LBC: Language-Based-Classifier for Out-Of-Variable Generalization

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have great success in natural language processing tasks such as response generation. However, their use in tabular data has been limited due to their inferior performance compared to traditional machine learning models (TMLs) such as XGBoost. We find that the pre-trained knowledge of LLMs enables them to interpret new variables that appear in a test without additional training, a capability central to the concept of Out-of-Variable (OOV). From the findings, we propose a Language-Based-Classifier (LBC), a classifier that maximizes the benefits of LLMs to outperform TMLs on OOV tasks. LBC employs three key methodological strategies: 1) Categorical changes to adjust data to better fit the model's understanding, 2) Advanced order and indicator to enhance data representation to the model, and 3) Using verbalizer to map logit scores to classes during inference to generate model predictions. These strategies, combined with the pre-trained knowledge of LBC, emphasize the model's ability to effectively handle OOV tasks. We empirically and theoretically validate the superiority of LBC. LBC is the first study to apply an LLM-based model to OOV tasks. The source code is at `https://github.com/ASDASDanonymous/Language-Based-Classifier-forOOVtasks`.

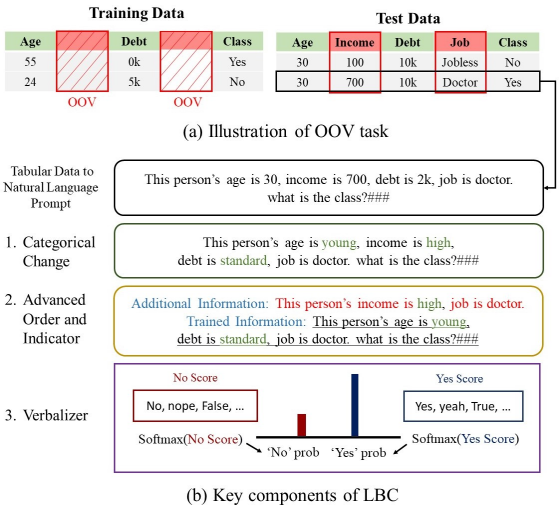

Figure 1: (a) Illustration of OOV task. The variables that were not present in the training data appear in the test data. (b) Key components of LBC to increase performance in OOV tasks. Categorical change refines data to make it easier for LBC to interpret. The advanced order and indicator method enhances the prompts that feed into LBC. The verbalizer aggregates the probabilities for a particular class scattered throughout the logit score and maps them to a specific class.

1 Introduction

LLMs have recently been applied to tabular data (Radford et al., 2018; Brown et al., 2020; Wang and Komatsuzaki, 2021; Devlin et al., 2018). Language-Interfaced-Fine-Tuning (LIFT) (Dinh et al., 2022) demonstrated that LLMs achieve reasonable performance on tabular data tasks while maintaining LLM's original structure. However, the pre-trained knowledge of LLMs holds even more potential: their ability to interpret Out-of-Variable (OOV), which refers to variables that were not seen during training but appear in the test data. So, we propose a new model called a **Language-Based-Classifier (LBC)** to solve the OOV tasks. Figure 1 outlines the structure of an LBC.

OOV tasks are an important problem and are the subject of several ongoing studies (Guo et al., 2024; Tzeng et al., 2015; Dreher et al., 2023). However, studies applying LLM to tabular data do not handle tabular data in an OOV setting. In real-world settings, a variety of constraints often hinder model training, emphasizing the importance of OOV tasks. For example, in healthcare, privacy and regulatory barriers prevent data sharing between hospitals. A model trained on Hospital A's data may encounter new, unseen variables when applied to Hospital B's data, leading to OOV situations. We argue that

LBC has strengths in handling OOV tasks, and our rationale is as follows. Converting tabular data to natural language prompts is intuitive, flexible, and easy. This transformation significantly simplifies the handling of OOVs, allowing us to seamlessly handle variables that might not have been discovered during training, overcoming a common limitation of TMLs. Furthermore, LBC leverages the pre-trained knowledge built into LLMs. Unlike TMLs, which struggle with data points or scenarios not present in the training set, LLMs leverage their inherent knowledge. We verified that LBC uses OOVs to increase the probability of the correct answer class based on pre-trained knowledge. These advantages are highlighted by the following three methodologies of LBC. First, Categorical Change involves converting numerical types to categorical types like 'high' and 'low' because these variables align better with the LBC, especially in OOV scenarios. Second, Advanced Order and Indicators optimize the sequence of variables to generate more effective prompts and introduce indicators to further boost performance. Third, the Verbalizer focuses on mapping LLM's logit scores to the desired class scores rather than relying on inconsistent output text, improving classification performance. We use the LOw-Rank Adaptation (LoRA) (Hu et al., 2021) to fine-tune the classifier. We theoretically prove that our model approximates an arbitrary classifier with LoRA fine-tuning.

To the best of our knowledge, LBC is the first study to apply an LLM-based classifier to solve the OOV tasks, and we validate LBC's superiority empirically and theoretically.

2 Related Works

2.1 Tabular Data Analysis with LLMs

LLMs now extend to analyzing tabular data. LIFT (Dinh et al., 2022) converts tabular data into natural language prompts for use in LLM and performs similarly to traditional models like XGBoost (Chen and Guestrin, 2016). LBC adds three methodologies to a similar foundation as LIFT, optimized to address OOV tasks. Models like TP-BERTa (Yan et al., 2024) and TabPFN (Hollmann et al., 2022) follow the LM structure but lack the ability to contextualize OOVs. On the other hand, LBC excels at handling OOV tasks and consistently outperforms existing models. LBC's performance in tabular data classification has been validated through theoretical analysis and statistical tests.

2.2 Out-of-Variable

Machine learning (ML) models often face the challenge of adapting to new environments with additional, unobserved variables (Ganin and Lempitsky, 2015). MomentLearn (Guo et al., 2023) was proposed to address this by using a predictor trained in a source environment and an additional objective matrix for partial derivatives for OOV tasks. However, its application in real-world scenarios is limited. The LBC method overcomes these limitations by leveraging the extensive prior knowledge of LLMs and methodologies for OOV tasks. Unlike MomentLearn, which is restricted to simple models such as linear or polynomial structures, LBC's use of LLMs allows for application to more complex models. This enhances its ability to discover intricate relationships between variables and offers greater generalization. Moreover, MomentLearn's reliance on an additional matrix, which must be trained with In-Variables, becomes less stable as the ratio of OOVs increases. In contrast, LBC only requires the training of a single predictor and has demonstrated robustness across varying OOV ratios, making it a more efficient and reliable solution for OOV challenges.

2.3 Verbalizer

Verbalizer is a mechanism for mapping the various output forms from an LLM to specific classes (Schick and Schütze, 2020; Schick and Schütze, 2021; Hu et al., 2022). Verbalizer contributes to reducing subjective bias in LLM by using a knowledge base to leverage diverse and comprehensive label words. It is also said that the noise of label words in classification can also be improved with a verbalizer. We argue that even in tabular data classification, we need a particular way to map the output of an LLM to the output of a classifier and that we should apply a verbalizer to it. The process by which LBC leverages the verbalizer to map LLM output to classifier output is shown in the verbalizer part of Figure 2.

2.4 Low-Rank Adaption

LoRA (Hu et al., 2021) has emerged as an innovation in adapting pre-trained models to specific tasks without extensive retraining of the entire model. LoRA introduces an approach to fine-tuning large pre-trained models. Instead of updating the whole parameter set, LoRA modifies a small subset of the model's weights through a low-rank matrix.

This method allows pre-trained models to adapt efficiently while maintaining their original structure and strengths. We theoretically validate the strong classification performance of LBC fine-tuned with LoRA, backed by the proven generalization ability of LoRA (Zeng and Lee, 2023).

3 Preliminary

3.1 Basic Dataset Conversion

This section describes the process of converting tabular data into text for input to LBC. Since our model relies on a frozen pre-trained LLM, converting tabular data into prompts is a crucial step. Let an instance of tabular data with K features be represented as $[[V_1 : x_1], [V_2 : x_2], \ldots, [V_K : x_K], [\text{class} : y]]$, where V_k is the kth variable name and x_k is the kth variable value. We need a method for the LLM to clearly distinguish between the variables in this dataset as inputs and the class as the output. This involves creating a conversion technique that clearly marks the end of the prompt and the beginning of the response while ensuring that the answer isn't overly lengthy. Therefore, we format the conversion as follows: "prompt: V_1 is x_1, V_2 is x_2, ..., V_K is x_K. What is the class? label: y@@@".

In this setup, the 'prompt' is the input to LLM, and the 'label' is the label for the data instance.

3.2 The Order of the Variables

During the tabular data to the prompt conversion process, one instance of tabular data converts to several different types of prompts based on the order of the variables. The total number of prompts that can be generated by changing the order of the variables is $K!$. Every prompt is a transformation of a single instance of tabular data, but the order of the variables gives it a different form, which causes LBC to interpret it differently. Therefore, the order of the variables is a factor that directly affects LBC's performance.

3.3 Fine-tuning LLM

Feeding the converted prompts into LBC yields a vector of vocabulary size, which is a logit for each word in the vocabulary. We use this logit to fine-tune the LLM. Let $\mathbf{Logit}$ be the logit vector for a single input prompt. During fine-tuning, J obtained from the model is used to compute the loss against the true labels. Let $\mathbf{Label}$ be the one-hot encoded vector of the true label for the input. The loss is calculated using a loss function J defined as follows:

$$J(\mathbf{Logit}, \mathbf{Label}) = \text{CE}(\mathbf{Logit}, \mathbf{Label})$$

where CE is cross-entropy with a logit loss function. After calculating the loss, the model's parameters are updated using an optimizer through gradient descent. The update rule in gradient descent can be described as follows:

$$\theta \leftarrow \theta - \eta \nabla_\theta J$$

Where θ is the model's parameters, η is the learning rate, and $\nabla_\theta J$ is the gradient of the loss with respect to the model parameters.

3.4 LLM-based Tabular Prediction

TMLs face significant challenges when processing textual data within feature sets. Text preprocessing inevitably leads to semantic information loss. Despite applying specialized techniques such as one-hot encoding or text vectorization methods (e.g., TF-IDF, Word2Vec, etc.), TMLs remain vulnerable to noise due to their lack of linguistic comprehension. Furthermore, the high dimensionality of text embeddings often impedes efficient learning, and attempts to mitigate this through dimensionality reduction techniques risk further information loss.

In contrast, LLMs offer a promising alternative for handling textual and numerical data in ML tasks. LLMs demonstrate superior capability in comprehending semantic content and discerning inter-feature relationships, which is beneficial when critical information is presented textually.

The previous approaches to LLM-based tabular data classification tasks (Dinh et al., 2022) rely on directly comparing the output text generated by the model with class texts such as 'no' or 'yes.' If the prediction is an exact match, it is classified with the corresponding class text. Conversely, if the output text differs, the model's prediction is marked as 'None' and automatically classified as incorrect. To address this limitation, we utilize the logit score to map directly to a specific class rather than using the model's output texts. For this mapping process, we utilize the probability values of the synonyms of the logit score's class text.

4 Methodology

4.1 Categorical Change

We find that LBC has a better interpretation of categorical variables than numerical ones because it

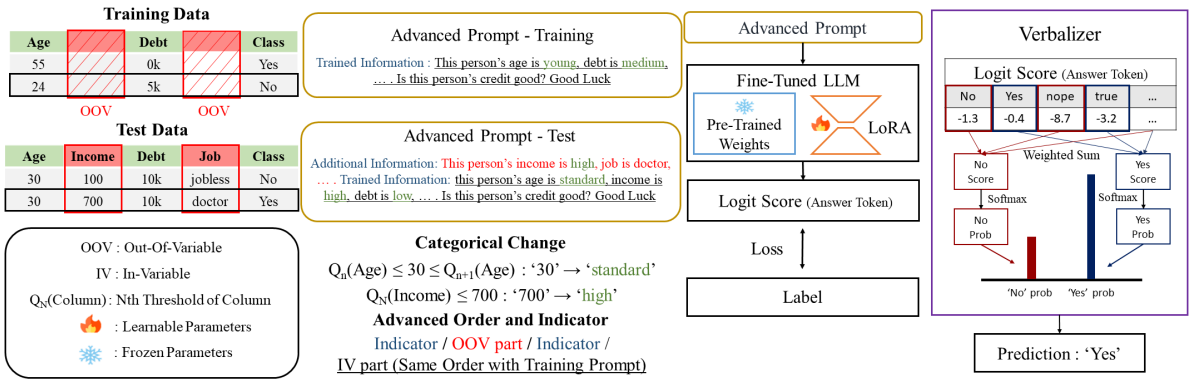

Figure 2: The overall process of an LBC performing an OOV task. LBC transforms tabular data into advanced prompt (AP) utilizing strategies that are 1) Categorical change and 2) Advanced order and indicator. These APs become input into an LLM that has been fine-tuned with a LoRA adapter to derive a logit score for the answer token. This logit score is assessed against the label to calculate loss, and during inference, the model prediction is generated by mapping the logit score to a class via a 3) Verbalizer.

is an LLM-based model. However, this poses a challenge as many key variables in tabular data are numerical. In particular, when LBC deals with OOVs, if the value of the input is numeric, pre-trained knowledge cannot be utilized, unlike categorical type values where the word itself has meaning. Therefore, we need a method to convert numerical variables to categorical types so that LBC leverages its pre-trained knowledge of important variables or OOVs for easier interpretation, and we find that mapping numerical variables to categorical variables using N categories improves the performance of LBC. The N categories are determined based on the principles of N-tiles, similar to quartiles but dividing the dataset into N equal parts. The thresholds are the values that divide the dataset into these N parts. For example, we converted values below the first threshold (Q_1) to "Category 1", between Q_1 and Q_2 to "Category 2", and so on, up to values above the last threshold (Q_{N-1}), which are converted to "Category N." A specific example sentence of Categorical Change is discussed in figure 2. The experimental results in Table 6 show that Categorical Change directly affects the performance of LBC.

4.2 The Advanced Order and Indicator

As shown in section 3.2, for a single instance of data, different prompts are generated depending on the order of the variables. The same problem occurs in the OOV task, where the number of variables increases due to the addition of OOVs, resulting in more variability in the prompts. This hinders LBC's ability to learn the relationships between tokens. Therefore, we find the format that performs best with optimal learning and inference among a large number of prompt formats, which can vary depending on the order of the OOVs and the trained variables that are not OOVs, called In-Variables (IVs). The format of the training and test prompts with both methods is as follows.

Training Prompt: IV Indicator + <u>IV part</u> + Question

Test Prompt: OOV Indicator + OOV part + IV Indicator + <u>IV part</u> + Question

By positioning the OOV part at the front of the prompt and matching the variable order of the <u>IV part</u> exactly as in training, the <u>IV part</u> in the test prompt has the exact same structure as the <u>IV part</u> in the training prompt. This allows LBC to apply the relationships between variables captured during training to the test as well. Also, since the indicator is always fixed in the same position, it allows LBC to distinguish between the OOV part and the IV part in training and inference. A prompt with both categorical change and advanced order and indicator applied is referred to hereafter as an **advanced prompt (AP)**. An example of an AP can be found in figure 2.

4.3 Generalization Ability of LBC: LoRA

According to Zeng and Lee (2023), an arbitrary model fine-tuned with LoRA approximates the target model. We theoretically prove that, under certain assumptions, LLMs are fine-tuned with LoRA approximate arbitrary classifiers. Theorem 1 supports the idea that LBC has a high generalization performance in tabular data classification. The proof of the theorem is in Appendix B

Theorem 1 Let $f(\boldsymbol{x})$ represent the ReLU neural network to which LoRA is applied, with no activation function in the last layer, and $\bar{f}(\boldsymbol{x})$ represents the target single-layer linear network. Let $g(x)$ is the logistic function $(1+e^{-x})^{-1}$. $\sigma(\boldsymbol{W})_i$ is the i-th greatest singular value of $\boldsymbol{W}$. $\boldsymbol{W}_l$ and $\overline{\boldsymbol{W}}$ are l-th layer weight matrix of the frozen model and the weight matrix of the target model, respectively. where R_l, R_E are $Rank(W_l), Rank(\overline{\boldsymbol{W}} - \prod \boldsymbol{W}_l)$, respectively. L is the number of layers in f.

4.4 Verbalizer

Language generative models were adapted for classification tasks by utilizing verbalizers in the loss function. During the training process, using verbalizers encourages the model to generate semantically accurate responses rather than comparing labels precisely at the token level. Since this approach does not fit the model to fixed token-level labels, we can expect faster convergence when training generative language models for classification problems. LBC slightly modifies the structure of traditional LLMs in training and inference to use a verbalizer.

Given a vector $\textbf{Logit} = \{l_{w_1}, l_{w_2}, \ldots, l_{w_V}\}$, where V is the vocabulary size and l_{w_i} is the score for the word w_i in the vocabulary, LBC's score for a single class C_k is calculated as follows:

$$\text{Score}(C_k) = \alpha_1 l_k + \alpha_2 \sum_{w \in S_k} l_w$$

Where k is the central word representing class C_k, α_1 and α_2 are the hyperparameters for the central word and synonyms, and S_k is the set of synonyms of central word k. For example, if k = 'Yes', then $S_k = \{$'yes', 'yeah', 'true'...$\}$. The probability for C_k is computed using a softmax function:

$$P(C_k) = \frac{\exp(\text{Score}(C_k))}{\sum_{k' \in K} \exp(\text{Score}(C_{k'}))}$$

where K is the set of central words of all classes. Besides, we modify the existing loss function as follows:

$$J = \alpha_1 \text{CE}(\textbf{Logit}, L_k) + \alpha_2 \sum_{w \in S_k} \text{CE}(\textbf{Logit}, L_w)$$

5 Experiments

5.1 Experiment Settings

We conducted experiments using reliable datasets that have been frequently used in studies, specifically selecting those that have been run multiple times on OpenML (Vanschoren et al., 2013), Kaggle, or other benchmarks. Information about the eleven datasets is in Table 5. Details on the evaluation methods are in Appendix D. As baselines, we selected five models, referred to as TMLs, which are known for their strong performance in tabular data classification. Details of the TMLs are in Appendix C. Additionally, to assess the performance improvements brought by LBC's three methodologies, we conducted direct comparisons with LIFT's methodology.

5.2 OOV Setting

To experiment with the performance of LBC on OOV tasks, it is essential to create scenarios where variables that do not exist in training appear in testing. However, we faced a problem because no existing tabular datasets fulfill this requirement. We randomly deleted 50% of the variable columns in the original tabular dataset. As a result, variables that are deleted become OOV, not learned by the model during training, and emerge as new variables in the test. This allows for the assessment of LBC's ability to interpret OOVs. We compare the performance of TMLs and LBC with the data generated by this method.

5.3 Avoiding Bias

When fine-tuning LBC, if prompts consistently end with the same token, such as a question mark, the model may focus more on that token than on the actual variables when predicting class labels. This issue is particularly pronounced in datasets with class imbalance. To mitigate this, inserting random words at the end of the sentence helps reduce bias towards specific tokens. An example of the use of random words is shown in figure 2.

6 Results

6.1 Performance in OOV tasks

Table 1 presents the accuracy, F1, and AUC scores of TMLs and LBCs on eight binary classification datasets after conducting 50% OOV conversion. In the average rows for the evaluation metrics, LBC consistently outperforms the five TMLs in binary classification problems. Building on these results, we extended our experiments to multiclass classification tasks, as shown in Table 2. LBCs continue to outperform TMLs, with LBC-LLaMA3 demonstrating strong performance in multiclass scenarios.

Table 1: LBC vs. TMLs in binary classification problems with 50% randomly selected OOV situations. The models are trained with 50% IVs, and LBCs add 50% OOVs in the test prompts. LBC outperforms the TMLs on evaluation scores.

Accuracy	DT	KNN	LogReg	SVM	XGBoost	LBC - GPTJ	LBC - Llama3
Blood	72.67	69.33	75.33	75.33	74.67	**76.00±0.00**	**76.00±0.38**
Breast Cancer	93.86	93.86	92.98	92.98	92.98	94.15±1.01	**94.44±0.50**
Creditcard	76.81	73.91	72.46	77.54	76.09	**83.81±0.42**	80.84±0.54
German	71.00	71.50	77.50	71.50	70.50	**78.50±0.86**	77.16±1.15
ILPD	70.94	60.68	72.65	70.94	64.86	**75.05±0.84**	72.07±0.49
Loan	69.11	66.67	69.92	69.11	59.35	80.59±1.22	**81.25±0.00**
Salary	**85.00**	83.00	83.00	81.50	83.00	84.00±0.86	84.67±0.28
Steel Plate	80.21	79.69	73.78	78.15	81.23	81.83±1.62	**81.91±1.47**
Avg.	77.53	74.83	77.18	75.01	76.38	**81.74±0.85**	80.98±0.60

F1	DT	KNN	LogReg	SVM	XGBoost	LBC - GPTJ	LBC - Llama3
Blood	0.68	**0.73**	0.68	0.63	**0.73**	0.67±0.00	0.67±0.00
Breast Cancer	**0.94**	**0.94**	0.93	0.93	0.93	0.93±0.00	0.93±0.00
Creditcard	0.67	0.59	0.62	0.62	0.67	**0.87±0.02**	0.81±0.01
German	0.73	0.77	0.77	0.73	0.78	0.71±0.01	**0.78±0.01**
ILPD	**0.76**	0.71	0.73	0.74	0.75	0.75±0.00	0.75±0.00
Loan	0.70	0.70	0.71	0.70	0.69	0.76±0.01	**0.78±0.01**
Salary	0.55	0.55	0.55	0.5	**0.59**	0.52±0.01	0.52±0.01
Steel Plate	0.8	0.79	0.72	0.79	**0.81**	0.80±0.01	0.80±0.01
Avg.	0.72	0.71	0.70	0.68	0.74	0.75±0.00	**0.76±0.01**

AUC	DT	KNN	LogReg	SVM	XGBoost	LBC - GPTJ	LBC - Llama3
Blood	0.67	0.61	0.67	**0.68**	**0.68**	0.67±0.00	0.67±0.00
Breast Cancer	0.97	0.98	0.98	**0.99**	**0.99**	**0.99±0.00**	**0.99±0.00**
Creditcard	0.79	0.8	0.83	0.84	0.80	**0.92±0.02**	0.85±0.01
German	0.67	0.69	**0.80**	0.67	0.69	**0.79±0.01**	0.78±0.01
ILPD	0.71	0.57	0.68	0.71	0.71	**0.75±0.01**	**0.75±0.00**
Loan	0.56	0.57	0.63	0.51	0.53	**0.79±0.01**	0.77±0.01
Salary	0.84	0.85	0.86	0.87	0.86	**0.88±0.01**	**0.88±0.01**
Steel Plate	0.87	0.89	0.89	0.89	0.89	**0.90±0.00**	**0.89±0.00**
Avg.	0.76	0.73	0.78	0.78	0.78	**0.84±0.00**	0.82±0.00

Table 2: LBC vs. TMLs in multiclass classification problems with 50% randomly selected OOV situations. LBC also outperforms the TMLs on evaluation scores in multiclass classification.

Accuracy	DT	KNN	LogReg	XGBoost	LBC - GPTJ	LBC - Llama3
CMC	46.10	43.39	48.15	45.42	49.71±0.78	**51.75±1.36**
Restaurant	79.50	83.50	80.50	84.50	81.16±0.57	**85.33±0.57**
OGB	50.00	51.50	55.00	56.50	56.73±1.44	**62.51±0.79**
Avg.	58.53	59.46	61.22	62.14	62.53±0.93	**66.53±0.91**

F1	DT	KNN	LogReg	XGBoost	LBC - GPTJ	LBC - Llama3
CMC	0.47	0.40	0.47	0.45	0.50±0.01	**0.51±0.01**
Restaurant	0.79	0.85	0.75	0.85	0.82±0.01	**0.86±0.01**
OGB	0.34	0.51	0.54	0.56	0.54±0.01	**0.57±0.02**
Avg.	0.53	0.59	0.59	0.62	0.62±0.01	**0.64±0.01**

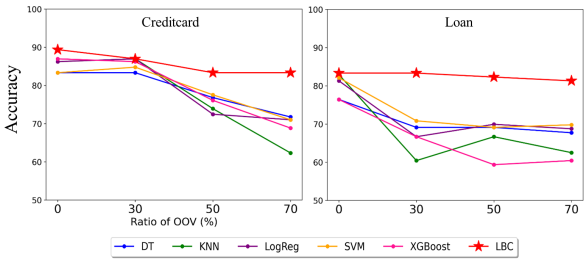

Figure 3: Graph of accuracy changing over OOV ratio (%): We observed the accuracy change of TMLs and LBCs by increasing the OOV ratio from 0, 30, 50, and 70 (%) for two datasets. Comparing the accuracy reduction of TMLs and LBCs, the reduction of LBCs is smaller compared to TMLs. It demonstrates that LBCs interpret OOVs, unlike TMLs.

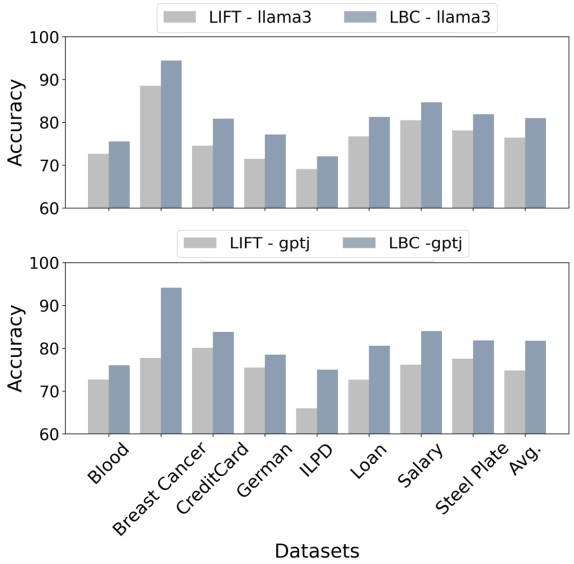

Figure 4: LIFT vs LBC in 50% randomly selected OOV situation. Both LLMs have a performance improvement when LBC's methodologies are applied rather than LIFT.

Table 7 provides the statistical test results on the Accuracy scores from Table 1 and Table 2. For each dataset, a T-test was conducted between the model with the highest performance among LBCs and the best-performing TML. The null hypothesis, H_0 : $\text{Accuracy}_{\text{LBC-best}} = \text{Accuracy}_{\text{TML-best}}$, was rejected for seven out of eleven datasets, with a p-value less than 0.05 used as the criterion for rejection. This provides empirical evidence that LBC effectively utilizes pre-trained knowledge to make accurate interpretations in OOV situations. Further analysis of this capability is discussed in Section 6.2.

Figure 4 shows how much LBC's methodologies improve the performance of LLM on the OOV task. There is a significant difference in performance between using LBC and using LIFT, which does not incorporate LBC's three methodologies. This demonstrates that, in addition to the advantage of LLM's pre-trained knowledge in interpreting OOVs, LBC's methodologies have a clear and positive impact on performance.

To validate the ability of LBC to perform well on OOV tasks, we conduct experiments on two datasets with different OOV ratios. In each dataset, we vary the OOV ratio to 0%, 30%, 50%, and 70% and observe the model's accuracy change. Figure 3 shows that for TMLs, the performance decreases significantly as the OOV ratio increases. In contrast, LBC shows no decrease in accuracy as the OOV ratio increases, or the decrease is small compared to TMLs. These findings suggest that LBC can effectively utilize the pre-trained knowledge of LLMs to outperform traditional ML methods even

as the percentage of OOVs increases.

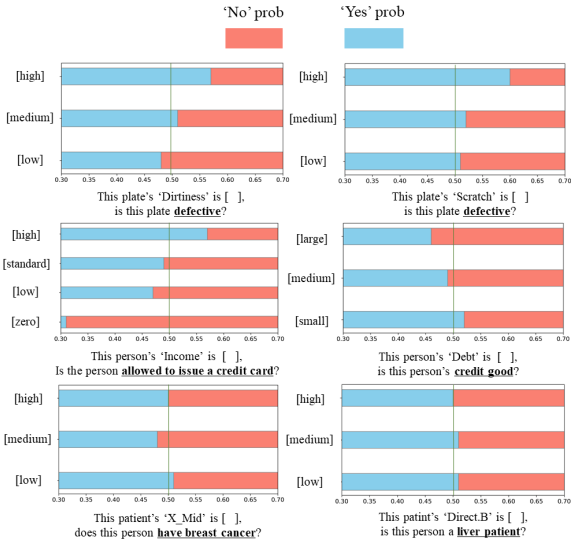

Figure 5: Observing how LBC applies its pre-trained knowledge to prompts about OOVs, thereby revealing biases in its pre-trained knowledge. Intuitively, LBC has a bias toward making its predictions closer to the correct answer. However, it is not responsive to special variable names that do not have a word meaning.

6.2 LBC's Ability to utilize Pre-Trained Knowledge

In this section, we specifically investigate how LBC uses their pre-trained knowledge to interpret OOVs in the OOV tasks. We conducted an experiment to observe the pre-trained bias for variables using an LBC that was trained in the structure of data without any information about the variables. For datasets with a "Yes" or "No" answer, the structure of the data is as follows:

$$\begin{aligned}\text{Prompt} &= \text{string(Start of sentence)}+\\ &\quad\ \text{string('Variable name' is [Variable value])}+\\ &\quad\ \text{string(Question\#\#\#)}\\ \text{Answer} &= \text{Yes@@@ or No@@@}\end{aligned}$$

In the training process, the prompt structure is utilized as it is, with experimental adjustments made to balance the likelihood of the trained LBC predicting 'Yes' or 'No.' During testing, we replace the names and values of various variables in the 'variable name' and 'variable value' placeholders within the prompts to evaluate the pre-trained biases of LBC towards those variables.

Figure 5 presents the outcomes for several variables of high importance in this experiment. It is evident that LBC leverages pre-trained knowledge to approximate the probabilities for variables not learned during training closer to the correct answers. Notably, the interpretation that the graph for "income" shows a high risk of issuing a credit card to an individual with an income of "0" matches with the actual distribution in the Creditcard dataset. However, for the unique variables in the table dataset, such as "Direct.B", which does not have the meaning of a common word, LBC shows almost balanced results and tends to make predictions without any clear bias. This shows that LBC maintains a neutral approach to uninterpretable variables and an even probability distribution without any particular tendency. These results support the high performance of LBC in handling OOV tasks.

Additionally, we experiment with the same conditions as TMLs without providing the LBC with any information about OOVs. Using the Creditcard dataset with the same settings as table 1, we feed the LBC with test prompts without OOV information and measure its performance. The results in table 3 show that the LBC's performance on prompts without OOV information was significantly lower than when with the information, highlighting the fact that the LBC gets information from OOVs.

Table 3: The performance of LBC w/o OOV. Comparing performance w/o and w/ OOVs shows that LBC effectively utilizes OOVs to improve performance.

Metric	Xgboost	LBC w/o OOV	LBC w/ OOV
Accuracy	76.09	74.63 ± 0.90	**83.81 ± 0.42**
F1	0.67	0.72 ± 0.01	**0.87 ± 0.02**
AUC	0.80	0.81 ± 0.00	**0.92 ± 0.02**

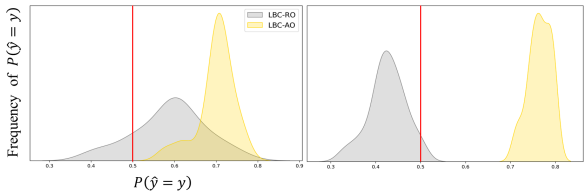

Figure 6: Frequeny of $P(\hat{y} = y)$ for the two prompt generation methods. We repeated the prompt generation 100 times for each of the two randomly selected examples from the Creditcard dataset in two ways: random order (RO) and advanced order (AO). The horizontal axis represents the model's probability for the correct class y, and the vertical axis shows frequency. The AO method provides more consistent and accurate results than RO. The red vertical line indicates the prediction boundary, where the differences between the two methods lead to varied predictions.

6.3 Importance of Advanced Prompt

In this section, we examine how Advanced Prompts, such as "Consider the order of vari-

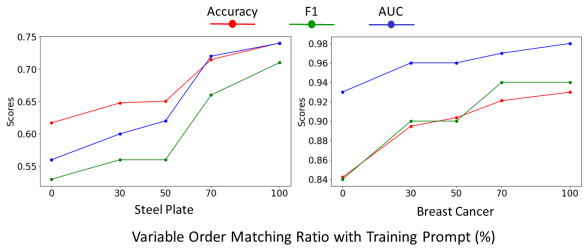

Figure 7: The changes in scores according to the ratio of IVs in the test prompt that maintains the same order as the IVs in the training prompt. Both the training and test prompts consist of only the IVs used in table 1. As the ratio of the same IV order increases between the training and test prompts, all scores improve. This demonstrates the importance of applying the same order of IVs in the test prompt as in the training prompt.

ables" or "Add an indicator," used to generate test prompts, affect LBC's probability output.

To verify the importance of the variables' order, we experiment with repeatedly generating two types of prompts by randomly selecting an instance from the tabular data: One, where the order of all variables is randomized (LBC-RO), and the other, where the order of the IVs matches to the IV of the training data, and only the order of the OOVs are randomized (LBC-AO). We randomly select two instances from the Creditcard dataset and generate 100 different prompts for each instance with the RO and AO methods, respectively, to compare the probability distributions generated by LBC for the two methods. Figure 6 illustrates the performance difference between prompts where the order of variables is matched with the training data and those where it is not. LBC-RO exhibits a large variance in the probability distribution, leading to variations in the model's predictions for a single data instance. In contrast, LBC-AO shows a small variance in the probability distribution, which means that the model makes consistent predictions.

To further investigate the benefits of matching the order of IVs of test prompts with the training prompts, we compose the training and test data using only the IVs, excluding the OOVs selected from the Steel Plate dataset used in Table 1. Then, for the variables that make up the test prompt, we experiment with increasing the ratio of variables in the same order as the variable order of the training prompt to check the scores for the three evaluation metrics. Figure 7 illustrates the scores for the three evaluation metrics. As the IVs ratio increases, the performance improves on all three metrics. This shows that LBC performs best when the test data follows the same variable order as the training data.

Table 4: Comparison of three performance metrics before and after applying LBC's methodology using in-context learning to a modern black-box LLM

Accuracy	Creditcard	German	ILPD	Loan	Avg.
GPT3.5	60.15	63.50	62.75	63.54	62.50
LBC-GPT3.5	**69.57**	**67.50**	**63.25**	**66.67**	**66.74**

F1	Creditcard	German	ILPD	Loan	Avg.
GPT3.5	0.61	0.55	0.57	0.60	0.58
LBC-GPT3.5	**0.69**	**0.57**	**0.61**	**0.66**	**0.63**

AUC	Creditcard	German	ILPD	Loan	Avg.
GPT3.5	0.60	0.52	0.51	0.55	0.54
LBC-GPT3.5	**0.69**	**0.57**	**0.54**	**0.59**	**0.60**

6.4 LBC - Black-box LLM

Although it is possible to configure LBC using the latest LLM, most of the latest models are black-box, so we conduct in-context learning experiments. In the demo, ten positive and negative examples are provided in equal measure to demonstrate the model's ability to generalize from a balanced dataset. The model with the LBC methodology incorporates categorical change, advanced order, and indicator methodologies. Verbalizer is excluded due to logits being inaccessible. Table 4 compares the Accuracy, F1, and AUC scores before and after applying the LBC methodology. We use GPT-3.5 as the model, and performance improves significantly on all datasets when we add the LBC methodology. This demonstrates that the LBC methodology can also be applied to black-box LLMs to improve performance. The performance on its own is not high because of a small number of training examples for in-context learning compared to fine-tuning, and it is considered to be better when it is fine-tuned.

7 Conclusion

In this work, we propose LBC to solve OOV tasks. LBCs utilize prompt-based inference, which allows information about OOVs to be added to prompts in a straightforward way and enables understanding of the new information through pre-trained knowledge. LBC's three methodologies maximize the above advantages to achieve high performance on OOV tasks. As a result, utilizing LLMs' pre-trained knowledge is a key strategy for solving the OOV task, and we plan to combine it with various statistical methods. LBC is the first approach to apply pre-trained LLM to OOV tasks.

8 Limitations

Based on our three methodologies, LBC demonstrates superior performance over TML in addressing the OOV generalization problem, leveraging pre-trained knowledge and the contextual understanding capabilities of LLMs. However, several limitations still exist. The first limitation is the potential presence of data that requires knowledge not covered in pre-training. When column names are unintelligible or involve extremely recent information not included in pre-training, LBC faces difficulties in interpretation. The second limitation is that LBC requires more resources than TML. In terms of training time and GPU specifications, LBC demands higher costs than TML. Therefore, in cases where the information content of OOV is low or when the problem does not involve OOV, LBC is less suitable compared to TML.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Massimo Buscema, Stefano Terzi, and William Tastle. 2010. Steel plates faults. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5J88N.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. 2022. Lift: Language-interfaced fine-tuning for non-language machine learning tasks. *Advances in Neural Information Processing Systems*, 35:11763–11784.

Kris K. Dreher, Leonardo Ayala, Melanie Schellenberg, Marco Hübner, Jan-Hinrich Nölke, Tim J. Adler, Silvia Seidlitz, Jan Sellner, Alexander Studier-Fischer, Janek Gröhl, Felix Nickel, Ullrich Köthe, Alexander Seitel, and Lena Maier-Hein. 2023. *Unsupervised Domain Transfer with Conditional Invertible Neural Networks*, page 770–780. Springer Nature Switzerland.

Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR.

Siyuan Guo, Jonas Wildberger, and Bernhard Schölkopf. 2023. Out-of-variable generalization. *arXiv preprint arXiv:2304.07896*.

Siyuan Guo, Jonas Wildberger, and Bernhard Schölkopf. 2024. Out-of-variable generalization for discriminative models. *Preprint*, arXiv:2304.07896.

Hans Hofmann. 1994. Statlog (German Credit Data). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5NC77.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2022. Tabpfn: A transformer that solves small tabular classification problems in a second. *arXiv preprint arXiv:2207.01848*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Shengding Hu, Ning Ding, Huadong Wang, Zhiyuan Liu, Jingang Wang, Juanzi Li, Wei Wu, and Maosong Sun. 2022. Knowledgeable prompt-tuning: Incorporating knowledge into prompt verbalizer for text classification. *Preprint*, arXiv:2108.02035.

Rabie El Kharoua. 2024a. Predict online gaming behavior dataset.

Rabie El Kharoua. 2024b. Predict restaurant menu items profitability.

Ron Kohavi. 1996. Census Income. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5GP7S.

Tjen-Sien Lim. 1997. Contraceptive Method Choice. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C59W2D.

Mazaharul Hasnine Mirza. 2023. Loan data set.

J. R. Quinlan. Credit Approval. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5FS30.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Bendi Ramana and N. Venkateswarlu. 2012. ILPD (Indian Liver Patient Dataset). UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5D02C.

Timo Schick and Hinrich Schütze. 2020. Exploiting cloze questions for few shot text classification and natural language inference. *arXiv preprint arXiv:2001.07676*.

Timo Schick and Hinrich Schütze. 2021. It's not just size that matters: Small language models are also few-shot learners. *Preprint*, arXiv:2009.07118.

Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. 2015. Simultaneous deep transfer across domains and tasks. *CoRR*, abs/1510.02192.

Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. 2013. Openml: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Jiahuan Yan, Bo Zheng, Hongxia Xu, Yiheng Zhu, Danny Z. Chen, Jimeng Sun, Jian Wu, and Jintai Chen. 2024. Making pre-trained language models great on tabular prediction. *Preprint*, arXiv:2403.01841.

I-Cheng Yeh. 2008. Blood Transfusion Service Center. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5GS39.

Yuchen Zeng and Kangwook Lee. 2023. The expressive power of low-rank adaptation. *arXiv preprint arXiv:2310.17513*.

Matjaz Zwitter and Milan Soklic. 1988. Breast Cancer. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C51P4M.

A Hyperparameters for Experiments

The hyperparameters for our experiments were set as follows: Learning rate in {1e-3, 1e-4, 1e-5}, LoRA rank in {8, 16, 48, 144, 196}, Epoch in {5, 7, 10, 12}. We conduct the grid search over those hyperparameters. Verbalizer α_1 in {0.6, 0.7, 0.8, 0.9}

B Proof of Theorem 1

According to (Zeng and Lee, 2023), an arbitrary model fine-tuned with LoRA approximates the target model. We extend this theory and theoretically prove that, under certain assumptions, LLMs are fine-tuned with LoRA approximate arbitrary classifiers. Theorem 1 supports the idea that LBC has a high generalization performance in tabular data classification.

Lemma 1 *The logistic function* $g(x) = (1 + e^{-x})^{-1}$ *is Lipschitz continuous with a Lipschitz constant of* $1/4$.

Proof of Lemma 1 A function $f : \mathbb{R} \to \mathbb{R}$ is Lipschitz continuous if

$$\exists K > 0, \forall x_1, x_2 \in \mathbb{R}, |f(x_1) - f(x_2)| \leq K|x_1 - x_2|. \tag{1}$$

By substituting f with g, and considering that g is a monotonic function, we can obtain the following expression:

$$\frac{g(x_1) - g(x_2)}{x_1 - x_2} \leq K.$$

By the mean value theorem,

$$g'(c) = \frac{g(x_2) - g(x_1)}{x_2 - x_1} \leq K, \text{and}$$
$$0 < g'(c) \leq \frac{1}{4}$$
$$(g'(c) = g(c)(1 - g(c)) \wedge 0 < g(c) < 1)$$
$$\to K \geq \frac{1}{4}.$$

A new theorem, which is a variation of Lemma 11 of (Zeng and Lee, 2023), can be proposed using Lemma 1 above.

Theorem 1. Let $f(\boldsymbol{x})$ represents the ReLU neural network to which LoRA be applied, with no activation function in the last layer, and $\bar{f}(\boldsymbol{x})$ represents the target single-layer linear network. Let $g(x)$ is the logistic function $(1 + e^{-x})^{-1}$. $\sigma(\boldsymbol{W})_i$ is the i-th greatest singular value of $\boldsymbol{W}$. $\boldsymbol{W}_l$ and $\overline{\boldsymbol{W}}$ are l-th layer weight matrix of the frozen model and the weight matrix of the target model, respectively.

$$\mathbb{E}\left\|g(f(\boldsymbol{x})) - g(\bar{f}(\boldsymbol{x}))\right\|_2^2$$
$$\leq \frac{1}{16}\mathbb{E}\left\|(f(\boldsymbol{x}) - \bar{f}(\boldsymbol{x}))\right\|_2^2$$
$$\text{(g is } 1/4 \text{ Lipschitz by Lemma 1)}$$
$$\leq \frac{1}{16}\|\mathbb{E}(\boldsymbol{x}\boldsymbol{x}^T)\|_{\mathrm{F}}\,\sigma^2\Big(\overline{\boldsymbol{W}} - \prod \boldsymbol{W}_l\Big)_{\min(\sum_{l=1}^{L} R_l, R_E)+1}.$$

where R_l, R_E are $Rank(W_l), Rank(\overline{\boldsymbol{W}} - \prod \boldsymbol{W}_l)$, respectively. L is the number of layers in f.

C Traditional Machine Learning Models

For Traditional Machine Learning Models, we selected 5 models. For tree-based models, we chose Decision Tree and XGBoost. Tree-based models have strong performance in tabular data classification. We also included K-Nearest Neighbor, Logistic Regression, and Support Vector Machine to increase the diversity of the models.

Decision Tree. A Decision Tree (DT) is a model used for classification and regression tasks. The model trains on data to make predictions based on simple decision rules. The advantage of decision trees is that they capture non-linear patterns in data, and the results of the model are easy to interpret.

K-Nearest Neighbor The K-Nearest Neighbor (KNN) algorithm is used in classification and regression to make predictions based on the data of the K closest neighbors.

Logistic Regression Logistic regression (LogReg) is a model often used for classification problems. This model is often used when the outcome is binary, and it estimates probabilities to perform classification based on decision boundaries.

Support Vector Machine A Support Vector Machine (SVM) is a machine learning model used for classification and regression problems. The model finds the optimal decision boundary to divide a given set of data into categories, and in this study, we used a Radial Basis Function (RBF) kernel.

XGBoost XGBoost is a high-performance machine learning model based on the Gradient Boosting algorithm, which is a decision tree-based ensemble learning method that combines multiple tree models. At each step, XGBoost adds a new model to reduce the error of the previous model and uses the Gradient Boosting technique in the

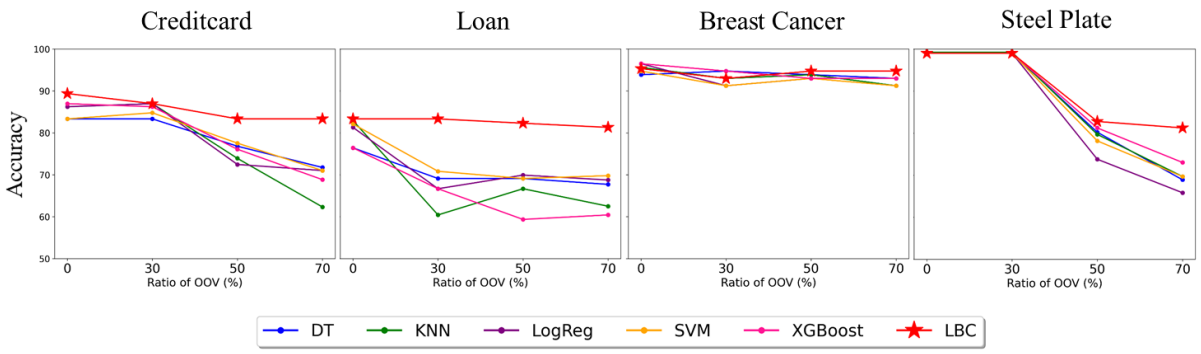

Figure 8: Graph of accuracy changing over OOV ratio (%): We observed the accuracy change of TMLs and LBCs by increasing the OOV ratio from 0, 30, 50, and 70 (%) for four datasets. Comparing the accuracy reduction of TMLs and LBCs, the reduction of LBCs is smaller compared to TMLs. It demonstrates that LBCs interpret OOVs, unlike TMLs.

process. XGBoost is a state-of-the-art model that is used on many benchmarks.

All 5 models were imported and used from scikit-learn. We also used scikit-learn's HalvingGridSearchCV class to explore the optimal hyperparameters.

D Evaluation Methods

Accuracy measures the proportion of correct predictions and is defined as Accuracy $= \frac{n_{correct}}{n_{samples}}$. Here, $n_{correct}$ is the number of correct predictions, and $n_{samples}$ is the total number of samples. **F1 score**, a harmonic mean of Precision and Recall, is calculated as F1 score $= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$, where Precision $= \frac{TP}{TP+FP}$ and Recall $= \frac{TP}{TP+FN}$.
AUC score represents the area under the ROC curve, which plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings.

Table 5: Dataset Statistics

Dataset	#Variable	#Class	#Instance
Blood (Yeh, 2008)	4	2	583
Breast Cancer (Zwitter and Soklic, 1988)	31	2	569
Creditcard (Quinlan)	15	2	690
German Credit (Hofmann, 1994)	20	2	1000+
ILPD (Ramana and Venkateswarlu, 2012)	11	2	583
Loan (Mirza, 2023)	10	2	615
Salary (Kohavi, 1996)	14	2	1000+
Steel Plate (Buscema et al., 2010)	34	2	1000+
CMC (Lim, 1997)	9	3	1000+
OGB (Kharoua, 2024a)	13	3	1000+
Restaurant (Kharoua, 2024b)	6	3	1000+

E Selecting Pre-trained LLM

Our research focuses not merely on prompt tuning using LLMs but on modifying the structure itself to construct a model that demonstrates high performance in classification. Specifically, one of our methodologies involves a verbalizer that requires direct access to the model's loss function and vocabulary. Therefore, we need to choose a powerful yet completely open-source LLM. Hence, we selected GPT-J 6B (Wang and Komatsuzaki, 2021) model and LLaMA-3 8B model. Both models exhibit strong performance in inference based on extensive pre-trained knowledge and have the advantage of being fully open-source. Additionally, we further validated our approach using black-box models such as GPT-3.5.

Dataset	LBC-LLaMA3 w/o C.C	LBC-LLaMA3
Creditcard	78.76	80.84
Loan	79.75	81.25
OGB	60.83	62.51
Steel Plate	79.69	80.91

Table 6: Compare the accuracy scores of the LBC-LLaMA3 model with and without Categorical Change (C.C.) across four datasets. The accuracy scores are higher when using Categorical Change, supporting the idea that LLMs interpret categorical variables better than numerical ones.

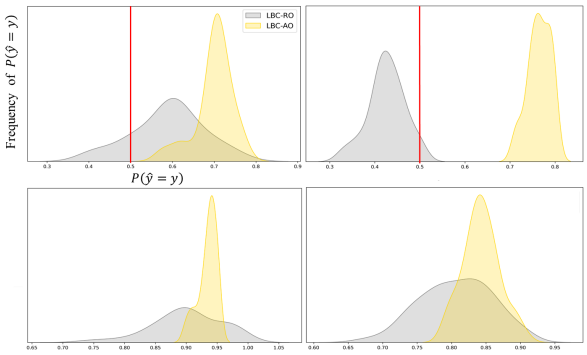

Figure 9: Frequeny of $P(\hat{y} = y)$ for the two prompt generation methods. We repeated the prompt generation 100 times for each of the four randomly selected examples from the Creditcard dataset in two ways: random order (RO) and advanced order (AO). The horizontal axis represents the model's probability for the correct class y, and the vertical axis shows frequency. The AO method provides more consistent and accurate results than RO. The red vertical line indicates the prediction boundary, where the differences between the two methods lead to varied predictions.

Table 7: Accuracy evaluation of the proposed models. We perform five repeated experiments on the model with the highest performance among the TML and LBC methods and conduct a t-test. The two columns on the left represent the mean accuracy of the repeated experiments. The p-values less than 0.05 are highlighted in bold and marked with an asterisk (*). For seven of the eleven datasets, it is valid that LBC outperforms TML.

Datasets	LBC-Best	TML-Best	T-stats	P-value
Blood	76.00	75.55	1.71	0.12
Breast Cancer	94.44	93.27	2.94	**0.01***
Creditcard	83.81	77.37	6.24	**0.00***
German	78.50	76.66	3.20	**0.03***
ILPD	75.05	72.19	3.43	**0.02***
Loan	81.25	70.27	18.72	**0.00***
Salary	84.67	84.83	-0.90	0.39
Steel Plate	81.91	80.51	1.89	0.09
CMC	51.75	47.88	6.11	**0.00***
OGB	62.51	57.16	3.03	**0.03***
Restaurant	85.33	84.66	0.60	0.57