# ADAPTIVE AGGREGATION FOR PARTITIONED TABULAR IN-CONTEXT LEARNING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Tabular foundation models perform in-context prediction by conditioning directly on labeled examples, but scaling them to large contexts is constrained by computational and memory requirements. This work studies whether full-context predictions of frozen tabular foundation models can be approximated from multiple shards, where the context is either partitioned to satisfy capacity constraints or naturally distributed across sources. We develop a theoretical framework that characterizes when such reconstruction is possible and how reconstruction error depends on partition structure and aggregation weights, motivating a simple training-free, query-adaptive pooling strategy. Experiments on real-world tabular classification and regression tasks examine when shard-level aggregation provides a useful approximation to full-context prediction and how its behavior varies across partitioning regimes. Together, these results provide a general perspective on reconstructing full-context predictions from partitioned contexts in tabular in-context learning.

## 1 INTRODUCTION

Tabular foundation models such as TabPFN (Hollmann et al., 2023; 2025) and TabICL (Qu et al., 2025; 2026) perform prediction by conditioning a frozen model on labeled examples provided at inference time. This in-context learning paradigm avoids fitting a new model for each downstream dataset and has achieved strong performance on tabular prediction tasks. It also shifts part of the computational burden to inference, since the available training examples must be processed as context. In practice, the context may already be partitioned across multiple machines, or it may have to be partitioned when the full dataset cannot be processed in a single model call.

In either case, the frozen model can be evaluated independently on each shard, producing a set of shard-level predictions that must then be combined. The main difficulty is that predictions obtained from partial contexts do not generally combine by simple averaging to reproduce the prediction obtained from the full context. This raises the central questions of this work: Can a full-context prediction be approximated from shard-level predictions, and can the distributed method bypass the context-length limit to achieve comparable or even better performance?

We study this problem through convex reconstruction. We introduce the convex decomposition deficit to quantify the discrepancy between the full-context prediction and its best convex reconstruction from shard-level predictions. For in-context predictors represented as a smooth readout of a context-dependent representation, our analysis separates the approximation error into two sources: the error of reconstructing the full-context representation from shard representations, and the dispersion of those representations under the nonlinear readout. This characterization provides a theoretical basis for query-dependent aggregation.

Guided by this analysis, we propose Adaptive Mass Pooling (AMP), a training-free aggregation rule for partitioned tabular in-context learning. For each query, AMP measures the local mass of relevant training examples stored in each shard and uses it to weight the corresponding shard-level prediction. The foundation model remains frozen, and aggregation is performed entirely at inference time. Experiments on synthetic and real tabular tasks study how well shard-level aggregation approximates full-context prediction across different foundation models, partitioning settings, and aggregation regimes.



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

## 2 Related Work

Prior work on scaling tabular in-context learning has focused on constructing informative contexts under a fixed context budget. LoCalPFN (Thomas et al., 2024) combines nearest-neighbor retrieval with task-specific fine-tuning, while the MICP component of MixturePFN (Xu et al., 2025) routes queries to cluster-specific support sets. CRUMB (Heredge et al., 2026) clusters unlabeled test queries and selects a distribution-matched training subset for each cluster, enabling shared-context inference without retraining. Together, these methods establish context construction as a central mechanism for scaling tabular foundation models under bounded inference capacity. Our work considers the complementary setting in which data are represented by multiple disjoint, capacity-limited shards. The frozen model is evaluated on each shard, and shard-level predictions are aggregated to approximate the prediction obtained from the full dataset.

Combining predictive distributions through convex mixtures is classical linear pooling (Gneiting & Ranjan, 2013), and a closely related structure also appears in language-model in-context learning. DENSE (Khalifa et al., 2023) partitions demonstrations into multiple subsets, evaluates each subset separately, and combines the resulting output probabilities using several ensemble rules. Its weighted variants assign query-dependent importance to each subset according to the average embedding similarity between the query and its demonstrations. These methods provide a direct precedent for prediction-level aggregation across multiple bounded contexts. Building on this perspective, we formulate aggregation with respect to a full-context reference prediction and study the extent to which that prediction can be reconstructed from obtained shard-level predictions.

A further connection arises from classical Nadaraya–Watson kernel regression (Nadaraya, 1964; Watson, 1964), which admits an exact decomposition across data subsets when using a shared nonnegative kernel and bandwidth. The full-sample estimator can be written as a mixture of subset estimators, with each weight determined by the subset's share of total kernel mass. Related work has connected Transformer attention to kernel smoothing, interpreting attention as a data-dependent similarity-weighted aggregation mechanism (Tsai et al., 2019). This view interprets subset relevance in terms of aggregate query-dependent mass. In our setting, this connection motivates using local mass as a query-adaptive signal for weighting shard-level predictions.

## 3 Methodology

### 3.1 Overview

Figure 1 illustrates the overall framework. When a context is too large for a single inference call or is naturally distributed across multiple sources, it can be represented by multiple shards. Each shard is independently processed by a frozen tabular foundation model, and the resulting predictions are aggregated to approximate the full-context prediction.

Our key idea is to measure the query-specific mass of relevant examples contained in each shard, and use this adaptive mass to determine their contributions. We study how such aggregation interacts with different partition structures and affects reconstruction quality.

### 3.2 Theoretical framework

To begin with, we introduce some concepts and related notations. We use $\mathcal{X}$ and $\mathcal{Y}$ to denote the feature space and outcome space, respectively. The prediction function can be formulated as

$$f : \mathcal{X} \times \bigsqcup_{n=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^{(n)} \to \mathcal{P}(\mathcal{Y}). \tag{3.1}$$

Here, $\mathcal{P}(\mathcal{Y})$ is the probability measure on the outcome space $\mathcal{Y}$. To better illustrate, we denote the realization of a query and the training data by $x \in \mathcal{X}$ and $\mathcal{D} := \{(x_i, y_i)\}_{i=1}^{n} \in (\mathcal{X} \times \mathcal{Y})^{(n)}$, respectively.

*Remark* 1. In many applications, the value of the prediction function $f(x, \mathcal{D})$ is a number $y \in \mathbb{R}$, which can also be viewed as a degenerate distribution $\delta_y \in \mathcal{P}(\mathcal{Y})$, where $\delta_y$ is the Dirac measure, defined by $\delta_y(S) := \mathbf{1}_S(y)$ for any $S \subseteq \mathcal{Y}$.

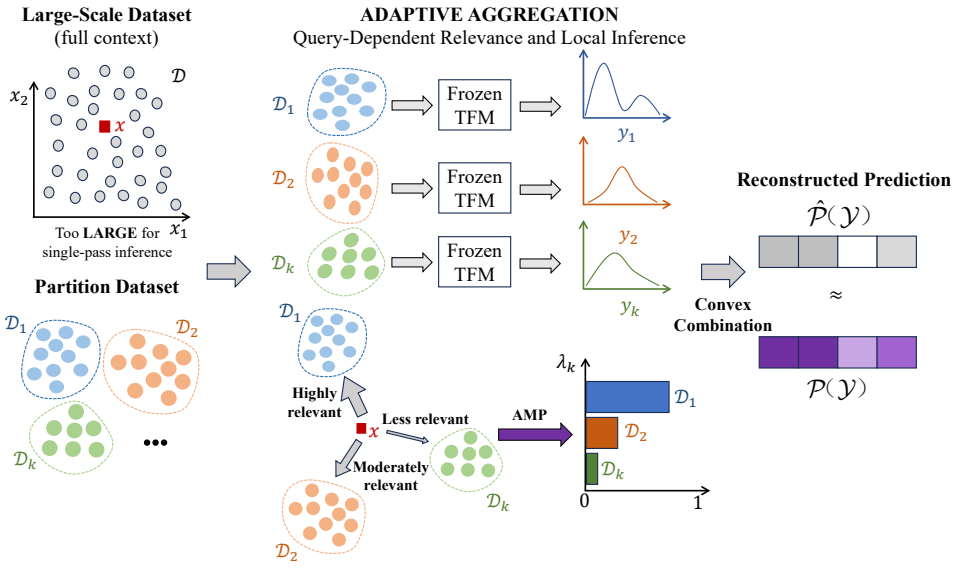



Figure 1: Overview of the framework. When a context is too large for a single inference call or is naturally distributed across multiple sources, it can be represented by multiple shards. Each shard is independently processed by a frozen tabular foundation model, and the resulting predictions are aggregated to approximate the full-context prediction.

Next, we introduce the concepts and notations in distributed learning. Suppose there are $K$ servers, where the $k$-th server stores local data $\mathcal{D}_k$. We use $\mathcal{D}^{(K)} := \{\mathcal{D}_1, \cdots, \mathcal{D}_K\}$ to denote the partition. We still use $\mathcal{D}$, which satisfies

$$\mathcal{D} = \bigsqcup_{k=1}^{K} \mathcal{D}_k, \quad n_k = |\mathcal{D}_k|, \quad n = \sum_{k=1}^{K} n_k.$$

For an input query $x$, the model based on the full dataset $\mathcal{D}$ produces an oracle prediction $f(x, \mathcal{D})$, and each local model based on the partial dataset $\mathcal{D}^{(K)}$ produces a local prediction $f(x, \mathcal{D}_k)$ for $k = 1, \ldots, K$. Our goal is to recover $f(x, \mathcal{D})$ from the local predictions $f(x, \mathcal{D}^{(K)}) := \{f(x, \mathcal{D}_k)\}_{k=1}^{K}$. A natural approach is using a convex combination of local predictions $f(x, \mathcal{D}^{(K)})$, which generalizes the simple equal-weight average. However, it's too ideal to assume all prediction functions $f(x, \mathcal{D})$ can be exactly expressed as a convex combination of $f(x, \mathcal{D}^{(K)})$. To capture the approximate error, we introduce the following concepts. We let $\Delta^{K-1} := \{\lambda \in \mathbb{R}^K : \sum_{k=1}^{K} \lambda_k = 1,\ \lambda_k \geq 0\}$ denote the set of convex weights.

**Definition 1.** For $\mathcal{D}^{(K)} = \{\mathcal{D}_1, \cdots, \mathcal{D}_K\}$, we define the convex combination of $f(x, \mathcal{D}^{(K)})$ with $\lambda(x) \in \Delta^{K-1}$, which is

$$f_\lambda(x, \mathcal{D}^{(K)}) := \sum_{k=1}^{K} \lambda_k(x) f(x, \mathcal{D}_k).$$

**Definition 2.** We define

$$\mathcal{E}_f^{(\lambda)}(x, \mathcal{D}^{(K)}) := \left\| f(x, \mathcal{D}) - f_\lambda(x, \mathcal{D}^{(K)}) \right\|, \quad \mathcal{E}_f(x, \mathcal{D}^{(K)}) := \min_{\lambda(x) \in \Delta^{K-1}} \mathcal{E}_f^{(\lambda)}(x, \mathcal{D}^{(K)}).$$

Here, we refer to $\mathcal{E}_f^{(\lambda)}(x, \mathcal{D}^{(K)})$ and $\mathcal{E}_f(x, \mathcal{D}^{(K)})$ as the *convex decomposition deficit* of $f$ with respect to the partition $\mathcal{D}^{(K)}$. If $\mathcal{E}_f(x, \mathcal{D}^{(K)}) \equiv 0$, we call $f$ *convex decomposable*.

*Remark* 2. Definition 2 is inspired by the Nadaraya–Watson estimator, defined as $f(x, \mathcal{D}) := \left(\sum_{(x_i, y_i) \in \mathcal{D}} \mathcal{K}(x, x_i)\right)^{-1} \sum_{(x_i, y_i) \in \mathcal{D}} \mathcal{K}(x, x_i) y_i$. We have $f(x, \mathcal{D}) = f_\lambda(x, \mathcal{D}^{(K)})$ with $\lambda_k(x) :=$

$\left(\sum_{(x_i,y_i)\in\mathcal{D}} \mathcal{K}(x,x_i)\right)^{-1} \sum_{(x_i,y_i)\in\mathcal{D}_k} \mathcal{K}(x,x_i)$, which implies the corresponding $\mathcal{E}_f(x,\mathcal{D}^{(K)}) = 0$. Thus, the Nadaraya–Watson estimator is convex decomposable.

For the in-context based model, there seems to exist a form similar to the Nadaraya–Watson estimator. Let $\psi : \mathcal{X}\times\mathcal{Y} \to \mathcal{U}$ be an embedding map, where $\mathcal{U}$ is a Hilbert space. Attention constructs complicated relations from embedding vectors, denoted by $a_{\mathcal{D}}(x,(x_i,y_i))$. Since the softmax function yields a convex combination, we have $(a_{\mathcal{D}}(x,(x_1,y_1)),\cdots,a_{\mathcal{D}}(x,(x_n,y_n))) \in \Delta^{n-1}$. Let $\delta_x$ be the Dirac measure, defined by $\delta_x(S) := \mathbf{1}_S(x)$ for any $S \subseteq \mathcal{X}$. Next, we define the discrete measurement based on these relationships

$$\nu_{\mathcal{D}}(x,\cdot) := \sum_{(x_i,y_i)\in\mathcal{D}} a_{\mathcal{D}}(x,(x_i,y_i))\delta_{(x_i,y_i)}. \tag{3.2}$$

The embedding of query $x$ can be formulated as $u(\cdot,\mathcal{D}) : \mathcal{X} \to \mathcal{U}$, which satisfies

$$u(x,\mathcal{D}) := \int_{\mathcal{X}\times\mathcal{Y}} \psi(x_i,y_i)\nu_{\mathcal{D}}(x,d(x_i,y_i)). \tag{3.3}$$

After embedding, there is a Fréchet differentiable function $\mathcal{F} : \mathcal{U} \to \mathcal{P}(\mathcal{Y})$ to transfer the embedding vector to a distribution $f(x,\mathcal{D}) := \mathcal{F}(u(x,\mathcal{D}))$.

**Assumption 1.** We assume that the Fréchet derivative $D_u\mathcal{F}$ satisfies

$$\|D_u\mathcal{F}(u)\| \le B,$$
$$\|D_u\mathcal{F}(u) - D_u\mathcal{F}(v)\| \le L\|u-v\|.$$

*Remark* 3. In tabular foundation models, $\mathcal{F}(u) = \sigma(Wu)$, where $W$ is a linear function and $\sigma$ is the softmax function. Therefore, the Fréchet derivative $D_u\mathcal{F}$ degenerates to the Jacobian matrix $\partial\mathcal{F}/\partial u = (\mathrm{diag}(\sigma) - \sigma\sigma^\top)W$, which naturally satisfies Assumption 1.

Now, we study the properties when conducting distributed learning on $f(x,\mathcal{D}) := \mathcal{F}(u(x,\mathcal{D}))$. Let $\mathcal{V}(x,\mathcal{D}^{(K)}) := \{\mathcal{V}(x,\mathcal{D}_k)\}_{k=1}^K$, where $\mathcal{V}(x,\mathcal{D}_k) = \|u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\|^2$. From Definition 1, we define $\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}) := \sum_{k=1}^K \lambda_k\mathcal{V}(x,\mathcal{D}_k)$ as the convex combination of $\mathcal{V}(x,\mathcal{D}^{(K)})$. [1] For the embedding $u$, we define $\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)})$ and $\mathcal{E}_u(x,\mathcal{D}^{(K)})$ analogously to Definition 2.

**Theorem 1.** *Under Assumption 1, for given $\lambda(x)$, we have*

$$\mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)}) \le B\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}) + \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}).$$

From Theorem 1, the error bound $\mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)})$ can be decomposed into two parts. The first part $\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)})$ relates to the approximation error from convex decomposition. The second part stems from the nonlinearity of $\mathcal{F}$ and is controlled by the "pseudo variance term" $\mathcal{V}_\lambda(x,\mathcal{D}^{(K)})$, which is the convex combination of residual $\mathcal{V}(x,\mathcal{D}^{(K)})$. When $u$ is convex decomposable, the oracle approximation error of $u$ is reduced, yielding Corollary 1.

**Corollary 1.** *If $u$ is convex decomposable, we have*

$$\mathcal{E}_f(x,\mathcal{D}^{(K)}) \le \max_{\lambda\in\Delta^{K-1}} \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}).$$

In Theorem 1, we transfer the approximation error of $f$ to that of $u$. However, the approximation error of $u$ is still unclear. Therefore, we study the property of $\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)})$.

**Theorem 2.** *Let $\psi_{\max} := \max_{(x,y)\in\mathcal{D}} \psi(x,y)$. We have*

$$\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}) \le \sum_{k=1}^K \sum_{(x_i,y_i)\in\mathcal{D}_k} \|a_{\mathcal{D}}(x,(x_i,y_i)) - \lambda_k(x)a_{\mathcal{D}_k}(x,(x_i,y_i))\|\,\|\psi_{\max}\|.$$

From Theorem 2, the convex decomposable deficit of $u$ is controlled by the difference in similarity between the full context $a_{\mathcal{D}}$ and the partial context $a_{\mathcal{D}_k}$. Taking the Nadaraya-Watson estimator as an example, there exists a $\lambda_k(x)$ that matches $a_{\mathcal{D}}$ with each partial context $a_{\mathcal{D}_k}$. Intuitively, the corresponding convex coefficient $\lambda(x)$ should have a form similar to $a_{\mathcal{D}}(x,(x_i,y_i))/a_{\mathcal{D}_k}(x,(x_i,y_i))$. However, the similarity between $a_{\mathcal{D}}(x,(x_i,y_i))$ and $a_{\mathcal{D}_k}(x,(x_i,y_i))$ is unknown, which motivates us to use a "proxy similarity measure" to construct a weight.

[1] Indeed, $\mathcal{V}(x,\mathcal{D}^{(K)})$ depends on $\lambda$. For consistency with Definition 1, we omit the subscript $\lambda$.

**Algorithm 1** Adaptive Mass Pooling (AMP)

**Require:** Partition $\mathcal{D}^{(K)}$; query $x_0$; hyperparameters $T, \alpha$; local predictors $f(\cdot, \mathcal{D}_k)$.
**Ensure:** Aggregated prediction $f_\lambda(x_0, \mathcal{D}^{(K)})$.

$d_T(x_0) \leftarrow$ the $T$-th smallest distance among $\{\|x_i - x_0\|\}_{i=1}^n$ ▷ similarity threshold
**for** $k = 1$ **to** $K$ **do**
  $n_k \leftarrow |\{(x, y) \in \mathcal{D}_k : \|x - x_0\| \le d_T(x_0)\}|$ ▷ local mass
  $f_k \leftarrow f(x_0, \mathcal{D}_k)$ ▷ local prediction
**end for**
$w_k \leftarrow n_k/n, \quad k = 1, \dots, K$
$\lambda_k \leftarrow \dfrac{w_k^\alpha}{\sum_{j=1}^K w_j^\alpha}, \quad k = 1, \dots, K$ ▷ normalized weights
**return** $\sum_{k=1}^K \lambda_k f_k$

### 3.3 Adaptive Aggregation

To match $a_{\mathcal{D}}(x, (x_i, y_i))/a_{\mathcal{D}_k}(x, (x_i, y_i))$, a proxy criterion is the ratio of the cardinalities of the subsets of $\mathcal{D}$ and $\mathcal{D}_k$ satisfying a given criterion. Specifically, the partition of $\mathcal{D}$ is denoted as $\mathcal{D}^{(K)} := \{\mathcal{D}_1, \cdots, \mathcal{D}_K\}$, where $\mathcal{D}_k := \{(x_i^{(k)}, y_i^{(k)})\}_{i=1}^{n_k}$. For any query $x_0$, the adaptive mass pooling strategy consists of three steps:

#### Step 1: Compute the threshold of similar level

Take the hyperparameter $T$. We compute

$$d_T(x_0) := \inf\{d \ge 0 : |\mathcal{B}(x_0, d) \cap \mathcal{D}| \ge T\}, \quad \mathcal{B}(x_0, d) := \{x \in \mathcal{X} : \|x - x_0\| \le d\}.$$

#### Step 2: Compute the similar member

Calculate $w_k(x_0, T) = n_k(x_0, T)/n$, where

$$n_k(x_0, T) := |\{(x, y) \in \mathcal{D}_k : \|x - x_0\| \le d_T(x_0)\}|.$$

#### Step 3: Aggregating the results

Take hyperparameter $\alpha$. Compute $\lambda_k(x_0, T) := w_k^\alpha(x_0, T)/\sum_{k=1}^K w_k^\alpha(x_0, T)$. Get the final result with

$$f_\lambda(x_0, \mathcal{D}^{(K)}) := \sum_{k=1}^K \lambda_k(x_0, T) f(x_0, \mathcal{D}_k). \tag{3.4}$$

These steps are summarized in Algorithm 1.

AMP recovers familiar aggregation rules under particular choices of its parameters and shards. If $T = n$ and $\alpha = 1$, then $n_k(x_0, T) = n_k$ and $\lambda_k(x_0, T) = n_k/n$, reducing AMP to size-weighted (FedAvg-style) prediction pooling. Moreover, if every shard contains a single example and its local prediction is the corresponding label distribution, then, for $\alpha = 1$, AMP assigns equal weight to the $T$ nearest examples and reduces to unweighted $T$-nearest-neighbor prediction.

## 4 Experiment

In this section, we study how aggregation weights affect the reconstruction of full-context predictions from shard-level predictions. Our theoretical analysis characterizes the role of locality in context construction, a principle also used by methods such as LoCalPFN. We therefore fix the partitioning mechanism and focus on the complementary problem of weight selection. Specifically, the training set is randomly partitioned into capacity-constrained shards, and all aggregation rules are evaluated on the same partition. This controlled setting isolates the effect of aggregation weights on reconstruction quality.

A central question is how predictions from different shards should be weighted. Uniform averaging assigns equal mass to every shard, $\lambda_k(x) = 1/K$, regardless of its size. A stronger query-independent baseline assigns weight in proportion to shard size, $\lambda_k(x) = n_k/n$. This is the weighting principle underlying FedAvg (McMahan et al., 2017; Li et al., 2020). Because $\mathcal{D} = \bigsqcup_{k=1}^{K} \mathcal{D}_k$ and $n = \sum_{k=1}^{K} n_k$, it assigns the same base mass $1/n$ to every training example. In distributed empirical-risk minimization, these weights recover the pooled empirical objective from the local objectives. We use this principle as a query-independent reference and call it size-weighted (FedAvg-style) pooling.

AMP instead uses the local mass $n_k(x_0, T)$ around the query, as defined in Section 3.3. The three rules thus assign equal mass across shards, equal mass across samples, or query-dependent mass across shards. To assess how much reconstruction quality remains attainable through weight selection, we also construct an oracle. For each query and fixed partition, it optimizes $\lambda(x) \in \Delta^{K-1}$ against the full-context predictive distribution $f(x, \mathcal{D})$ while keeping all shard predictions fixed. The oracle minimizes Total Variation (TV) distance. We evaluate its resulting prediction using both TV and CDF-MSE; it is a lower bound for TV reconstruction over convex weights, but is not separately optimized for CDF-MSE.

Table 1 describes the datasets. Tables 2 reports means and standard deviations over five random seeds. Within each seed and partition setting, all weighting rules use the same data split and cached shard predictions. We report TV and CDF-MSE for reconstruction fidelity. For downstream performance, we use accuracy and log loss for classification, and RMSE and MAE for regression. Comparisons between weighting rules are always made within the same base model and partition.

Table 2 shows that query-dependent weighting usually improves reconstruction over the size-weighted reference. AMP has lower mean TV than FedAvg-style pooling in 14 of the 18 dataset–model rows, and lower mean CDF-MSE in 14 of 18 rows. The gains appear with both TPF-v3 and TIC-v2. For example, on optdigits, AMP reduces TV from $8.16$ to $7.22$ for TPF-v3 and from $5.80$ to $5.07$ for TIC-v2, in the table's $\times 10^{-2}$ units. Equal weighting generally gives larger reconstruction errors, indicating that shard size remains useful even when no query-dependent weighting is available.

$$\frac{n_k}{n} \qquad \text{for every } k.$$

If the shards have equal sizes and receive equal query-local mass, both rules further reduce to uniform averaging. Random sharding can approach this balanced regime, leaving little useful variation for AMP to exploit. The table also contains cases in which AMP is worse than size weighting, including pima with TPF-v3 and transfusion with TIC-v2. The overall pattern is therefore a frequent advantage for query-dependent weights, with a clear limit when the partition carries little differential local information.

The oracle quantifies a separate source of headroom. On spambase with TPF-v3, AMP attains a TV error of $5.67$, whereas the oracle reaches .36; on optdigits, the corresponding values are $7.22$ and .76. These gaps show that the fixed shard predictions can sometimes reconstruct the full-context prediction more closely under query-specific convex weights. Because the oracle uses the full-context prediction as its target, it is a diagnostic reference rather than a deployable aggregation rule.

Reconstruction fidelity does not fully determine predictive performance against the observed labels. Table 3 shows several cases in which predicting on shards and then aggregating exceeds prediction with the full context. On pima, AMP raises TPF-v3 accuracy from .760 to .786 and TIC-v2 accuracy from .786 to .805. Its log loss is also lower than full-context log loss for both models. On transfusion with TPF-v3, equal weighting raises accuracy from .753 to .767, although its log loss is slightly higher than the full-context value. Thus, an improvement in one downstream metric need not extend to every metric.

These gains are specific rather than universal. Full-context prediction remains better than all three aggregation rules on RMSE and MAE for each regression dataset in Table 3. The reconstruction and downstream comparisons answer different questions: the former asks whether the aggregate matches the full-context predictive distribution, whereas the latter asks how well it predicts the label. Consequently, the full-context prediction is a meaningful reconstruction target but is not an upper bound on downstream performance. The present experiment establishes that gains after sharding

Table 1: Datasets used in the experiments. Every dataset is run with both TabPFN-v3 and TabIC-v2.

| Dataset | Task | #Feat. | $N_{\text{train}}/N_{\text{test}}$ |
|---|---|---|---|
| banknote | classification | 4 | 1097/275 |
| spambase | classification | 57 | 3680/921 |
| optdigits | classification | 64 | 4496/1124 |
| concrete | regression | 8 | 824/206 |
| transfusion | classification | 4 | 598/150 |
| pima | classification | 8 | 614/154 |
| german-credit | classification | 74 | 800/200 |
| fish-toxicity | regression | 6 | 726/182 |
| wine (red) | regression | 11 | 1279/320 |

Table 2: Reconstruction distance to same-model full-context predictions under Random sharding (lower is better). TPF-v3 denotes TabPFN-v3 and TIC-v2 denotes TabICL-v2. TV is reported $\times 10^{-2}$ and CDF-MSE $\times 10^{-3}$. Bold marks the best deployable aggregator; Oracle is a diagnostic lower bound.

| | | Reconstruction TV ↓ ($\times 10^{-2}$) | | | | CDF-MSE ↓ ($\times 10^{-3}$) | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | Model | FedAvg | AMP | Equal | *Oracle* | FedAvg | AMP | Equal | *Oracle* |
| banknote | TPF-v3 | $.16 \pm .16$ | $\mathbf{.14} \pm .15$ | $.81 \pm 1.02$ | $.00 \pm .00$ | $\mathbf{.112} \pm .211$ | $.117 \pm .231$ | $1.035 \pm 1.402$ | $.000 \pm .000$ |
| | TIC-v2 | $.19 \pm .16$ | $\mathbf{.15} \pm .14$ | $.90 \pm 1.00$ | $.00 \pm .00$ | $\mathbf{.078} \pm .115$ | $.083 \pm .148$ | $1.081 \pm 1.291$ | $.000 \pm .000$ |
| spambase | TPF-v3 | $5.87 \pm .30$ | $\mathbf{5.67} \pm .33$ | $6.54 \pm .73$ | $.36 \pm .05$ | $14.26 \pm 1.36$ | $\mathbf{13.57} \pm 1.35$ | $15.18 \pm 1.31$ | $.69 \pm .18$ |
| | TIC-v2 | $5.99 \pm .26$ | $\mathbf{5.80} \pm .29$ | $6.61 \pm .69$ | $.29 \pm .09$ | $16.06 \pm 1.73$ | $\mathbf{15.29} \pm 1.78$ | $17.07 \pm 1.92$ | $.62 \pm .32$ |
| optdigits | TPF-v3 | $8.16 \pm .45$ | $\mathbf{7.22} \pm .36$ | $10.45 \pm 3.26$ | $.76 \pm .07$ | $8.06 \pm .33$ | $\mathbf{6.99} \pm .28$ | $9.18 \pm 1.71$ | $.36 \pm .07$ |
| | TIC-v2 | $5.80 \pm .29$ | $\mathbf{5.07} \pm .21$ | $7.64 \pm 2.64$ | $.22 \pm .04$ | $6.25 \pm .25$ | $\mathbf{5.49} \pm .29$ | $7.02 \pm 1.15$ | $.12 \pm .07$ |
| concrete | TPF-v3 | $47.28 \pm .64$ | $\mathbf{46.94} \pm .56$ | $50.66 \pm 5.50$ | $34.72 \pm .57$ | $15.78 \pm .13$ | $\mathbf{15.68} \pm .01$ | $19.33 \pm 5.19$ | $4.10 \pm .28$ |
| | TIC-v2 | $41.34 \pm 1.76$ | $\mathbf{40.92} \pm 1.54$ | $44.16 \pm 5.87$ | $26.11 \pm .92$ | $14.59 \pm 1.03$ | $\mathbf{14.39} \pm 1.00$ | $17.35 \pm 5.31$ | $2.24 \pm .07$ |
| transfusion | TPF-v3 | $2.67 \pm .12$ | $\mathbf{2.59} \pm .12$ | $2.65 \pm .62$ | $.35 \pm .05$ | $1.20 \pm .11$ | $\mathbf{1.18} \pm .08$ | $1.20 \pm .20$ | $.12 \pm .15$ |
| | TIC-v2 | $\mathbf{1.45} \pm .06$ | $1.53 \pm .07$ | $1.50 \pm .43$ | $.03 \pm .04$ | $\mathbf{.44} \pm .02$ | $.52 \pm .03$ | $.47 \pm .08$ | $.01 \pm .06$ |
| pima | TPF-v3 | $\mathbf{3.75} \pm .17$ | $3.89 \pm .18$ | $3.77 \pm .87$ | $.35 \pm .09$ | $\mathbf{3.02} \pm .21$ | $3.04 \pm .30$ | $3.06 \pm .30$ | $.40 \pm .10$ |
| | TIC-v2 | $\mathbf{3.12} \pm .17$ | $3.22 \pm .14$ | $3.16 \pm .43$ | $.24 \pm .07$ | $1.99 \pm .15$ | $\mathbf{1.96} \pm .11$ | $2.03 \pm .21$ | $.13 \pm .06$ |
| german-credit | TPF-v3 | $6.28 \pm .29$ | $\mathbf{6.02} \pm .29$ | $6.50 \pm 2.11$ | $.49 \pm .07$ | $6.60 \pm .27$ | $\mathbf{6.23} \pm .35$ | $7.06 \pm 1.00$ | $.28 \pm .13$ |
| | TIC-v2 | $5.95 \pm .30$ | $\mathbf{5.79} \pm .26$ | $6.07 \pm 1.20$ | $.24 \pm .06$ | $5.66 \pm .54$ | $\mathbf{5.52} \pm .41$ | $5.93 \pm .86$ | $.09 \pm .03$ |
| fish-toxicity | TPF-v3 | $22.58 \pm .35$ | $\mathbf{22.40} \pm .74$ | $22.59 \pm 2.73$ | $19.31 \pm .68$ | $5.16 \pm .18$ | $\mathbf{4.85} \pm .10$ | $5.16 \pm 1.54$ | $1.96 \pm .08$ |
| | TIC-v2 | $13.25 \pm .36$ | $\mathbf{13.07} \pm .16$ | $13.26 \pm 1.73$ | $8.40 \pm .23$ | $3.49 \pm .13$ | $\mathbf{3.32} \pm .17$ | $3.50 \pm 1.07$ | $1.12 \pm .05$ |
| wine (red) | TPF-v3 | $33.31 \pm .89$ | $\mathbf{33.09} \pm 1.40$ | $33.31 \pm 4.10$ | $24.98 \pm .54$ | $17.85 \pm .38$ | $\mathbf{17.12} \pm .39$ | $17.85 \pm 5.23$ | $.76 \pm .05$ |
| | TIC-v2 | $49.31 \pm 1.26$ | $49.33 \pm .77$ | $\mathbf{49.31} \pm 5.75$ | $44.95 \pm 1.39$ | $21.31 \pm .32$ | $\mathbf{20.91} \pm 1.38$ | $21.31 \pm 6.46$ | $8.81 \pm .43$ |

occur; it does not isolate whether they arise from the changed contexts, the aggregation weights, or their interaction.

## 5 Discussion

AMP improves reconstruction over size-weighted pooling in most dataset–model settings, suggesting that query-local mass is often more informative than shard size alone. However, the gains vary across datasets, models, and metrics, and AMP does not uniformly dominate query-independent aggregation. The oracle results reveal a substantial gap between AMP and the best convex reconstruction, indicating that more accurate weight estimation could yield further improvements. Meanwhile, nonzero oracle errors, particularly in regression, show that convex pooling cannot recover interactions that arise only when examples from different shards are jointly processed.

Reconstruction fidelity and downstream performance should also be distinguished. Since the full-context output is a model prediction rather than ground truth, closer reconstruction does not necessarily imply better predictive performance. Aggregation occasionally matches or improves full-context classification accuracy, potentially due to an ensemble-like regularization effect, but full-context inference remains stronger on several classification and regression tasks. Thus, the results support AMP as a useful approximation under context partitioning rather than a general replacement for full-context inference.

Table 3: Downstream performance of shard-level aggregation compared with same-model full-context prediction. Mean ± std over multiple seeds; bold indicates cases where aggregation exceeds full-context performance.

| | | Accuracy ↑ | | | | Log loss ↓ ($\times 10^{-3}$) | | | |
|---|---|---|---|---|---|---|---|---|---|
| Dataset | Model | Full | FedAvg | AMP | Equal | Full | FedAvg | AMP | Equal |
| *(a) Classification* | | | | | | | | | |
| banknote | TPF-v3 | $1.000 \pm .000$ | $1.000 \pm .000$ | $1.000 \pm .000$ | $1.000 \pm .000$ | $.0 \pm .0$ | $1.7 \pm 1.7$ | $1.4 \pm 1.6$ | $8.8 \pm 10.8$ |
| | TIC-v2 | $1.000 \pm .000$ | $1.000 \pm .000$ | $1.000 \pm .000$ | $1.000 \pm .000$ | $.0 \pm .0$ | $1.9 \pm 1.7$ | $1.5 \pm 1.5$ | $9.7 \pm 10.6$ |
| spambase | TPF-v3 | $.967 \pm .003$ | $.947 \pm .007$ | $.949 \pm .007$ | $.947 \pm .007$ | $103.2 \pm 9.4$ | $156.5 \pm 5.9$ | $153.6 \pm 6.2$ | $162.7 \pm 9.5$ |
| | TIC-v2 | $.967 \pm .002$ | $.949 \pm .008$ | $.949 \pm .008$ | $.948 \pm .008$ | $96.1 \pm 6.8$ | $149.3 \pm 6.8$ | $146.6 \pm 7.6$ | $155.6 \pm 8.5$ |
| optdigits | TPF-v3 | $.993 \pm .001$ | $.965 \pm .001$ | $.967 \pm .001$ | $.965 \pm .001$ | $32.0 \pm .5$ | $134.5 \pm 5.9$ | $122.1 \pm 4.9$ | $160.8 \pm 38.1$ |
| | TIC-v2 | $.993 \pm .001$ | $.975 \pm .001$ | $.976 \pm .002$ | $.974 \pm .001$ | $26.5 \pm 1.0$ | $97.0 \pm 4.3$ | $88.0 \pm 3.8$ | $117.3 \pm 30.3$ |
| transfusion | TPF-v3 | $.753 \pm .035$ | $\mathbf{.760} \pm .035^{\dagger\ddagger}$ | $\mathbf{.760} \pm .035^{\dagger\ddagger}$ | $\mathbf{.767} \pm .035^{\dagger\ddagger}$ | $521.2 \pm 39$ | $521.9 \pm 21$ | $522.1 \pm 21$ | $522.2 \pm 25$ |
| | TIC-v2 | $.747 \pm .034$ | $.747 \pm .034$ | $.747 \pm .034$ | $.747 \pm .034$ | $524.1 \pm 13$ | $526.4 \pm 27$ | $528.1 \pm 24$ | $526.8 \pm 22$ |
| pima | TPF-v3 | $.760 \pm .033$ | $\mathbf{.779} \pm .033^{\dagger\ddagger}$ | $\mathbf{.786} \pm .033^{\dagger\ddagger}$ | $\mathbf{.779} \pm .033^{\dagger\ddagger}$ | $454.6 \pm 37$ | $\mathbf{447.0} \pm 19^{\dagger\ddagger}$ | $\mathbf{448.5} \pm 20^{\dagger\ddagger}$ | $\mathbf{447.2} \pm 23^{\dagger\ddagger}$ |
| | TIC-v2 | $.786 \pm .033$ | $\mathbf{.792} \pm .033^{\dagger\ddagger}$ | $\mathbf{.805} \pm .033^{\dagger\ddagger}$ | $\mathbf{.792} \pm .033^{\dagger\ddagger}$ | $451.0 \pm 8$ | $\mathbf{447.6} \pm 19^{\dagger\ddagger}$ | $\mathbf{449.4} \pm 21^{\dagger\ddagger}$ | $\mathbf{447.7} \pm 22^{\dagger\ddagger}$ |
| german-credit | TPF-v3 | $.780 \pm .030$ | $.755 \pm .030$ | $.765 \pm .030$ | $.750 \pm .030$ | $477.5 \pm 14$ | $501.0 \pm 22$ | $500.4 \pm 24$ | $501.9 \pm 24$ |
| | TIC-v2 | $.780 \pm .030$ | $.760 \pm .030$ | $.765 \pm .030$ | $.760 \pm .030$ | $471.3 \pm 28$ | $492.4 \pm 22$ | $490.2 \pm 19$ | $494.0 \pm 24$ |
| *(b) Regression* | | | | | | | | | |
| | | RMSE ↓ | | | | MAE ↓ | | | |
| Dataset | Model | Full | FedAvg | AMP | Equal | Full | FedAvg | AMP | Equal |
| concrete | TPF-v3 | $4.309 \pm .069$ | $5.386 \pm .130$ | $5.401 \pm .174$ | $5.624 \pm .497$ | $2.383 \pm .144$ | $3.869 \pm .192$ | $3.876 \pm .236$ | $4.137 \pm .598$ |
| | TIC-v2 | $4.282 \pm .028$ | $5.366 \pm .262$ | $5.353 \pm .251$ | $5.569 \pm .603$ | $2.337 \pm .160$ | $3.809 \pm .335$ | $3.795 \pm .347$ | $3.996 \pm .631$ |
| fish-toxicity | TPF-v3 | $.9564 \pm .01$ | $.9634 \pm .05$ | $.9625 \pm .03$ | $.9633 \pm .09$ | $.6568 \pm .04$ | $.6760 \pm .04$ | $.6732 \pm .05$ | $.6760 \pm .10$ |
| | TIC-v2 | $.9414 \pm .01$ | $.9661 \pm .04$ | $.9659 \pm .03$ | $.9660 \pm .10$ | $.6472 \pm .04$ | $.6752 \pm .05$ | $.6737 \pm .05$ | $.6751 \pm .11$ |
| wine (red) | TPF-v3 | $.5704 \pm .01$ | $.6145 \pm .02$ | $.6136 \pm .02$ | $.6144 \pm .06$ | $.3862 \pm .03$ | $.4769 \pm .04$ | $.4753 \pm .03$ | $.4769 \pm .07$ |
| | TIC-v2 | $.5823 \pm .01$ | $.6255 \pm .03$ | $.6260 \pm .02$ | $.6254 \pm .07$ | $.4008 \pm .03$ | $.4861 \pm .03$ | $.4857 \pm .03$ | $.4861 \pm .07$ |

## 6 Limitations

AMP relies on Euclidean neighbor counts, whose quality depends on feature scaling, the distance metric, $T$, and $\alpha$. Computing the global nearest-neighbor threshold may also create communication overhead in distributed settings. Finally, our experiments simulate capacity constraints on datasets where full-context inference remains feasible. Future work should evaluate over-capacity datasets, systems costs, alternative similarity measures, and different shard configurations.

## AI Use Statement

During the preparation of this paper, the authors used large language models (LLMs) to assist with research discussions, exploration of related work, experimental design discussions, and manuscript refinement. The core research ideas, theoretical developments, experimental implementations, analyses, and conclusions were developed and verified by the authors.

## References

Tilmann Gneiting and Roopesh Ranjan. Combining predictive distributions. 2013.

Jamie Heredge, Mattia J Villani, Pranav Deshpande, Akshay Seshadri, and Niraj Kumar. Crumb: Efficient prior fitted network inference via distributionally matched context batching. *arXiv preprint arXiv:2606.11473*, 2026.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations*, 2023.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025. doi: 10.1038/s41586-024-08328-6.

Muhammad Khalifa, Lajanugen Logeswaran, Moontae Lee, Honglak Lee, and Lu Wang. Exploring demonstration ensembling for in-context learning. *arXiv preprint arXiv:2308.08780*, 2023.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, 2:429–450, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pp. 1273–1282. PMLR, 2017.

Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICL: A tabular foundation model for in-context learning on large data. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony Caterini. Retrieval & fine-tuning for in-context tabular models. *Advances in Neural Information Processing Systems*, 37:108439–108467, 2024.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 4344–4353, 2019.

Geoffrey S Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pp. 359–372, 1964.

Derek Xu, Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular pfns. In *International Conference on Learning Representations*, volume 2025, pp. 24630–24665, 2025.

# Appendices

## A Proofs of theoretical results

### A.1 Proof of Theorem 1

Recall Definition 2, we have

$$\mathcal{E}_f^{(\lambda)}(x, \mathcal{D}^{(K)}) := \left\| f(x, \mathcal{D}) - f_\lambda(x, \mathcal{D}^{(K)}) \right\|, \tag{A.1}$$

where

$$f_\lambda(x, \mathcal{D}^{(K)}) := \sum_{k=1}^{K} \lambda_k f(x, \mathcal{D}_k).$$

The coefficient $\lambda$ is a convex combination which satisfies

$$\lambda \in \Delta^{K-1} := \{\lambda \in \mathbb{R}^K : \sum_{k=1}^{K} \lambda_k = 1,\ \lambda_k \geq 0\}.$$

Within in-context learning framework, the prediction function

$$f(x, \mathcal{D}) := \mathcal{F}(u(x, \mathcal{D})). \tag{A.2}$$

The $\mathcal{F}$ is a Fréchet differentiable map, which satisfies

$$\|D_u \mathcal{F}(u)\| \leq B, \tag{A.3}$$
$$\|D_u \mathcal{F}(u) - D_u \mathcal{F}(v)\| \leq L\|u - v\|. \tag{A.4}$$

Based on Definition 1, for given $\lambda$, we denote

$$\mathcal{F}_\lambda(u(x, \mathcal{D}^{(K)})) := \sum_{k=1}^{K} \lambda_k \mathcal{F}(u(x, \mathcal{D}_k)) \tag{A.5}$$

$$u_\lambda(x, \mathcal{D}^{(K)}) := \sum_{k=1}^{K} \lambda_k u(x, \mathcal{D}_k) \tag{A.6}$$

as the convex combination of $\mathcal{F}(u(x, \mathcal{D}^{(K)}))$ and $u(x, \mathcal{D}^{(K)})$, respectively. From Equation (A.1) and (A.2), we have

$$\begin{aligned}
\mathcal{E}_f^{(\lambda)}(x, \mathcal{D}^{(K)}) &= \left\| \mathcal{F}(u(x, \mathcal{D})) - \mathcal{F}_\lambda(u(x, \mathcal{D}^{(K)})) \right\| \\
&\leq \left\| \mathcal{F}(u_\lambda(x, \mathcal{D}^{(K)})) - \mathcal{F}_\lambda(u(x, \mathcal{D}^{(K)})) \right\| + \left\| \mathcal{F}(u(x, \mathcal{D})) - \mathcal{F}(u_\lambda(x, \mathcal{D}^{(K)})) \right\|.
\end{aligned}$$

We focus on first term $\left\| \mathcal{F}(u_\lambda(x, \mathcal{D}^{(K)})) - \mathcal{F}_\lambda(u(x, \mathcal{D}^{(K)})) \right\|$. For any $k$, we have

$$\mathcal{F}(u(x, \mathcal{D}_k)) - \mathcal{F}(u_\lambda(x, \mathcal{D}^{(K)})) = D_u \mathcal{F}(u_\lambda)[u(x, \mathcal{D}_k) - u_\lambda(x, \mathcal{D}^{(K)})] + \mathcal{R}_k, \tag{A.7}$$

where

$$\|\mathcal{R}_k\| \overset{Eq.\ (A.4)}{\leq} \frac{L}{2} \|u(x, \mathcal{D}_k) - u_\lambda(x, \mathcal{D}^{(K)})\|^2. \tag{A.8}$$

Thus, we have

$$\begin{aligned}
&\left\|\mathcal{F}_\lambda(u(x,\mathcal{D}^{(K)})) - \mathcal{F}(u_\lambda(x,\mathcal{D}^{(K)}))\right\| \\
\overset{Eq.\ (A.5)}{=}& \left\|\sum_{k=1}^{K}\lambda_k\left(\mathcal{F}(u(x,\mathcal{D}_k)) - \mathcal{F}(u_\lambda(x,\mathcal{D}^{(K)}))\right)\right\| \\
\overset{Eq.\ (A.7)}{=}& \left\|\sum_{k=1}^{K}\lambda_k\left(D_u\mathcal{F}(u_\lambda)\left[u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right] + \mathcal{R}_k\right)\right\| \\
\leq& \left\|\sum_{k=1}^{K}\lambda_k\left(D_u\mathcal{F}(u_\lambda)\left[u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right]\right)\right\| + \left\|\sum_{k=1}^{K}\lambda_k\mathcal{R}_k\right\| \\
=& \left\|(D_u\mathcal{F}(u_\lambda))\sum_{k=1}^{K}\lambda_k\left[u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right]\right\| + \left\|\sum_{k=1}^{K}\lambda_k\mathcal{R}_k\right\| \\
=& \left\|(D_u\mathcal{F}(u_\lambda))\left[\sum_{k=1}^{K}\lambda_k u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right]\right\| + \left\|\sum_{k=1}^{K}\lambda_k\mathcal{R}_k\right\| \\
\overset{Eq\ (A.6)}{=}& \left\|\sum_{k=1}^{K}\lambda_k\mathcal{R}_k\right\| \overset{Eq\ (A.8)}{\leq} \frac{L}{2}\sum_{k=1}^{K}\lambda_k\left\|u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right\|^2 \\
:=& \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}).
\end{aligned} \tag{A.9}$$

The $\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}) := \sum_{k=1}^{K}\lambda_k\mathcal{V}(x,\mathcal{D}_k)$ is the convex combination of $\mathcal{V}(x,\mathcal{D}^{(K)})$, where

$$\mathcal{V}(x,\mathcal{D}_k) := \left\|u(x,\mathcal{D}_k) - u_\lambda(x,\mathcal{D}^{(K)})\right\|^2, \quad \text{for } k = 1,\cdots,K.$$

Next, we analyze the second term $\left\|\mathcal{F}(u(x,\mathcal{D})) - \mathcal{F}(u_\lambda(x,\mathcal{D}^{(K)}))\right\|$. Based on Equation (A.3), we directly have

$$\begin{aligned}
\left\|\mathcal{F}(u(x,\mathcal{D})) - \mathcal{F}(u_\lambda(x,\mathcal{D}^{(K)}))\right\| &= \left\|D_u(\xi)\left[u(x,\mathcal{D}) - u_\lambda(x,\mathcal{D}^{(K)})\right]\right\| \\
&\overset{Eq.\ (A.3)}{\leq} B\left\|u(x,\mathcal{D}) - u_\lambda(x,\mathcal{D}^{(K)})\right\| \\
&\overset{Def}{=}{}^{2} B\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}).
\end{aligned} \tag{A.10}$$

Based on Equation (A.9) and (A.10), we have

$$\mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)}) \leq B\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}) + \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}),$$

which is exactly the result for Theorem 1.

### A.2 Proof of Corollary

Recap the Definition 2,

$$\mathcal{E}_f(x,\mathcal{D}^{(K)}) := \min_{\lambda(x)\in\Delta^{K-1}}\mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)}).$$

Take $\lambda(x) = \arg\min_{\lambda(x)\in\Delta^{K-1}}\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)})$. Based on the Theorem 1, we have

$$\begin{aligned}
\mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)}) &\leq B\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}) + \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}) \\
&= B\mathcal{E}_u(x,\mathcal{D}^{(K)}) + \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}).
\end{aligned} \tag{A.11}$$

Futhermore, we have

$$\begin{aligned}\mathcal{E}_f(x,\mathcal{D}^{(K)}) \leq& \mathcal{E}_f^{(\lambda)}(x,\mathcal{D}^{(K)}) \\ \overset{Eq.\ (A.11)}{\leq}& B\mathcal{E}_u(x,\mathcal{D}^{(K)}) + \frac{L}{2}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}) \\ \leq& B\mathcal{E}_u(x,\mathcal{D}^{(K)}) + \frac{L}{2}\cdot\max_{\lambda\in\Delta^{K-1}}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}). \end{aligned} \tag{A.12}$$

When $u$ is convex decomposable, from Definition 2, we have

$$\mathcal{E}_u(x,\mathcal{D}^{(K)}) = 0. \tag{A.13}$$

Therefore, we have

$$\mathcal{E}_f(x,\mathcal{D}^{(K)}) \overset{Eq.\ (A.12)}{\leq} B\mathcal{E}_u(x,\mathcal{D}^{(K)}) + \frac{L}{2}\cdot\max_{\lambda\in\Delta^{K-1}}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}) \overset{Eq.\ (A.13)}{=} \frac{L}{2}\cdot\max_{\lambda\in\Delta^{K-1}}\mathcal{V}_\lambda(x,\mathcal{D}^{(K)}).$$

This completes the proof.

### A.3 Proof of Theorem 2

Let $\lambda(x)\in\Delta^{K-1}$. We have

$$\begin{aligned}\mathcal{E}_u^{(\lambda)}(x,\mathcal{D}^{(K)}) =& \left\| u(x,\mathcal{D}) - \sum_{k=1}^K \lambda_k(x)u(x,\mathcal{D}_k)\right\| \\ \overset{Eq.3.3}{=}& \left\| \int_{\mathcal{X}\times\mathcal{Y}} \psi(x_i,y_i)\nu_{\mathcal{D}}(x,d(x_i,y_i)) - \sum_{k=1}^K \lambda_k(x)\int_{\mathcal{X}\times\mathcal{Y}}\psi(x_i,y_i)\nu_{\mathcal{D}_k}(x,d(x_i,y_i))\right\| \\ :=& \left\| \int_{\mathcal{X}\times\mathcal{Y}} \psi(x_i,y_i)\mu_\lambda(x,d(x_i,y_i))\right\| \\ &, \end{aligned}$$

where

$$\mu_\lambda(x,\cdot) := \nu_{\mathcal{D}}(x,\cdot) - \sum_{k=1}^K \lambda_k(x)\nu_{\mathcal{D}_k}(x,\cdot).$$

From Equation 3.2 and 3.3, we have

$$\begin{aligned}\mu_\lambda(x,\cdot) =& \nu_{\mathcal{D}}(x,\cdot) - \sum_{k=1}^K \lambda_k(x)\nu_{\mathcal{D}_k}(x,\cdot) \\ =& \sum_{(x_i,y_i)\in\mathcal{D}} a_{\mathcal{D}}(x,(x_i,y_i))\delta_{(x_i,y_i)} - \sum_{k=1}^K \lambda_k(x)\sum_{(x_i,y_i)\in\mathcal{D}_k} a_{\mathcal{D}_k}(x,(x_i,y_i))\delta_{(x_i,y_i)} \\ =& \sum_{k=1}^K \sum_{(x_i,y_i)\in\mathcal{D}_k} \left[a_{\mathcal{D}}(x,(x_i,y_i)) - \lambda_k(x)a_{\mathcal{D}_k}(x,(x_i,y_i))\right]\delta_{(x_i,y_i)}. \end{aligned}$$

Therefore, $\int_{\mathcal{X}\times\mathcal{Y}}\psi(x_i,y_i)\mu_\lambda(x,d(x_i,y_i))$ satisfies

$$\begin{aligned}\left\|\int_{\mathcal{X}\times\mathcal{Y}}\psi(x_i,y_i)\mu_\lambda(x,d(x_i,y_i))\right\| =& \left\|\sum_{k=1}^K\sum_{(x_i,y_i)\in\mathcal{D}_k}\left[a_{\mathcal{D}}(x,(x_i,y_i)) - \lambda_k(x)a_{\mathcal{D}_k}(x,(x_i,y_i))\right]\psi(x_i,y_i)\right\| \\ \leq& \sum_{k=1}^K\sum_{(x_i,y_i)\in\mathcal{D}_k}\left\|\left[a_{\mathcal{D}}(x,(x_i,y_i)) - \lambda_k(x)a_{\mathcal{D}_k}(x,(x_i,y_i))\right]\psi(x_i,y_i)\right\| \\ \leq& \sum_{k=1}^K\sum_{(x_i,y_i)\in\mathcal{D}_k}\left\|a_{\mathcal{D}}(x,(x_i,y_i)) - \lambda_k(x)a_{\mathcal{D}_k}(x,(x_i,y_i))\right\|\left\|\psi_{\max}\right\|, \end{aligned}$$

where $\psi_{\max} := \max_{(x,y)\in\mathcal{D}}\psi(x,y)$.