Predicting Enterprise Users' Consuming Potential for Cloud Services

Yunlong Cheng⋆[1], Tianyao Shi⋆[1], Xiuyuan Wei[1], Yulong Song[1], Xiaofeng Gao[1]✉, Zhipeng Bian[2], and Zhenli Sheng[2]

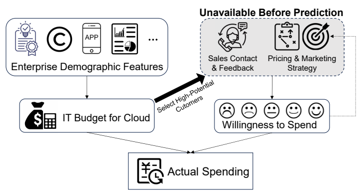
[1] Shanghai Key Laboratory of Scalable Computing and Systems, College of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai, China
`{aweftr, sthowling, 20001225, sylacd, gaoxiaofeng}@sjtu.edu.cn`
[2] Huawei Technologies Co., Ltd., Shengzhen, China
`{bianzhipeng1, shengzhenli}@huawei.com`

Abstract. Cloud platforms' revenue mainly depends on enterprise users. To target high-potential customers, the platform wants to identify yet-to-adopt-cloud enterprises with substantial IT budgets for cloud services. Since directly predicting future spending is impractical, we propose a new problem: predicting enterprise users' annual IT budgets, which represent their consuming potential. To the best of our knowledge, no existing literature has studied this topic. In this paper, we propose a novel holistic two-stage framework, BSA-DaMaM, that successfully counters all major challenges of this problem—the lack of ground-truth labels for budgets, the difference in expected prediction grains for various user groups, and the high missing ratio of enterprise demographic features. The first stage leverages Influence Functions and expert annotations to improve budget approximations in a Human-in-the-loop paradigm. For the second stage, we design a Dual-attention Missing-aware Multi-gate Mixture-of-Experts (DaMa-MMoE) network, which learns missing-aware user embeddings and adapts to different prediction grain requirements. Extensive experiments on proprietary data and online deployment validate the effectiveness of BSA-DaMaM.

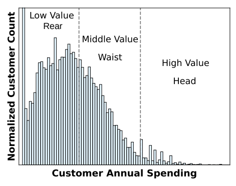
1 Introduction

Cloud platforms rely heavily on enterprise users for revenue. Identifying high-potential, not adopted enterprises and predicting their spending can enable targeted marketing efforts, thereby increasing platform revenue. This is typically framed as the lifetime value (LTV) prediction problem [1, 2, 16], but directly predicting actual spending for non-adopters is impractical. According to Behavioral Economics, customer spending depends on two factors: budget and consuming

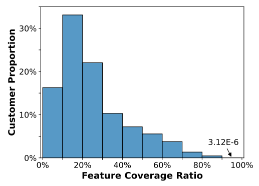
⋆ Y. Cheng and T. Shi contributed equally. X. Gao is the corresponding author. This work was supported by the National Key R&D Program of China [2024YFF0617700]; the National Natural Science Foundation of China [U23A20309, 62372296, 62272302], Shanghai Municipal Science and Technology Major Project [2021SHZDZX0102], and the Huawei Cloud [TC20220718012, TC20230526044].


Unavailable Before Prediction
Enterprise Demographic Features
Sales Contact & Feedback
Pricing & Marketing Strategy
Select High-Potential Customers
IT Budget for Cloud
Willingness to Spend
Actual Spending

(a)



(b)



(c)

Fig. 1: (a) Business logic behind enterprises' actual spending. Budget and willingness are two independent latent factors that decide spending. (b) The long-tail distribution of enterprise users' annual spending. The leftmost bar representing zero-spending users is clipped for beauty. (c) The distribution of non-missing feature coverage of enterprises.

willingness, as is shown in Fig. 1a. For non-adopters, there is no behavioral data, and only demographic data (e.g., patent counts) is available, making predictions of willingness unreliable. While willingness is assessed after sales contact, budget values, reflecting consumption potential, can be predicted. By identifying high-budget companies, the sales team can refine strategies based on feedback on their willingness. To the best of our knowledge, **no existing literature has studied this new problem of predicting user budgets**.

The new problem comes with new challenges. First, there are no ground-truth labels for budgets. The observed spending is often far below actual budgets, making it an unreliable approximation. Expert annotations are helpful but are scarce and imprecise, sometimes differing by hundreds of thousands of dollars, leading to Data Inconsistency [12] and Noisy Labels [13]. Second, marketing needs have different requirements on prediction grains for different user groups. As shown in Fig. 1b, the head users (2% of total) contributing over 80% of the total revenue for our cloud. The sales team needs exact predictions (i.e., regression) for these high-value users, while rough classifications suffice for other groups. A single model struggles to cover this wide range of budgets. Finally, the enterprise demographic data suffer from low data quality. Fig. 1c shows sampled paid enterprises' feature coverage ratio distribution on our platform. Most users have less than 50% feature coverage, and missing features often cluster together.

In this paper, we define a new problem of predicting enterprise users' IT budgets for cloud services and propose a novel holistic framework, BSA-DaMaM, to address the challenges above. The framework includes two major components. One is the Budget-Spending Alignment (BSA), which bridges the gap between budgets and spending records in a Human-in-the-loop manner. Another is the Dual-attention Missing-aware Multi-gate Mixture of Experts (DaMa-MMoE, or DaMaM in short) network, where the DaMa learns missing-aware user embeddings and the MMoE satisfies the prediction grain requirements by multi-task learning. The MMoE structure [10] enables different experts to focus on missing semantics of different feature groups.

In short, we make the following contributions:

- To the best of our knowledge, we are the first to define the problem of predicting enterprise users' IT budgets in the cloud services setting.
- We propose BSA-DaMaM, a novel holistic two-stage framework that addresses key challenges: the lack of ground-truth labels for user budgets, varying prediction grain requirements across user groups, and the high missing ratio of demographic features, through innovative algorithm design and pragmatic engineering practice.
- Extensive experiments on industrial datasets verifies that BSA-DaMaM achieves satisfying performance.

2 Problem Statement

2.1 User Budgets Prediction

Let X denote the demographic features of enterprise users and $y \in [0, +\infty)$ be the corresponding actual annual average spending on a cloud platform A. The budget z and the willingness q ($q \in [0,1]$) are independent latent factors determining y ($z \geq y$), with z depending on X and q dependent on the user feedback U and marketing strategies V. The relationship of them can be described as

$$y = \mathbb{E}[z \cdot q \,|\, X, U, V] = \mathbb{E}[z \,|\, X] \cdot \mathbb{E}[q \,|\, U, V].$$

The sampled dataset is $\mathcal{D} = \mathcal{D}_o \cup \mathcal{D}_a$, where $\mathcal{D}_o = \{(\mathbf{x}_i, y_i)\}_{i=1}^{M}$ and $\mathcal{D}_a = \{(\mathbf{x}_i, y_i, \tilde{z}_i)\}_{i=1}^{N}$ are the original samples and samples with expert annotations on budgets, respectively ($N \ll M$). Given that U, V are unavailable, we aim to train a model f to predict z on an adjusted dataset $\check{\mathcal{D}} = \{(\mathbf{x}_i, \check{z}_i)\}_{i=1}^{M+N}$, where $\check{z}_i = nanmax(y_i, \tilde{z}_i)$ approximates the real z_i. The prediction can be represented as $\hat{z}_i = f(\mathbf{x}_i \,|\, \check{\mathcal{D}}, \theta)$, where $\theta \in \Theta$ is the parameter set of our model.

2.2 Budget-Spending Alignment

Since $\check{z}_i$ comes from either y_i or $\tilde{z}_i$, it is likely that $\check{z}_i \neq z_i$, making such users noisy(-label) samples. We define *data inconsistency* as follows: let an oracle $\mathcal{O}$ with a threshold ϵ_z perfectly measures the similarity of any given user pairs (c_i, c_j) in terms of budget values, $\mathcal{O}(\cdot,\cdot) \in \{0,1\}$ and $\mathcal{O}(\mathbf{x}_i, \mathbf{x}_j) = 1$ iff $d(z_i, z_j) = \frac{|z_i - z_j|}{|z_i + z_j|} \leq \epsilon_z$. Data inconsistency happens in $\check{\mathcal{D}}$ if

$$\exists (i,j),\ \mathcal{O}(\mathbf{x}_i, \mathbf{x}_j) \cdot \mathbb{1}(d(\check{z}_i, \check{z}_j) > \epsilon_z) = 1.$$

Budget-spending alignment aims to find a set of noisy samples $\check{\mathcal{D}}_n = \{(\mathbf{x}_i, \check{z}_i) | \check{z}_i \neq z_i\} \subset \check{\mathcal{D}}$ and its corresponding corrections of budgets $\{\check{z}_i'\}$, such that after replacing $\check{\mathcal{D}}_n$ with $\check{\mathcal{D}}_n' = \{(\mathbf{x}_i, \check{z}_i') | \check{z}_i \neq z_i\}$, the overall inconsistency in $\check{\mathcal{D}}' = \check{\mathcal{D}} \backslash \check{\mathcal{D}}_n \cup \check{\mathcal{D}}_n'$

$$INC(\check{\mathcal{D}}') = \sum_{i \neq j} \mathcal{O}(\mathbf{x}_i, \mathbf{x}_j) \cdot \mathbb{1}(d(\check{z}_i', \check{z}_j') > \epsilon_z) \tag{1}$$

is minimized. Once such $\check{\mathcal{D}}'$ is obtained, we remove all the accent notations for convenience and denote it as $\mathcal{D}' = \{(\mathbf{x}_i, z_i)\}_{i=1}^{M+N}$.

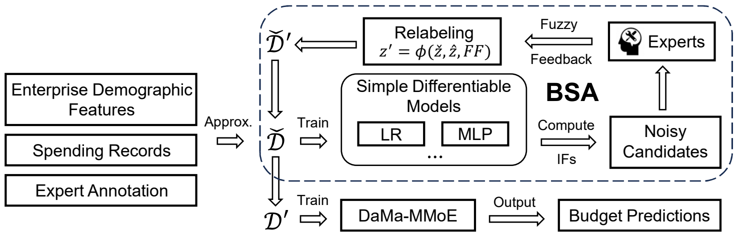


Fig. 2: The two-stage BSA-DaMaM framework.

2.3 Multi-Task Objectives

Enterprise users can be divided into C sections $\{1, \cdots, C\}$ based on their budget values z, with a step-wise mapping function m. Users in section C, having the largest budgets, are termed head users, while the remaining sections correspond to waist and tail users. The practical needs of the sales team require different prediction grains for the two user groups above, as specified by:

Task 1. **User Multi-Classification on Budgets**. Given a set of samples $\mathcal{D}^{clf} = \{(\mathbf{x}_i, z_i^c) \in X \times \{1, \cdots, C\}\}_{i=1}^{M+N}$, where $z_i^c = m(z_i)$. The model aims to predict user u's most likely budget section $\hat{z}_u^c \in \{1, \cdots, C\}$.

Task 2. **Head User Budgets Regression.** Given a set of samples $\mathcal{D}^{reg} = \{(\mathbf{x}_i, z_i) \in X \times [th, +\infty)\}_{i=1}^{N_C}$, where N_C is the number of head users in $\mathcal{D}^{clf}$ ($N_C \ll M + N$), and th is the threshold of being head users, $m(z_i | z_i \geq th) = C$. The model aims to predict head user u's budget value $\hat{z}_u \in [th, +\infty)$.

3 Method

Fig. 2 illustrates the proposed BSA-DaMaM framework. It operates in a two-stage paradigm. First, BSA iteratively removes noisy labels from the adjusted dataset $\check{\mathcal{D}}$. Then, the DaMaM network optimizes two budget prediction tasks, focusing on various aspects of missing states.

3.1 The BSA Framework

The budget-spending alignment is essentially a two-step procedure: first, find noisy samples whose budget and spending do not match, and then find their correct labels of budgets. We compute the Influence Functions (IFs) [18] of training samples via simple differentiable models to achieve the first step. For a sample $c = (\mathbf{x}, \check{z})$, we define the influence of upweighting c on the loss at a test point c_t as $\mathcal{I}_{\mathrm{u,l}}(c, c_t)$. Since BSA is not focused on optimal performance, simple differential models like Logistic Regression (LR) and Multi-Layer Perceptron (MLP) are used. If $(\mathbf{c_i}, \mathbf{c_j})$ **is an incident of data inconsistency, then** $\mathcal{I}_{\mathbf{u,l}}(\mathbf{c_i}, \mathbf{c_j}) > \mathbf{0}$. Theoretically, similar samples c_i and c_j should benefit from each other's upweighting, with $\mathcal{I}_{\mathrm{u,l}}(c_i, c_j) < 0$. However, now upweighting

Algorithm 1: The BSA Process

```
Data: Adjusted dataset $\check{\mathcal{D}}$, budget difference threshold $\epsilon_z$, loss threshold $\epsilon_l$,
      sample size $T_c, T_w$, expand/shrink ratio $\alpha$, max iteration $T$
Result: Aligned dataset $\mathcal{D}'$
1  $t \leftarrow 0, \check{\mathcal{D}}^{(0)} \leftarrow \check{\mathcal{D}}$, $\hat{\theta}^{(0)} \leftarrow \arg\min_{\theta \in \Theta} \frac{1}{|\check{\mathcal{D}}|} \sum_{i=1}^{|\check{\mathcal{D}}|} l(c_i; \theta)$;
2  $\mathcal{D}_c \leftarrow$ Sample($\{c_i \mid l(c_i; \hat{\theta}^{(0)}) < \epsilon_l\}, T_c$) $\cup$ ExpertClean(
     Sample($\{c_i \mid d(\hat{z}_i^{(0)}, \check{z}_i^{(0)}) > \epsilon_z, \check{z}_i^{c(0)} \geq 1\}, T_w$));
3  while $\mathit{ValLossTrace}(\hat{\theta}^{(t)})$ and $t \leq T$ do
4      $\{\hat{z}_i^{(t)}\} \leftarrow \{f(\mathbf{x_i} | \check{\mathcal{D}}^{(t)}, \hat{\theta}^{(t)})\}$;
5      $\check{\mathcal{D}}_h^c \leftarrow \bigcup_{c_j \in \mathcal{D}_c}$ Top-$k$($\{c_i^{(t)} \mid \mathcal{I}_{\mathrm{u,l}}^{(t)}(c_i^{(t)}, c_j) > 0, d(\check{z}_i^{(t)}, z_j) > \epsilon_z\}$);
6      $\check{\mathcal{D}}_n, FF \leftarrow$ Expert($\check{\mathcal{D}}_h^c$);
7      $\check{\mathcal{D}}_n' \leftarrow \{(\mathbf{x}_i, \phi(\check{z}_i^{(t)}, \hat{z}_i^{(t)}, FF_i, \alpha)) \mid c_i^{(t)} \in \check{\mathcal{D}}_n\}$;
8      $t \leftarrow t + 1$;
9      $\check{\mathcal{D}}^{(t)} \leftarrow \check{\mathcal{D}}^{(t-1)} \setminus \check{\mathcal{D}}_n \cup \check{\mathcal{D}}_n'$;
10     $\hat{\theta}^{(t)} \leftarrow \arg\min_{\theta \in \Theta} \frac{1}{|\check{\mathcal{D}}|} \sum_{i=1}^{|\check{\mathcal{D}}|} l(c_i^{(t)}; \theta)$;
11 end
12 $\mathcal{D}' \leftarrow \check{\mathcal{D}}^{(t)}$;
```

c_i increases the loss on c_j, this suggests that either c_i or c_j is a noisy sample. Let $\check{\mathcal{D}}_h = \{c_i \mid \exists c_j, \mathcal{I}_{\mathrm{u,l}}(c_i, c_j) > 0, d(\check{z}_i, \check{z}_j) > \epsilon_z\}$ denote the set of *harmful* training samples, then we have $\check{\mathcal{D}}_n \subset \check{\mathcal{D}}_h$. We can use $\check{\mathcal{D}}_h$ as a noisy candidate set.

The complexity of calculating $\mathcal{I}_{\mathrm{u,l}}$ for all (c_i, c_j) pairs is $\mathrm{O}(n^2 p)$, which is too expensive and time-consuming. To mitigate this, we pick samples from a small, clean subset of data $\mathcal{D}_c$ ($|\mathcal{D}_c| \ll n$) to confirm that c_j is not a noisy, then getting the corresponding harmful training samples $\check{\mathcal{D}}_h^c$ from $\mathcal{D}_c$ is tractable in time. A clean $\mathcal{D}_c$ can be obtained via the popular *small-loss trick* [13], where training points with loss smaller than threshold ϵ_l are considered clean. We then sample a subset of such data points with size T_c to get $\mathcal{D}_c$. Additionally, we sample T_w mispredicted data points from head and waist users and let experts decide which are clean samples, in case there are not enough data points in these sections.

After identifying candidate noisy samples, the experts will inspect them and can only give **fuzzy feedback**, i.e., Too Large (TL), Too Small (TS), or Almost Correct (AC), rather than precise values. We devise a simple relabeling heuristic $\check{z}' = \phi(\check{z}, \hat{z}, FF, \alpha)$ to exploit fuzzy feedback:

$$\check{z}_i^{c\prime} = \begin{cases} \min(\check{z}_i^c, \hat{z}_i^c - 1) \\ \max(\check{z}_i^c, \hat{z}_i^c + 1) \\ \hat{z}_i^c \end{cases}, \check{z}_i' = \begin{cases} \min(\check{z}_i, \alpha \cdot \hat{z}_i) & TL \\ \max(\check{z}_i, \frac{1}{\alpha} \cdot \hat{z}_i) & TS \\ round(\hat{z}_i) & AC \end{cases}$$

where $\alpha \in (0, 1)$ is the expand/shrink ratio and $round(\cdot)$ means to round $\hat{z}_i$ to nearest multiple of 50k/500k/... The tentative corrections replace the old labels, and then the model is retrained. The Training-Identification-Correction loop continues until a stop criterion, such as a maximum of T rounds or signs of noisy labels disappear, is met. Alg. 1 summarizes the BSA framework.

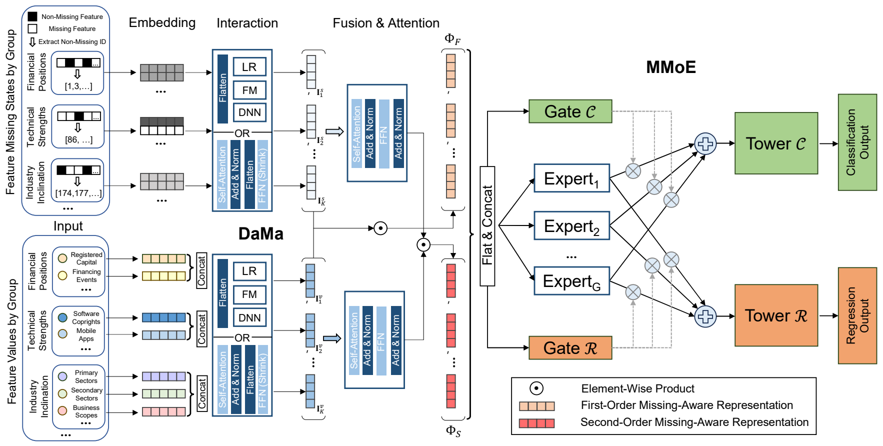


Fig. 3: The DaMa-MMoE network. DaMa learns missing-aware feature embeddings through a symmetric dual-attention subnet. MMoE leads experts to focus on different missing states and features for different budget prediction tasks.

3.2 The DaMa-MMoE Network

Once we get the aligned dataset $\mathcal{D}'$, we design the DaMa-MMoE Network to address the remaining challenges of low feature coverage and multi-task objectives, as illustrated in Fig. 3. DaMa first encodes the feature IDs and feature values of $\mathbf{x} \in \mathbb{R}^m$ into d_e-dim dense representations through two sets of **embedding layers**: $\mathbf{e}_i^s = Emb_s(i)$ and $\mathbf{e}_i^v = Emb_v(x_i)$, where $\mathbf{e}_i^s, \mathbf{e}_i^v \in \mathbb{R}^{d_e}$.

For each feature group k and corresponding set of features X_k, the feature missing state is $\mathbf{s}_k := [i \,|\, x_i \neq \rho_i, i \in X_k]$, where ρ_i is the default filling value of x_i. $\mathbf{s}_k$ is a varied-length sequence of all non-missing features' ID in X_k. We pad $\mathbf{s}_k$ at the tail to match the max length of sequences and get the padded $\mathbf{s}_k'$. The group missing state embedding is thus given by $\mathbf{e}_i^{s,k} = Emb_s(\mathbf{s}_k')$.

Next, DaMa models the interaction among features and non-missing states in each feature group X_k through **interaction layers**. Each feature group has both dense numerical features and sparse categorical features. To learn their relationships, encoding methods like WDL, deepFM, and transformers [15] can be used. The interaction layers outputs are given by $\mathbf{I}_s^k = Interaction_s(\mathbf{e}_i^{s,k})$ and $\mathbf{I}_v^k = Interaction_v([\mathbf{e}_i^v \,|\, i \in X_k])$, where $\mathbf{I}_s^k, \mathbf{I}_v^k \in \mathbb{R}^{d_i}$, d_i is the output dimension.

Attention and Fusion Layers. After the interaction layers, we get two sequences of embeddings $\mathbf{I}^s = [\mathbf{I}_1^s, \cdots, \mathbf{I}_K^s]$ and $\mathbf{I}^v = [\mathbf{I}_1^v, \cdots, \mathbf{I}_K^v]$, which capture the interactions among feature-missing states and feature values inside each feature group. The first-order missing-aware representation is given by $\Phi_F = \mathbf{I}^s \odot \mathbf{I}^v$, where $\odot$ is the element-wise product operation. Φ_F captures the *intra-group* missing information and semantics of feature values. We use transformers to model the second-order feature interactions across groups: $\Phi_S = \mathbf{A}^s \odot \mathbf{A}^v$, where

$\mathbf{A}^s$ and $\mathbf{A}^v$ are the transformer output of $\mathbf{I}^s$ and $\mathbf{I}^v$, respectively. Φ_S captures the *inter-group* missing information and semantics of feature values.

MMoE. Multi-gate Mixture-of-Experts capture the difference of multiple tasks with a lightweight parameters. Suppose there are M_T tasks, and h^k is the tower network corresponding to the k-th task, the MMoE can be written as

$$\hat{z}^k \in \{\hat{z}^c, \hat{z}\},\ \hat{z}^k = h^k\Big(\sum_{i=1}^{G} g^k(\Phi_{\mathbf{x}})_i f_i(\Phi_{\mathbf{x}})\Big), \tag{2}$$

where $\Phi_{\mathbf{x}} = (\Phi_F \| \Phi_S)$ is the concatenation of missing-aware representations, $g^k(\Phi_{\mathbf{x}}) = softmax(W_{gk}\Phi_{\mathbf{x}})$ is the gating layer for the k-th task, and G is the number of experts. Each tower network gets a dedicated embedding based on the routing of the corresponding gating layer. We use MLPs as the tower networks. The final objective is $\mathcal{L} = L_{clf} + \beta \cdot L_{reg} + \gamma \cdot \|\theta\|_2^2$, where β, γ are weights for the regression task and l_2-regularization. For the classification task, the output layer is one-dim and Ordinal Regression loss [7] is used. For the regression task, the output layer models a log-normal distribution [16] and log-normal loss is used.

4 Experiments

4.1 Datasets

To the best of our knowledge, no publicly available dataset exists for budget prediction. Therefore, we use data sampled from Huawei Cloud's spending records from 2018 to 2022, including a total of 156k enterprises, and supplemented with expert annotations for approximately 3k enterprises. Enterprises are divided into 5 sections, with head users accounting for 2.18% of the total. Enterprise demographic features include 10 different tables collected from several data providers, each highlighting a unique aspect of an enterprise's strengths.

4.2 Evaluation on Budget-Spending Alignment

We compare BSA with four baselines: Rule-based data cleaning [3], Nearest Neighbor (which modifies outliers to the mean of their neighbors' budgets), Top-loss (which identify samples with the largest training loss as noisy candidates), and DUTI [20]. We use the **relative performance gain** as the evaluation metric since we lack ground-truth labels for noisy samples. The gain is calculated as the performance difference in test set before and after label correction. We use macro-average F1-score for classification and SMAPE for regression tasks.

Fig. 4 shows that as the number of corrected samples increases, BSA's relative performance gain improves steadily and achieves the best with sufficient corrections. Rule-based data cleaning has little to no positive impact, while NN gains in classification but fails in regression. Top-loss presents a similar curve to that of BSA because they both use fuzzy feedback to correct samples but performs worse due to its weaker noisy candidate identification. DUTI excels with few corrected samples but struggles with scalability, and undersampling limits its ability to identify noisy candidates in classification.

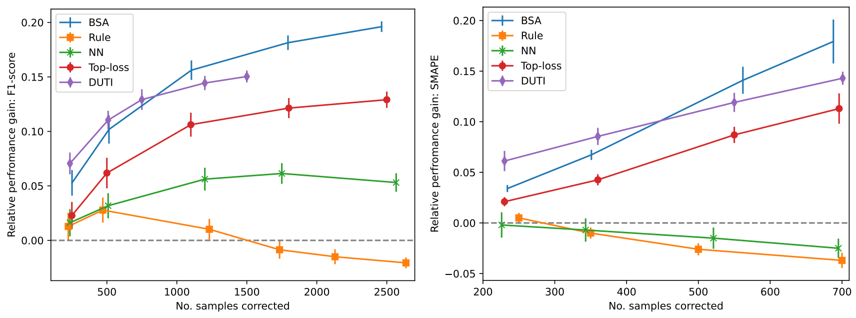


Fig. 4: Relative performance gain of dataset debugging methods.

Methods	F1	AUC	SMAPE	N.Gini
XGBoost	0.4667	0.7164	0.3471	0.5528
ZILN	0.3968	0.6332	0.3273	0.6052
MDME	0.5349	0.8101	0.2905	0.6824
Bidden-MarfNet	0.4120	0.6712	0.2820	0.6917
DaMa-MMoE	**0.5488**	**0.8239**	**0.2746**	**0.7011**

Table 1: Performance of different methods on two tasks.

4.3 Evaluation on Budget Prediction

We compare DaMa-MMoE with four well-known LTV prediction methods on the aligned dataset $\mathcal{D}'$: Two-stage XGBoost [4], ZILN [16], MDME [9], Bidden-MarfNet [19]. For classification, we use macro-average F1-score and AUC as the metric. For regression, we use SMAPE to evaluate the models' ability to predict budget values and Normalized Gini [16] to evaluate the accuracy of the users' ranking. We modify the definition of p in the ZILN loss from whether a sample pays to whether it belongs to head users to match the regression task's scope.

Table 1 compares the results. DaMa-MMoE outperforms all baselines across all metrics for its most suitable design for the data characteristics in the budget prediction. ZILN and Bidden-MarfNet struggle with classification because they are not designed for multi-classification. They give decent performance on classifying the lowest rear and the head users' class. When it comes to SMAPE and Normalized Gini, deep models perform better than XGBoost due to their stronger learning capabilities. Bidden-MarfNet outperforms MDME, probably due to its dedicated design for the feature missing issue, as well as the absence of short-term budgets to guide the annual-budget MDME.

4.4 Post-Deployment Performance

Our model replaced the previous XGBoost-based classification model in Huawei Cloud's marketing tool. The sales team can now query high-potential users by specifying the budget value conditions and generate user acquisition tasks based on budget predictions. Although it usually takes quite a period for the sales team from identifying high-potential users to settling down the contract, we observed a significant increase in usage statistics in terms of query and task-push activities.

We compare the statistics from 12/2023 $\sim$ 01/2024 with those from 12/2022 $\sim$ 01/2023. The overall month-over-month (MoM) uplift measures the change ratio of January compared to November, which shows an increase of 22.9%. The average MoM uplift is the geometric average of the two months' usage change, which has risen by 17%. The overall year-over-year change, comparing the same total count in the two months from the previous year, reflects an increase of 30.2%. These results suggest that the sales team is effectively using budget predictions, and the lack of complaints indicates satisfactory model performance.

5 Related Work

Learning with Noisy Labels. There are two main categories of approaches that address the noisy label problem. The first is to learn with the noise and mitigate its impact. This includes non-deep learning methods [6] and deep learning methods like robust architecture, sample selection, and loss adjustment [13, 11].However, these methods are often limited to symmetric noise and artificial data, struggling with real, asymmetric, and imbalanced data. The second category is to correct error labels in training samples from the data perspective, which is called training data(set) debugging [8, 20].

Customer Lifetime Value (LTV) Prediction. Early LTV prediction methods were mainly based on historical data or classic statistical models such as the RFM and Pareto/NBD model [5]. In recent years, ML and deep learning methods have gained much popularity for LTV prediction problems [1, 17]. By using models such as random forests, future values can be predicted based on individual customers [14]. Combinations of the SMOTE and XGBoost algorithms proved to be effective in improving prediction accuracy [2, 4].

6 Conclusion

In this paper, we propose a new problem of predicting enterprise users' IT budgets for cloud services. To address the lack of ground-truth labels, we first approximate them using spending records and expert annotations, then fix the possibly faulty approximations through the proposed BSA process based on Influence Functions. Next, our novel DaMa-MMoE network tackles the remaining challenges by encoding intra- and inter-group missing-aware information via a symmetric dual-attention network to obtain two missing-aware representation vectors. Extensive experiments on real-world datasets validate our approach, which we hope can be applied to other B2B businesses and inspire further research on identifying high-potential users.

References

1. Chamberlain, B.P., Cardoso, A., Liu, C.B., Pagliari, R., Deisenroth, M.P.: Customer lifetime value prediction using embeddings. In: SIGKDD. pp. 1753–1762 (2017)
2. Chen, P.P., Guitart, A., del Río, A.F., Periánez, A.: Customer lifetime value in video games using deep learning and parametric models. In: 2018 IEEE international conference on big data (big data). pp. 2134–2140 (2018)
3. Chu, X., Ilyas, I.F., Papotti, P.: Holistic data cleaning: Putting violations into context. In: ICDE. pp. 458–469 (2013)
4. Drachen, A., Pastor, M., Liu, A., Fontaine, D.J., Chang, Y., Runge, J., Sifa, R., Klabjan, D.: To be or not to be... social: Incorporating simple social features in mobile game customer lifetime value predictions. In: ACSW. pp. 1–10 (2018)
5. Fader, P.S., Hardie, B.G., Lee, K.L.: Rfm and clv: Using iso-value curves for customer base analysis. Journal of marketing research **42**(4), 415–430 (2005)
6. Franay, B., Verleysen, M.: Classification in the presence of label noise: A survey. TNNLS **25**(5), 845–869 (2013)
7. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Tao, D.: Deep ordinal regression network for monocular depth estimation. In: CVPR. pp. 2002–2011 (2018)
8. Koh, P.W., Liang, P.: Understanding black-box predictions via influence functions. In: ICML. pp. 1885–1894 (2017)
9. Li, K., Shao, G., Yang, N., Fang, X., Song, Y.: Billion-user customer lifetime value prediction: An industrial-scale solution from kuaishou. In: CIKM. pp. 3243–3251 (2022)
10. Ma, J., Zhao, Z., Yi, X., Chen, J., Hong, L., Chi, E.H.: Modeling task relationships in multi-task learning with multi-gate mixture-of-experts. In: SIGKDD. pp. 1930–1939 (2018)
11. Malach, E., Shai, S.S.: Decoupling "when to update" from "how to update". In: NeurIPS. pp. 960–970 (2017)
12. Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. arXiv preprint arXiv:1412.6596 (2014)
13. Song, H., Kim, M., Park, D., Shin, Y., Lee, J.G.: Learning from noisy labels with deep neural networks: A survey. TNNLS (2022)
14. Vanderveld, A., Pandey, A., Han, A., Parekh, R.: An engagement-based customer lifetime value system for e-commerce. In: SIGKDD. pp. 293–302 (2016)
15. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. NeurIPS **30** (2017)
16. Wang, X., Liu, T., Miao, J.: A deep probabilistic model for customer lifetime value prediction (2019)
17. Wu, T., Gu, R., Li, Y., Ma, H., Chen, Y., Zhu, Y., Yu, X., Xu, T., Huang, Y.: Raven: Benchmarking monetary expense and query efficiency of OLAP engines on the cloud. In: DASFAA. vol. 13946, pp. 593–605 (2023)
18. Wu, W., Flokas, L., Wu, E., Wang, J.: Complaint-driven training data debugging for query 2.0. In: SIGMOD. pp. 1317–1334 (2020)
19. Yang, X., Jia, B., Wang, S., Zhang, S.: Feature missing-aware routing-and-fusion network for customer lifetime value prediction in advertising. In: WSDM. pp. 1030–1038 (2023)
20. Zhang, X., Zhu, X., Wright, S.J.: Training set debugging using trusted items. In: AAAI. pp. 4482–4489 (2018)