=1}^{N_i}\hat{f}_\theta(\mathbf{x}_{i,l})\right).
\end{aligned}
\tag{47}
$$

Therefore, the random subsampling of a a single subset in the method proposed by Bruno et al. (2021) *is not* an unbiased estimate of the gradient of the full set. □

B. Optimization

Define $\bar{\theta}=(\theta,\lambda)$ and $g_t(\bar{\theta})=(\frac{\partial L_{t,1}(\theta,\lambda)}{\partial\theta},\frac{\partial L_{t,2}(\theta,\lambda)}{\partial\lambda})^\top$. We assume that $\ell(q,y)\geq 0$ for all (q,y). Let $(\bar{\theta}_t)_{t\in\mathbb{N}_+}$ be a sequence generated by $\bar{\theta}_{t+1}=\bar{\theta}_t-\eta_t g_t(\bar{\theta})$ with an initial point $\bar{\theta}_1$ and a step size sequence $(\eta_t)_{t\in\mathbb{N}_+}$, where $\bar{\theta}_t\in\mathcal{R}\subseteq\mathbb{R}^D$ for $t\in\mathbb{N}_+$ with an open convex set $\mathcal{R}$. Here, $\mathbb{R}^D$ is an open set and thus it is allowed to choose $\mathcal{R}=\mathbb{R}^D$ (or any other open convex set). We do not assume that the loss function or model is convex. We also do not make any assumption on the initial point $\bar{\theta}_1$. To analyze the optimization behavior formally, we consider the following standard assumption in the literature (Lee et al., 2016; Mertikopoulos et al., 2020; Fehrman et al., 2020):

Assumption B.1. There exist $\varsigma>0$ such that for any $\bar{\theta},\bar{\theta}'\in\mathcal{R}$, $t\in[T]$, and $k\in\{1,2\}$,

$$\|\nabla L_{t,k}(\bar{\theta})-\nabla L_{t,k}(\bar{\theta}')\|_2\leq\varsigma\|\bar{\theta}-\bar{\theta}'\|_2.$$

We use the following lemma on a general function from a previous work (Kawaguchi et al., 2022, Lemma 2):

Lemma B.2. *For any differentiable function* $\varphi:\mathrm{dom}(\varphi)\to\mathbb{R}$ *with an open convex set* $\mathrm{dom}(\varphi)\subseteq\mathbb{R}^{n_\varphi}$, *if* $\|\nabla\varphi(z')-\nabla\varphi(z)\|_2\leq\varsigma_\varphi\|z'-z\|_2$ *for all* $z,z'\in\mathrm{dom}(\varphi)$, *then*

$$\varphi(z')\leq\varphi(z)+\nabla\varphi(z)^\top(z'-z)+\frac{\varsigma_\varphi}{2}\|z'-z\|_2^2\quad\text{for all } z,z'\in\mathrm{dom}(\varphi). \tag{48}$$

In turn, Lemma B.2 implies the following lemma:

Lemma B.3. *For any differentiable function $\varphi : \operatorname{dom}(\varphi) \to \mathbb{R}_{\geq 0}$ with an open convex set $\operatorname{dom}(\varphi) \subseteq \mathbb{R}^{n_\varphi}$ such that $\|\nabla\varphi(z') - \nabla\varphi(z)\| \leq \varsigma_\varphi \|z' - z\|$ for all $z, z' \in \operatorname{dom}(\varphi)$, the following holds: for all $z \in \operatorname{dom}(\varphi)$ such that $z - \frac{1}{\varsigma_\varphi}\nabla\varphi(z) \in \operatorname{dom}(\varphi)$,*

$$\|\nabla\varphi(z)\|_2^2 \leq 2\varsigma_\varphi \varphi(z) \quad \textit{for all } z \in \operatorname{dom}(\varphi). \tag{49}$$

Proof. Since $\varphi : \operatorname{dom}(\varphi) \to \mathbb{R}_{\geq 0}$ (nonnegative), if $\nabla\varphi(z) = 0$, the desired statement holds. Thus, we consider the remaining case of $\nabla\varphi(z) \neq 0$ in the rest of this proof. We invoke Lemma B.2 with $z' = z - \frac{1}{\varsigma_\varphi}\nabla\varphi(z)$, yielding

$$\begin{aligned} 0 \leq \varphi(z') &\leq \varphi(z) + \nabla\varphi(z)^\top (z' - z) + \frac{\varsigma_\varphi}{2}\|z' - z\|_2^2 \\ &= \varphi(z) - \frac{1}{\varsigma_\varphi}\|\nabla\varphi(z)\|_2^2 + \frac{1}{2\varsigma_\varphi}\|\nabla\varphi(z)\|_2^2 \\ &= \varphi(z) - \frac{1}{2\varsigma_\varphi}\|\nabla\varphi(z)\|_2^2 \end{aligned}$$

By rearranging, this implies that $\|\nabla\varphi(z)\|_2^2 \leq 2\varsigma_\varphi \varphi(z)$. □

Since we are dealing with a general non-convex and non-invex function (as the choice of architecture and loss is very flexible) where gradient-based optimization might only converge to a stationary point (to avoid the curse of dimensionality), we consider the convergence in terms of stationary points of L:

Theorem B.4. *Suppose that Assumption B.1 holds and the step size sequence $(\eta_t)_{t \in \mathbb{N}_+}$ satisfies $\sum_{t=1}^\infty \eta_t^2 < \infty$. Then, there exists a constant c independent of (t, T) such that*

$$\min_{t \in [T]} \mathbb{E}[\|\nabla L(\bar\theta_t)\|_2^2] \leq \frac{c\mathbb{E}[L(\bar\theta_1)]}{\sum_{t=1}^T \eta_t}.$$

Proof. Assumption B.1 implies that

$$\|\nabla L_{t,k}(\bar\theta) - \nabla L_{t,k}(\bar\theta')\|_2^2 \leq \varsigma^2 \|\bar\theta - \bar\theta'\|_2^2$$

which implies that

$$\|g_t(\bar\theta) - g_t(\bar\theta')\|_2^2 = \|\nabla L_{t,1}(\bar\theta) - \nabla L_{t,1}(\bar\theta')\|_2^2 + \|\nabla L_{t,2}(\bar\theta) - \nabla L_{t,2}(\bar\theta')\|_2^2 \leq 2\varsigma^2 \|\bar\theta - \bar\theta'\|_2^2.$$

This implies that

$$\|g_t(\bar\theta) - g_t(\bar\theta')\|_2 \leq \sqrt{2}\varsigma \|\bar\theta - \bar\theta'\|_2$$

Using this and Theorem 3.10 along with Jensen's inequality, we have that for any $\bar\theta, \bar\theta' \in \mathcal{R}$,

$$\|\nabla L(\bar\theta) - \nabla L(\bar\theta')\|_2 = \|\mathbb{E}[g_t(\bar\theta)] - \mathbb{E}[g_t(\bar\theta')]\|_2 \leq \mathbb{E}[\|g_t(\bar\theta) - g_t(\bar\theta')\|_2] \leq \sqrt{2}\varsigma \|\bar\theta - \bar\theta'\|_2.$$

Thus, L satisfies the conditions of Lemma B.2 and Lemma B.3. Since $\bar\theta_t \in \mathcal{R} \subseteq \mathbb{R}^D$ for $t \in \mathbb{N}_+$, using Lemma B.2 for the function L, we have that

$$L(\bar\theta_{t+1}) \leq L(\bar\theta_t) + \nabla L(\bar\theta_t)^\top (\bar\theta_{t+1} - \bar\theta_t) + \frac{\sqrt{2}\varsigma \|\bar\theta_{t+1} - \bar\theta_t\|_2^2}{2}.$$

Using $\bar\theta_{t+1} - \bar\theta_t = -\eta_t g_t(\bar\theta_t)$,

$$L(\bar\theta_{t+1}) \leq L(\bar\theta_t) - \eta_t \nabla L(\bar\theta_t)^\top g_t(\bar\theta_t) + \frac{\sqrt{2}\varsigma \eta_t^2 \|g_t(\bar\theta_t)\|_2^2}{2}.$$

Using Lemma B.3 for $\|g_t(\bar\theta_t)\|_2^2 = \|\frac{\partial L_{t,1}(\theta_t, \lambda_t)}{\partial \theta_t}\|_2^2 + \|\frac{\partial L_{t,2}(\theta_t, \lambda_t)}{\partial \lambda_t}\|_2^2$, we have that

$$L(\bar\theta_{t+1}) \leq L(\bar\theta_t) - \eta_t \nabla L(\bar\theta_t)^\top g_t(\bar\theta) + \sqrt{2}\varsigma^2 \eta_t^2 (L_{t,1}(\bar\theta_t) + L_{t,2}(\bar\theta_t)).$$

Define $L_t(\bar{\theta}_t) = L_{t,1}(\bar{\theta}_t) + L_{t,2}(\bar{\theta}_t)$. Using the linearity and monotonicity of expectation,

$$\begin{aligned}
&\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L(\bar{\theta}_{t+1})|\bar{\theta}_t] \\
&\leq L(\bar{\theta}_t) - \eta_t \nabla L(\bar{\theta}_t)^\top \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[g_t(\bar{\theta})] + \sqrt{2}\varsigma^2\eta_t^2 \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L_t(\bar{\theta}_t)] \\
&\leq L(\bar{\theta}_t) - \eta_t\|\nabla L(\bar{\theta}_t)\|_2^2 + \sqrt{2}\varsigma^2\eta_t^2(1+a)L(\bar{\theta}_t)
\end{aligned}$$

where the second inequality follows from Theorem 3.10 and $\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L_t(\bar{\theta}_t)] = \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L_{t,1}(\bar{\theta}_t)] + \mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L_{t,2}(\bar{\theta}_t)] \leq (1+a)L(\bar{\theta}_t)$ where a is the expectation of the maximum ratio $\frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|}$.

Taking expectation over $\bar{\theta}_t$ with the law of total expectation $\mathbb{E}[L(\bar{\theta}_{t+1})] = \mathbb{E}_{\bar{\theta}_t}\mathbb{E}_{((\bar{X}_i,\bar{y}_i))_{i=1}^m}\mathbb{E}_{(\bar{\zeta}_t(\bar{X}_i))_{i=1}^m}[L(\bar{\theta}_{t+1})|\bar{\theta}_t]$,

$$\mathbb{E}[L(\bar{\theta}_{t+1})] \leq \mathbb{E}[L(\bar{\theta}_t)] - \eta_t\mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] + \sqrt{2}\varsigma^2\eta_t^2(1+a)\mathbb{E}[L(\bar{\theta}_t)]. \tag{50}$$

Since $\mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] \geq 0$, this implies that

$$\begin{aligned}
\mathbb{E}[L(\bar{\theta}_{t+1})] &\leq \mathbb{E}[L(\bar{\theta}_t)] + \sqrt{2}\varsigma^2\eta_t^2(1+a)\mathbb{E}[L(\bar{\theta}_t)] \\
&= (1+\sqrt{2}\varsigma^2\eta_t^2(1+a))\mathbb{E}[L(\bar{\theta}_t)] \\
&\leq \exp(\sqrt{2}\varsigma^2\eta_t^2(1+a))\mathbb{E}[L(\bar{\theta}_t)],
\end{aligned}$$

where the last inequality follows from $1+q \leq \exp(q)$ for all $q \in \mathbb{R}$. Applying this inequality recursively over t, it holds that for any $t \in \mathbb{N}_+$,

$$\mathbb{E}[L(\bar{\theta}_t)] \leq \exp\left(\sqrt{2}\varsigma^2(1+a)\sum_{j=1}^{t-1}\eta_j^2\right)\mathbb{E}[L(\bar{\theta}_1)].$$

Using this inequality in equation 50,

$$\mathbb{E}[L(\bar{\theta}_{t+1})] \leq \mathbb{E}[L(\bar{\theta}_t)] - \eta_t\mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] + \sqrt{2}\varsigma^2\eta_t^2(1+a)\exp\left(\sqrt{2}\varsigma^2(1+a)\sum_{j=1}^{t-1}\eta_j^2\right)\mathbb{E}[L(\bar{\theta}_1)].$$

Rearranging and summing over t with,

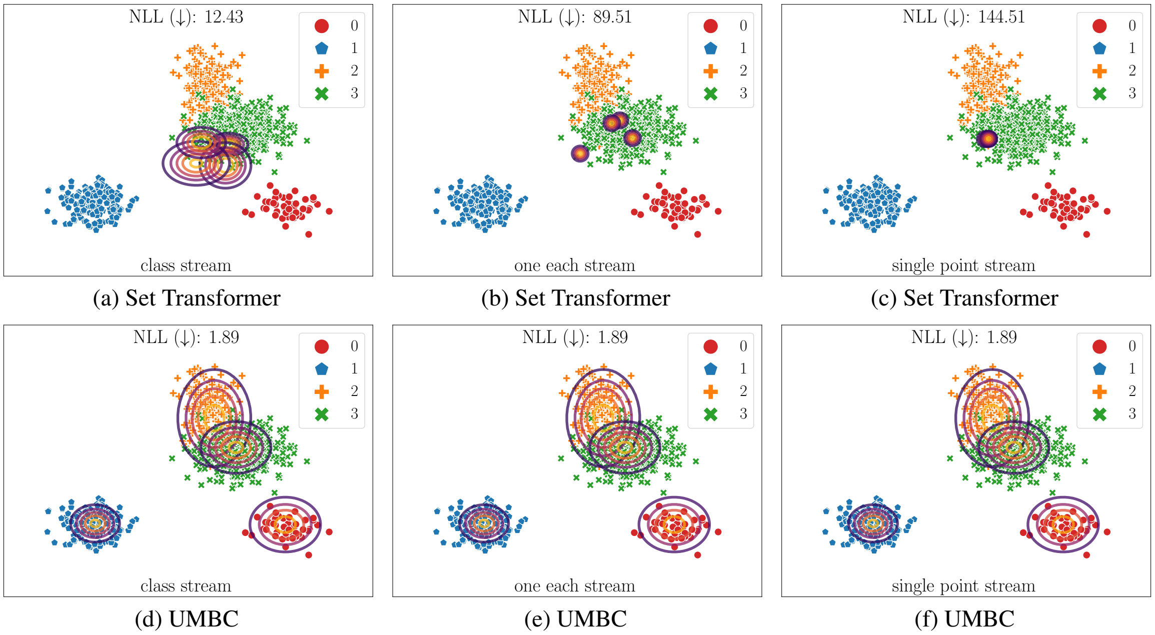
$$\begin{aligned}
\sum_{t=1}^{T}\eta_t\mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] &\leq \sum_{t=1}^{T}(\mathbb{E}[L(\bar{\theta}_t)] - \mathbb{E}[L(\bar{\theta}_{t+1})]) + \sqrt{2}\varsigma^2(1+a)\mathbb{E}[L(\bar{\theta}_1)]\sum_{t=1}^{T}\eta_t^2\exp\left(\sqrt{2}\varsigma^2(1+a)\sum_{j=1}^{t-1}\eta_j^2\right) \\
&\leq \mathbb{E}[L(\bar{\theta}_1)] - \mathbb{E}[L(\bar{\theta}_{T+1})] + \sqrt{2}\varsigma^2(1+a)\mathbb{E}[L(\bar{\theta}_1)]\exp\left(\sqrt{2}\varsigma^2(1+a)\sum_{j=1}^{T-1}\eta_j^2\right)\left(\sum_{t=1}^{T}\eta_t^2\right) \\
&= (1+\sqrt{2}\varsigma^2(1+a)R_T\exp(\sqrt{2}\varsigma^2(1+a)R_{T-1}))\mathbb{E}[L(\bar{\theta}_1)] - \mathbb{E}[L(\bar{\theta}_{T+1})]
\end{aligned}$$

where we define $R_T = \sum_{t=1}^{T}\eta_t^2$. Since $\eta_t > 0$ and $\mathbb{E}[\nabla\left\|L(\bar{\theta}_t)\right\|_2^2] > 0$ for all $t \in [T] := \{1,\ldots,T\}$,

$$\begin{aligned}
\min_{t\in[T]}\mathbb{E}[\left\|\nabla L(\bar{\theta}_t)\right\|_2^2]\left(\sum_{t=1}^{T}\eta_t\right) &= \sum_{t=1}^{T}\eta_t\min_{t'\in[T]}\mathbb{E}[\left\|\nabla L(\bar{\theta}_{t'})\right\|_2^2] \\
&\leq \sum_{t=1}^{T}\eta_t\mathbb{E}[\left\|\nabla L(\bar{\theta}_t)\right\|_2^2].
\end{aligned}$$

This implies that

$$\min_{t\in[T]}\mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] \leq \left(\sum_{t=1}^{T}\eta_t\right)^{-1}\left((1+\sqrt{2}\varsigma^2(1+a)R_T\exp(\sqrt{2}\varsigma^2(1+a)R_{T-1}))\mathbb{E}[L(\bar{\theta}_1)] - \mathbb{E}[L(\bar{\theta}_{T+1})]\right).$$



(a) Set Transformer (b) Set Transformer (c) Set Transformer

(d) UMBC (e) UMBC (f) UMBC

Figure 8. **Top Row**: Set Transformer provides inconsistent predictions on streaming sets when inputs cannot be stored directly. **Bottom Row**: UMBC+Set Transformer gives consistent predictions in all streaming settings.

Since $R_{T-1} \le R_T \le \sum_{t=1}^{\infty} \eta_t^2 < \infty$ and $\mathbb{E}[L(\bar{\theta}_{T+1})] \ge 0$, this implies that there exists a constant c independent of (t, T) such that

$$\min_{t\in[T]} \mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] \le c\mathbb{E}[L(\bar{\theta}_1)] \left(\sum_{t=1}^{T} \eta_t\right)^{-1}.$$

□

For example, if $\eta_t = \eta_1 t^{-q}$ with $q \in (0.5, 1)$ and $\eta_1 > 0$, then we have $\min_{t\in[T]} \mathbb{E}[\|\nabla L(\bar{\theta}_t)\|_2^2] = \mathcal{O}(\frac{1}{T^{1-q}})$.

C. Details on the Mixture of Gaussians Amortized Clustering Experiment

We used a modified version of the MoG amortized clustering dataset which was used by Lee et al. (2019). We modified the experiment, adding separate, random covariance parameters into the procedure in order to make a more difficult dataset. Specifically, to sample a single task for a problem with K classes,

1. Sample set size for the batch $N \sim U(\text{train set size}/2, \text{train set size})$.
2. Sample class priors $\pi \sim \text{Dirichlet}([\alpha_1, \dots, \alpha_K])$ with $\alpha_1 = \dots = \alpha_K = 1$.
3. Sample class labels $z_i \sim \text{Categorical}(\pi)$ for $i = 1, ..., N$.
4. Generate cluster centers $\boldsymbol{\mu}_i = (\mu_{i,1}, \mu_{i,2}) \in \mathbb{R}^2$, where $\mu_{i,j} \sim U(-4, 4)$ for $i = 1, ..., K$ and $j = 1, 2$.
5. Generate cluster covariance matrices $\boldsymbol{\Sigma}_i = \text{diag}(\sigma_{i,1}, \sigma_{i,2}) \in \mathbb{R}^{2\times 2}$, where $\sigma_{ij} \sim U(0.3, 0.6)$ for $i = 1, ..., K$ and $j = 1, 2$.
6. For all z_n, if $z_n = i$, sample data $\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$

In our MoG experiments, we set $K = 4$. The Motivational Example in Figure 2 also used the MoG dataset, and performed MBC testing of the set transformer corresponding to the procedure outlined in Appendix E

C.1. Streaming Settings

The four total streaming settings in Figures 2 and 8 are described below:

Table 6. MVN generic model (Used by all encoders)

Output Size	Layers	Amount
$N_i \times 2$	Input Set	$\times 1$
$N_i \times 128$	Linear(2, 128), ReLU	$\times 1$
$N_i \times 128$	Set Encoder	$\times 1$
$K \times 5$	Decoder	$\times 3$

Table 7. Set Encoder Specific settings for baseline models.

Name	Set Encoder	Output Size	Set Decoder	Output Size
DeepSets (Zaheer et al., 2017)	Mean Pooling	128	Linear, ReLU	128
SSE (Bruno et al., 2021)	Slot Set Encoder	$K \times 128$	Linear, ReLU	128
FSPool (Zhang et al., 2020)	Featurewise Sort Pooling	128	Linear, ReLU	128
Diff. EM. (Kim, 2022)	Expectation Maximization Layer	1286	Linear, ReLU	128
Set Transformer (Lee et al., 2019)	Pooling by Multihead Attention	$K \times 128$	Set Attention Block	$K \times 128$

Table 8. Set Encoder Specific settings for UMBC models. UMBC models account for the extra encoder by using fewer layers in the decoder (2 layers instead of 3).

Name	MBC Set Encoder	Output Size	non-MBC Set Encoder	Output Size	Set Decoder	Output Size
(Ours) UMBC+FSPool	UMBC Layer	$K \times 128$	Featurewise Sort Pooling	128	Linear, ReLU	128
(Ours) UMBC+Diff EM	UMBC Layer	$K \times 128$	Expectation Maximization Layer	1286	Linear, ReLU	128
(Ours) UMBC+Set Transformer	UMBC Layer	$K \times 128$	Set Attention Block	$K \times 128$	Linear, ReLU	$K \times 128$

- **single point stream** $\rightarrow$ streams each point in the set one by one. This causes the most severe under-performance by non-MBC models.
- **class stream** $\rightarrow$ streams an entire class at once. Models which make complex pairwise comparisons cannot compare the input class with any other clusters, thereby degrading performance of models such as the Set Transformer.
- **chunk stream** $\rightarrow$ streams 8 random points at a time from the dataset, Providing, random and limited information to non-MBC models.
- **one each stream** $\rightarrow$ streams a set consisting of a single instance from each class. non-MBC models can see examples of each class, but with a limited sample size, therefore non-MBC models such as Set Transformer fail to make accurate predictions.

C.2. Experimental Setup

We train each model for 50 epochs, with each epoch containing 1000 iterations. We use the Adam optimizer with a learning rate of $1 \cdot 10^{-3}$ and no weight decay. We do not perform early stopping. We make a single learning rate adjustment at epoch 35 which adjusts the learning rate to 10^{-4}. When measuring NLL for results, we measure the NLL of the full set of 1024 points. Unless otherwise specified, UMBC models use the softmax activation function. We list the architectures in Tables 6 to 8. All models have an additional linear output which outputs $K \times 5$ parameters for the Gaussian mixture outlined in Equation (14).

D. Measuring the Variance of Pooled Features

In Figure 3, we show the quantitative effect on the pooled representation between the plain Set Transformer, UMBC+Set Transformer, FSPool and DiffEM. The UMBC model always shows 0 variance, while the non-MBC models produce variance between aggregated encodings of random partitions. For a single chunk, however, non-MBC models show no variance, as random partitions of a single chunk would be equivalent to permuting the elements within the chunk (*i.e*. non-MBC models still produce an encoding which is permutation invariant). The variance increases drastically when the set is partitioned into two chunks and then the behavior differs between the non-MBC models. Set Transformer happens to show decreasing variance as the number of chunks increases. Note that as the number of chunks increases, the cardinality of each chunk decreases. Therefore, the variance decreases as the chunk cardinality also decreases, but this *does not* indicate that the models is performing better. For example, in Figure 2, when a singleton set is input to the Set Transformer, the predictions become almost meaningless even though they may have lower variance. The procedure for aggregating the encodings of set partitions for the non-MBC models is outlined in Appendix E.

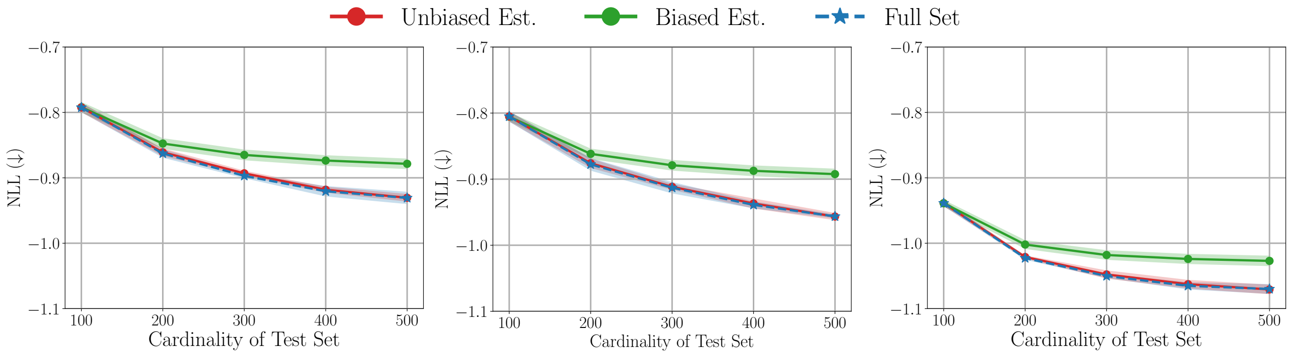
To perform this experiment, we used a randomly initialized model with 128 hidden units, and sampled 256 set elements from four different distributions in order to make a total set size of 1024. We then created 100 random partitions for various chunk sizes. Chunk sizes and distributions are shown in Appendix D. We then encode the whole set in chunks and and report the observed variance over the 100 different random partitions at each of the various chunk sizes (Figure 3). Note that the encoded set representation is a vector and Figure 3 shows a scalar value. To achieve this, we take the feature-wise variance over the 100 encodings and report the mean and standard deviation over the feature dimension. Specifically, given $\mathbf{Z} = [\mathbf{z}_1 \cdots \mathbf{z}_{100}]^\top \in \mathbb{R}^{100 \times 128}$ representing all 100 encodings with $\mathbf{z}_i = (z_{i,1}, \ldots, z_{i,128})$, we compute feature-wise

Figure 9. Distributions used in sampling random inputs for the encoding variance experiment in Figure 2

Distribution	Dimension	Number of Points
Normal(0, 1)	128	256
Uniform(-3, 3)	128	256
Exponential(1)	128	256
Cauchy(0, 1)	128	256

Figure 10. The number of chunks and elements per chunk.

Number of Chunks	1	2	4	8	16	32
Elements per Chunk	1024	512	256	128	64	32



(a) Deepsets (b) Slot Set Encoder (c) UMBC + ST

Figure 11. Performance of **(a)** DeepSets **(b)** SSE, and **(c)** UMBC with varying set sizes for image completion on CelebA dataset.

variance as

$$\hat{z}_j = \frac{1}{(100-1)}\sum_{i=1}^{100}(z_{i,j}-\mu_j)^2, \quad \mu_j = \frac{1}{100}\sum_{i=1}^{100} z_{i,j}$$

for $j = 1, \ldots, 128$. We then achieve the values of y-axis and error bars in Figure 3 by a mean and standard deviation over the feature dimension,

$$y = \frac{1}{128}\sum_i \hat{z}_i, \qquad y_\sigma = \sqrt{\frac{1}{(128-1)}\sum_i (\hat{z}_i - y)^2}. \tag{51}$$

E. A Note on MBC Testing of non-MBC models

In the qualitative experiments Figures 2 and 3, we apply MBC testing to non-MBC models in order to study the effects of using non-MBC models in MBC settings. Non-MBC models do not prescribe a way to accomplish this in the original works, so we took the approach of processing each chunk up until the pooled representation. We then performed mean pooling over the encoded chunks in the following way. Let X be an input set and let $\zeta(X) = \{X_1, \ldots, X_n\}$ be a partition of the set X, *i.e.* $X = \bigcup_{j=1}^{N} X_j$ with $X_i \cap X_j = \emptyset$ for $i \neq j$. Denote $\tilde{f}_\theta$ a non-MBC set encoding function, then our pseudo-MBC testing procedure is as follows,

$$Z = \frac{1}{N}\sum_{j=1}^{N} \tilde{f}_\theta(X_j) \tag{52}$$

F. Details on the Image Completion Experiments

F.1. Additional Experimental Results

In figure Figure 11, we evaluate our proposed unbiased full set gradient approximation algorithm (**red**) with Deepsets, Slot Set Encoder (SSE) and UMBC + Set Transformer (ST) and compare our algorithm against the one training with a randomly sampled subset of 100 elements , which is a biased estimator, (**green**) and the one computing full set gradient (**blue**). Across

Table 9. UMBC Set Encoder of Conditional Neural Process.

Output Size	**Layers**
$N_i \times 5$	Input Context Set
$N_i \times 128$	Linear(5, 128), ReLU
$N_i \times 128$	Linear(128, 128), ReLU
$N_i \times 128$	Linear(128, 128), ReLU
$N_i \times 128$	Linear(128, 128), ReLU
128×128	UMBC Layer
128×128	Layer Normalization
128×128	Set Attention Block (Lee et al., 2019)
128×128	Set Attention Block
128	Pooling by Multihead Attention (Lee et al., 2019)

Table 10. Decoder of Conditional Neural Process.

Output Size	**Layers**
$128, 1024 \times 2$	Input Set Representation and Coordinates
1024×130	Tile & Concatenate
1024×128	Linear(130, 128), ReLU
1024×128	Linear(128, 128), ReLU
1024×128	Linear(128, 128), ReLU
1024×128	Linear(128, 128), ReLU
1024×6	Linear(128, 6)

all models, our unbiased estimator significantly outperforms the models trained with a randomly sampled subset. Notably, the model trained with our proposed algorithm is indistinguishable from the model trained with full set gradient while our method only incurs constant memory overhead for any set size. These empirical results again verify efficiency of our unbiased full set gradient approximation.

F.2. Experimental Setup

We train all models on CelebA dataset for 200,000 steps with Adam optimizer (Kingma & Ba, 2015) and 256 batch size but no weight decay. We set the learning rate to $5 \cdot 10^{-4}$ and use a cosine annealing learning rate schedule. In Tables 9 and 10, we specify the architecture of Conditional Neural Process with UMBC + Set Transformer. We use $k = 128$ slots and set dimension of each slot to $d_s = 128$. For the attention layer, we use the softmax for the activation function σ and set the dimension of attention output to $d = 128$. As an input to the set encoder, we concatenate the coordinates of each $\mathbf{x}_{i,c_j} \in \mathbb{R}^2$ and the corresponding pixel value $y_{i,c_j} \in \mathbb{R}^3$ from the context X_i for $j = 1, \ldots, N_i$, resulting in a $N_i \times 5$ matrix.

G. Details on the Long Document Classification Experiments

We train all models for 30 epochs with AdamW optimizer (Loshchilov & Hutter, 2019) and batch size 8. We use constant learning rate $5 \cdot 10^{-5}$. For our UMBC model, we pretrain the model while freezing BERT for 30 epochs and finetune the whole model for another 30 epochs. In Table 11, we specify architecture of UMBC + BERT (Devlin et al., 2019) without positional encoding. We use $k = 256$ slots and set dimension of each slot to $d_s = 128$. We use slot-sigmoid for the activation function σ and set the dimension of the attention output to $d = 768$.

Table 11. UMBC Set Encoder and BERT decoder for long document classification.

Output Size	**Layers**
N_i	Input Document
$N_i \times 768$	Word Embedding
$N_i \times 768$	Layer Normalization
$N_i \times 768$	Linear(768,768), ReLU
$N_i \times 768$	Linear(768,768), ReLU
256×768	UMBC Layer
256×768	Layer Normalization
256×768	BERT w/o Positional Encoding (Devlin et al., 2019)
768	[CLS] token Pooler
4271	Dropout(0.1), Linear(768, 4271)

H. Details on the Camelyon16 Experiments

The Camelyon16 Whole Slide Image dataset consists of 270 training instances and 129 validation instances. The dataset was created for a competition, and therefore the test set is hidden. We therefore follow the example set by previous works (Li et al., 2021) and report performance achieved on the validation set. For preprocessing, we consider the 20× slide magnification setting, and use OTSU's thresholding method to detect regions containing tissue within the WSI. We then split the activated regions into non overlapping patches of size 256×256. An example of single input patches can be

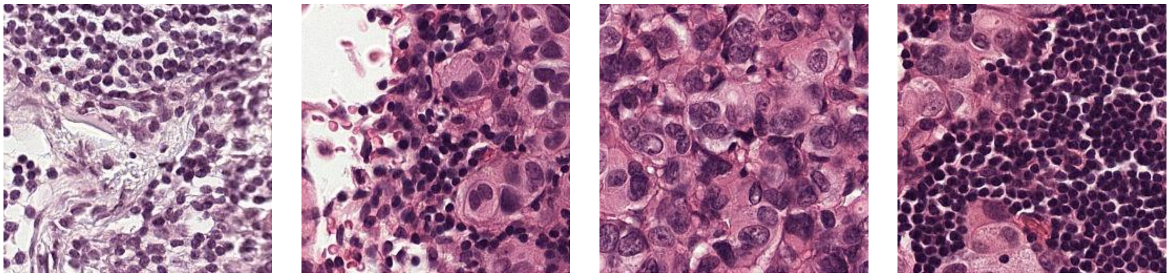


Figure 12. Examples of 4 single patches from the Camelyon16 dataset. On average, each set in the dataset contains over $9,300$ patches like the ones pictured above.

Table 12. Camelyon16 generic model (Used by all encoders)

Output Size	**Layers**	**Name**	**Amount**
$N_i \times 256 \times 256 \times 3$	Input Set	Bag of Instances	$\times 1$
$N_i \times 512$	ResNet18(InstanceNorm)	Feature Extractor	$\times 1$
$N_i \times 256$	Linear, ReLU, Linear	Projection	$\times 1$
1	Linear, Max Pooling	Instance Classifier	$\times 1$
$N_i \times 128$	Set Encoding Function	Bag Classifier	$\times 1$
1	Set Decoder	Bag Classifier	$\times 1$

Table 13. Camelyon16 Bag Classifier Models.

Name	**Set Encoder**	**Output Size**
AB-MIL	Gated Attention (Ilse et al., 2018)	1
DSMIL	DS-MIL Aggregator (Li et al., 2021)	1
Deepsets	Max Pooling → Linear → ReLU → Linear	1
Slot Set Encoder	SSE (Bruno et al., 2021)→ Linear → ReLU → Linear	1
UMBC+SetTransformer	UMBC(K=64) → SAB → PMA(K=1, p=0.5) → Linear	1

seen in Figure 12. The largest input set contains $37,345$ image patches which are each $\in \mathbb{R}^{256\times256\times3}$. All patch extraction code can be found in the supplementary file. Table 14 contains statistics related to the numbers of patches per input for the training and the test set as well as the distribution of positive and negative labels.

H.1. Experimental Setup

We use a ResNet18 (He et al., 2016) which was pretrained with self-supervised contrastive learning (Chen et al., 2020) by Li et al. (2021). The pretrained ResNet18 weights can be downloaded from this repository. Following the classification experiments done by Lee et al. (2019), we place dropout layers before and after the PMA layer of the Set Transformer in our UMBC model. We will describe our pretraining and finetuning steps below in detail.

Pretraining. For pretraining, we extract the features from the pretrained ResNet18 and only train the respective MIL models (Table 13) on the extracted features. We pretrain for 200 epochs with the Adam optimizer which uses a learning rate of $5 \cdot 10^{-4}$ and a cosine annealing learning rate decay which reaches the minimum at $5 \cdot 10^{-6}$. We use $\beta_1 = 0.5$, and $\beta_2 = 0.9$ for Adam. We train with a batch size of 1 on a single GPU, and save the model which showed the best performance on the validation set, where the performance metric is $(\text{Accuracy} + \text{AUC})/2$. Other details can be found in Section 4.4. These results can be seen in the left column of Table 4.

Finetuning. For finetuning, we use our unbiased gradient approximation algorithm with a chunk size of 256. We freeze the pretrained MIL head and only finetune the backbone resnet model. Therefore, we sequentially process each 256 chunk for each input set until the entire set has been processed. We train for 10 total epochs, and use the AdamW optimizer with a learning rate of $5 \cdot 10^{-5}$, and a weight decay of $1 \cdot 10^{-2}$ which is not applied to bias or layernorm parameters. We use a one epoch linear warmup, and then a cosine annealing learning rate decay at every iteration which reaches a minimum at $5 \cdot 10^{-6}$. We train on 1 GPU, with a batch size of 1 and with a single instance on each GPU.

Table 14. Statistics for the Camelyon16 training and test sets we used. **Left**: Number of patches (set size) per instance. **Right**: The distribution of positive and negative samples.

Metric	Train	Test
Mean	9,329	9,376
Min	154	1558
Max	32,382	37,345

Metric	Train	Test
Positive (+)	110	49
Negative (−)	160	80

I. Generalizing Attention Activations

As shown in Equation (7), any attention activation function which can be expressed as a strictly positive elementwise function combined with sum decomposable normalization constants ν_p and $\bar{f}_\theta$ represents a valid attention activation function. Table 15 shows 5 such functions with their respective normalization constants, although there are an infinite number of possible functions which can be used.

The softmax operation we propose $h_1 : \mathbb{R}^d \mapsto (0,1)^d$ which is outlined immediately before Theorem 3.4 is mathematically equivalent to the standard softmax $h_2 : \mathbb{R}^d \mapsto (0,1)^d$ which is commonly implemented in deep learning libraries because f and g have the same domain, the same codomain, and $h_1(\mathbf{x}) = h_2(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^d$. Therefore the functions are mathematically equivalent, even though the implementations are not. Our proposed function h_1 requires separately applying the exponential, and storing and updating the normalization constant while h_2 is generally implemented in such a way that everything is done in a single operation.

Table 15. Valid UMBC attention activation functions with slot normalization ν_1 and normalization over the set elements $\bar{f}_\theta$. Slot-exp uses $\hat{A}_{i,j} - \max_{1 \le i \le k} \hat{A}_{i,j}$ instead of ν_p.

function σ	ν_p	$\bar{f}_\theta$	name	reference
`sigmoid`	$p = 1$	-	slot-sigmoid	(Bruno et al., 2021)
exp()	$p = 1$	✓	slot-softmax	(Locatello et al., 2020)
exp()	$p = 2$	✓	softmax	(Lee et al., 2019)
exp()	-[2]	✓	slot-exp	-
`sigmoid`()	$p = 2$	✓	sigmoid	-

J. Algorithm

We outline our unbiased full set gradient approximation here.

Algorithm 1 Unbiased Full Set Gradient Estimation

```
**Input**: Dataset $((X_i, y_i))_{i=1}^n$, batch size $m$, the number of subsets $m'$, learning rate $(\eta_t)_{t=1}^T$, total steps $T$, and functions $f_\theta$, and $g_\lambda$.
Randomly initialize $\theta$ and $\lambda$
**for all** $t = 1, \dots, T$ **do**
    Sample $((\bar{X}_i, \bar{y}_i))_{i=1}^m \sim D[((X_i, y_i))_{i=1}^n]$
    $L_{t,1}(\theta, \lambda) \leftarrow 0, L_{t,2}(\theta, \lambda) \leftarrow 0$
    **for all** $i = 1 \dots, m$ **do**
        Partition a set $\bar{X}_i$ to get $\zeta_t(\bar{X}_i)$
        Sample $\bar{\zeta}_t(\bar{X}_i) \sim D[\zeta_t(\bar{X}_i)]$ with $|\bar{\zeta}_t(\bar{X}_i)| = m'$
        $f_\theta(\bar{X}_i) = \sum_{\bar{S} \in \bar{\zeta}_t(\bar{X}_i)} f_\theta(\bar{S}) + \sum_{S \in \zeta_t(\bar{X}_i) \setminus \bar{\zeta}_t(\bar{X}_i)} \texttt{StopGrad}(f_\theta(S))$
        $L_{t,1}(\theta, \lambda) \leftarrow L_{t,1}(\theta, \lambda) + \frac{1}{m} \frac{|\zeta_t(\bar{X}_i)|}{|\bar{\zeta}_t(\bar{X}_i)|} \ell(g_\lambda(f_\theta(\bar{X}_i)), \bar{y}_i)$
        $L_{t,2}(\theta, \lambda) \leftarrow L_{t,2}(\theta, \lambda) + \frac{1}{m} \frac{1}{|\bar{\zeta}_t(\bar{X}_i)|} \ell(g_\lambda(f_\theta(\bar{X}_i)), \bar{y}_i)$
    **end for**
    $\theta \leftarrow \theta - \eta_t \frac{\partial L_{t,1}(\theta, \lambda)}{\partial \theta}$
    $\lambda \leftarrow \lambda - \eta_t \frac{\partial L_{t,2}(\theta_t, \lambda)}{\partial \lambda}$
**end for**
```