From Supervised to Generative: A Novel Paradigm for Tabular Deep Learning with Large Language Models

Xumeng Wen
Microsoft Research Asia
Beijing, China
xumengwen@microsoft.com

Han Zhang*
Tsinghua University
Beijing, China
zh950713@gmail.com

Shun Zheng
Microsoft Research Asia
Beijing, China
shun.zheng@microsoft.com

Wei Xu
Tsinghua University
Beijing, China
weixu@tsinghua.edu.cn

Jiang Bian
Microsoft Research Asia
Beijing, China
jiang.bian@microsoft.com

ABSTRACT

Tabular data is foundational to predictive modeling in various crucial industries, including healthcare, finance, retail, sustainability, etc. Despite the progress made in specialized models, there is an increasing demand for universal models that can transfer knowledge, generalize from limited data, and follow human instructions. These are challenges that current tabular deep learning approaches have not fully tackled. Here we introduce Generative Tabular Learning (GTL), a novel framework that integrates the advanced functionalities of large language models (LLMs)—such as prompt-based zero-shot generalization and in-context learning—into tabular deep learning. GTL capitalizes on the pre-training of LLMs on diverse tabular data, enhancing their understanding of domain-specific knowledge, numerical sequences, and statistical dependencies critical for accurate predictions. Our empirical study spans 384 public datasets, rigorously analyzing GTL's convergence and scaling behaviors and assessing the impact of varied data templates. The GTL-enhanced LLaMA-2 model demonstrates superior zero-shot and in-context learning capabilities across numerous classification and regression tasks. Notably, it achieves this without fine-tuning, outperforming traditional methods and rivaling state-of-the-art models like GPT-4 in certain cases. Through GTL, we not only foster a deeper integration of LLMs' sophisticated abilities into tabular data comprehension and application but also offer a new training resource and a test bed for LLMs to enhance their ability to comprehend tabular data. To facilitate reproducible research, we release our code, data, and model checkpoints at https://github.com/microsoft/Industrial-Foundation-Models.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; **Modeling methodologies**; *Transfer learning*.

*Han did this work during his internship at Microsoft Research Asia, Beijing, China.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08.
https://doi.org/10.1145/3637528.3671975

KEYWORDS

tabular data, large language models, generative modeling, instruction following, in-context learning, zero-shot learning

ACM Reference Format:
Xumeng Wen, Han Zhang, Shun Zheng, Wei Xu, and Jiang Bian. 2024. From Supervised to Generative: A Novel Paradigm for Tabular Deep Learning with Large Language Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3637528.3671975

1 INTRODUCTION

Tabular data, with its widespread presence across numerous critical industrial domains such as healthcare, finance, retail, sustainability, and climate [7, 13, 29, 32], serves as a cornerstone for predictive modeling [19]. This modeling underpins a diverse array of real-world applications, including disease risk stratification, credit assessment, sales volume prediction, grid stability measurement, climate estimation, etc. Given its significance, it has attracted substantial research attention from the machine learning community.

While there have been considerable advancements in developing specialized predictive models for individual tasks within tabular data, employing effective learning models such as tree ensemble models [7, 18, 28] based on gradient boosting decision trees [9] and more recent neural networks [2, 11, 16, 17], the rich and diverse application scenarios across various domains have underscored a critical need for universal models. These models should possess the ability to seamlessly transfer to new datasets and exhibit robust generalization capabilities, especially in scenarios with few-shot labels. This necessity has catalyzed the emergence of tabular deep learning [11] as a research focus, which explores the concept of pre-training a universal neural network on a broad spectrum of tabular data. This approach aims to create models that, once pre-trained, can be effortlessly adapted to a wide range of tasks, thereby enhancing their utility and efficiency across different applications [3, 14, 21, 25, 33, 36, 40, 42, 44, 46].

The rapidly evolving foundation models in language and multimodal domains have illuminated the potential for pre-training universal deep models for tabular data, showcasing remarkable generalization capabilities [5, 15, 26, 34]. Specifically, these foundation models highlight two key attributes: Firstly, upon pre-training, they can rapidly adapt to new tasks simply by specifying a task-oriented

prompt [27]. This form of adaptation is not only user-friendly but also critically enhances the model's ability to transfer knowledge acquired during pre-training to novel, low-resource scenarios efficiently. Secondly, in situations involving few-shot labels, these models exhibit an intriguing in-context learning ability, evidenced by their impressive capacity to learn from demonstrations included as part of the prompt [5]. This capability, without fine-tuning, implicitly leverages Bayesian inference [41] to extend the model's pre-trained knowledge to new data samples directly during forward propagation. Together, these attributes suggest a promising direction for advancing tabular deep learning by integrating the adaptability and learning efficiencies found in foundation models.

But most existing pre-training methods for tabular data predominantly adhere to the supervised paradigm, necessitating a subsequent fine-tuning process for adaptation to new tabular tasks. Despite pioneering efforts to transcend these limitations through advanced generalization capabilities, i.e., zero-shot learning to unseen tasks [13, 40, 42] and in-context learning based on fresh tabular instances [8, 14], the outcomes often encapsulate constrained generalization abilities and a limited scope of application. For example, LIFT [8], comparing fine-tuning with in-context learning on six classifications, uses a narrow evaluation scope by concentrating exclusively on accuracy, which may not capture the model's comprehensive generalization capability and its efficacy in diverse scenarios. Similarly, TabPFN [14], which develops a prior-data fitted network [24] and primarily caters to classification datasets with numerical features, exhibits a limitation by not accommodating zero-shot inference. Additionally, while TabLLM [13], UniTabE [42], and TransTab [40] have showcased successful zero-shot demonstrations, primarily in a handful of classification tasks, they often neglected in-context learning. Instead, they depend on fine-tuning for adapting models to downstream tasks, missing opportunities for more nuanced data-driven adaptation. The limited scope of these models underscores the demand for a more holistic tabular deep learning paradigm that not only embraces but also extends the advanced generalization capabilities found in other cutting-edge foundation models.

Nonetheless, integrating the sophisticated capabilities of modern large language models (LLMs) [5, 26] - specifically, prompt-based zero-shot generalization and in-context learning — into tabular deep learning presents significant challenges. A primary issue is that LLMs, primarily pre-trained on language data, often struggle to fully grasp the nuances of language-formatted tabular data. Despite their proficiency in acquiring broad world knowledge and advanced reasoning skills from textual corpora, these models frequently lack the ability to understand domain-specific knowledge that is crucial for effective tabular deep learning. This shortfall is particularly evident in their handling of numerical features represented as token sequences and in identifying the intricate statistical dependencies that exist between prediction targets and tabular features, as well as those between in-context demonstrations and data samples for prediction. These competencies are essential for comprehending the unique characteristics of tabular datasets and leveraging them for accurate predictive modeling.

To address this shortcoming, we introduce *Generative Tabular Learning* (GTL) - a novel paradigm that advocates for continued pre-training of an LLM on extensive tabular data, transcribed in an instruction-oriented language format, spanning multiple domains. GTL is meticulously designed to foster an enhanced comprehension of tabular features, their interrelations with prediction targets and in-context data examples, and the linkage between task instructions and tabular predictions. To facilitate this process, we create specific text templates to convert a tabular data instance into an instruction-oriented language format. This conversion caters to various configurations, such as including meta-information and in-context examples or not, and maintains a record of the positions of numerical tokens and target tokens. As a result, GTL equips the model with advanced instruction-following abilities tailored for tabular deep learning, enabling it to generatively adapt to downstream tasks by interpreting natural language prompts for new task instructions or in-context examples.

In our experiments, we have compiled 384 public tabular datasets, covering 176 classification and 208 regression tasks across a variety of industrial domains. For holdout evaluation, we randomly selected 20 classification and 24 regression tasks, while all other tasks were utilized for GTL. Besides, we employed LLaMA-2 [35] as our base LLM. Our experimental investigation includes a study of GTL's convergence behaviors, demonstrating effective knowledge transfer from learned datasets to unseen ones across dataset domains. Furthermore, our exploration of GTL's scaling laws reveals key factors that significantly influence the model's generalization capability. Our investigation also encompasses an examination of the performance disparities when employing different data templates to convert tabular data instances into pure text, enabling us to analyze their advantages and drawbacks in various scenarios.

After optimizing the GTL configuration on LLaMA-2, we compared the resulting LLaMA-GTL model with competitive baselines. In stark contrast to traditional tabular models, which require fine-tuning, LLaMA-GTL, utilizing only in-context inference without any parameter updates, achieves state-of-the-art performance in numerous extremely few-shot cases (≤ 32 data samples). Even when the number of shots increases, displaying robust statistical patterns, LLaMA-GTL maintains a performance level comparable to the best performing baseline. Additionally, TabPFN, a representative tabular baseline with in-context learning, currently only supports classification tasks. However, LLaMA-GTL extends its coverage to regression tasks, achieving even greater performance improvements over existing fine-tuning baselines in this area. Further, we compared LLaMA-GTL with GPT-4 [26], perhaps the most advanced LLM to date. Despite the comparison being somewhat disadvantageous for LLaMA-GTL in terms of model size and potential data contamination risk in GPT-4 [4], LLaMA-GTL has surpassed GPT-4 in many classification tasks and significantly narrowed the gap between LLaMA and this proprietary LLM in regression tasks.

Our contributions can be summarized as follows:

- We propose GTL, a generative learning paradigm that extends the advanced capabilities of LLMs—specifically, prompt-based zero-shot generalization and in-context learning—to the realm of tabular deep learning. GTL uniquely enhances the model's understanding of tabular data by continuing the pre-training process on a diverse array of tabular datasets formatted in an instruction-oriented language, addressing the critical gap in current methodologies.

- To support GTL, we have developed a comprehensive data construction pipeline that transforms tabular instances into an instruction-oriented language format. This pipeline not only fosters further research into instruction-following capabilities within tabular deep learning but also enriches the training and evaluation resources available for LLMs, aiming to bolster their comprehension and predictive accuracy with tabular data.
- To demonstrate the efficacy of GTL, we have trained the LLaMA-GTL model, exhibiting its exceptional zero-shot and in-context learning performance across a variety of classification and regression tasks on tabular data. LLaMA-GTL has set new benchmarks in adaptability and generalization for tabular models, surpassing traditional methods and even state-of-the-art LLMs like GPT-4 in specific tasks.

2 RELATED WORK

Our literature review encompasses four aspects.

Predictive Modeling on Tabular Data. The development of effective algorithms for predictive modeling on tabular data has been a longstanding research topic. In the early days, tree-based models were found to be particularly suitable for tabular data, leading to the development of several gradient boosting decision trees [7, 18, 28]. Subsequently, as deep learning gained prominence [20], numerous studies attempted to create suitable network architectures for tabular data [2, 11, 16, 17] and introduced self-supervised learning schemes [3, 33, 36, 44]. Despite these tabular neural networks not consistently outperforming tree-based models [11, 12, 32], further research has advanced tabular deep learning and established new state-of-the-art results in few-shot [14, 25] and transfer learning scenarios [21, 40, 46]. However, these techniques typically required re-training or fine-tuning to adapt to new data schemas and tasks. Moreover, as mentioned in the introduction, while some advancements have introduced support for either zero-shot or in-context learning capabilities, they usually address limited scenarios. In contrast, this paper embarks on a comprehensive exploration of zero-shot and in-context generalization within the sphere of tabular deep learning. Our focus lies in the ease of adaptation without the necessity for parameter tuning, extreme data efficiency, and wide-ranging transferability across diverse domain knowledge, data schemas, and tasks. These are areas that have not been sufficiently explored in the existing literature on tabular deep learning.

When LLMs meet Tabular Deep Learning. The remarkable success of LLMs, scaled to unprecedented sizes and trained on massive text corpus, has demonstrated their broad knowledge and phenomenal capabilities in transfer learning and instruction following [5, 27]. This success has spurred the application of LLMs to tabular deep learning with the aims of 1) leveraging the comprehensive world knowledge already acquired, 2) enabling instruction following to support diverse tasks without the need for tuning or even data, and 3) effectively harnessing meta-information in tabular data, such as column names, task descriptions, and background knowledge. In this domain, several pioneering studies have set the stage. For instance, LIFT [8] introduced language-interfaced fine-tuning, which fine-tuned GPT-3 [5] and GPT-J [37] on multiple tabular learning datasets, revealing that the performance of fine-tuned LLMs was roughly on par with traditional solutions. Specifically, it examined the comparison between fine-tuning and in-context learning, albeit only on six classification tasks using the accuracy metric. TabLLM [13], a subsequent study that adopted T0 [30] as the base LLM, demonstrated competitive performance of fine-tuned LLMs in very few-shot scenarios, and it slightly underperformed in comparison to classical tabular models when more shots were available. TabLLM explored zero-shot learning on tabular data, but it only covered classification tasks, its inference design necessitated multiple forward passes over the base LLM for multi-class tasks, and it did not explore in-context learning. Moreover, MediTab [39], which focused on the healthcare domain, utilized LLMs to generate supplementary data for a specific target task, and TapTap [45] employed LLMs to create synthetic tabular data. Additionally, we have recently identified two contemporaneous studies, TP-BERTa [1] and UniPredict [38], which also investigated the pre-training of LLMs on tabular data. However, these studies still conformed to the supervised learning paradigm for downstream tasks. Distinct from these explorations merging LLMs with tabular deep learning, our research accentuates a generative learning paradigm for LLMs on tabular data, promoting comprehensive instruction-following capabilities for both zero-shot and in-context learning. Furthermore, our approach supports not only binary classification tasks but also multi-class and regression cases across various domains.

Augmented LLMs. Since the groundbreaking success of LLMs [5], ongoing efforts have been made to augment LLMs with capabilities that are difficult to acquire through next-token predictions on pure text alone [23]. These efforts can generally be divided into two categories. The first category leverages external sources or tools to gather additional information, thereby endowing LLMs with unprecedented capabilities. Notable examples in this category include ReAct [43], which augments LLMs with a simple Wikipedia API, PAL [10], which combines LLMs with Python interpreters, and Toolformer [31], which teaches LLMs to use multiple tools. The second category introduces an additional learning procedure on new data, thereby inherently acquiring some new abilities. For instance, [27] further trained LLMs to align with human values, and [6] trained LLMs on code data, facilitating remarkable code understanding and generation. Viewed from the lens of augmented LLMs, this paper aligns with the second category. Our distinct contributions encompass the introduction of tabular data as a novel learning resource and testing platform for LLMs, along with the formulation of effective learning objectives for this data type.

3 GENERATIVE TABULAR LEARNING FOR LARGE LANGUAGE MODELS

Figure 1 provides a comprehensive overview of our generative paradigm for tabular deep learning with LLMs, including the pipeline for constructing tabular data in an instruction-oriented language format across various domains (Section 3.2), the detailed optimization process in GTL (Section 3.3), and an example of the prompt-based adaptation of the GTL-guided LLM to downstream tasks.

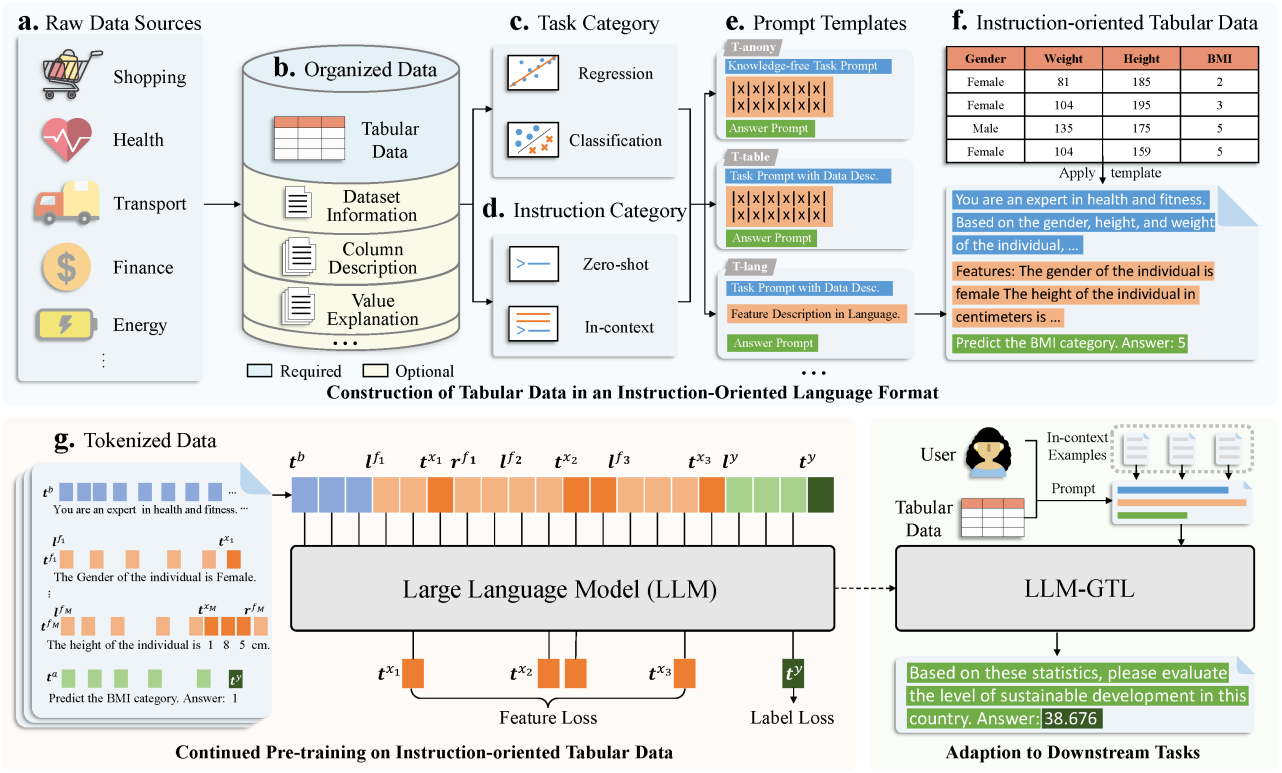

Figure 1: An overview of generative tabuar learning for LLMs.

3.1 Problem Formulation

Notations. In the field of tabular learning, we typically handle a tabular task $\mathcal{T} : \mathcal{X} \to \mathcal{Y}$, which associates a tabular instance $\boldsymbol{x} \in \mathcal{X}$, consisting of M features, $\{x_i\}_{i=1}^{M}$, with a prediction target, $y \in \mathcal{Y}$. For regression tasks, $\mathcal{Y} \subseteq \mathbb{R}^1$ and for C-class classification tasks, $\mathcal{Y} = \{0, 1, \cdots, C-1\}$. Note that tabular features are generally divided into numerical and categorical types, while we do not explicitly distinguish between them here. Besides, each tabular task may also be associated with various meta-information elements, such as the task background, prediction target interpretation, and feature descriptions. We denote the comprehensive meta-information of task $\mathcal{T}$ as $\mathcal{M}^{\mathcal{T}}$.

Traditional tabular data learning methods primarily focus on constructing a discriminative model to learn the dependency, $p(y|\boldsymbol{x})$, between the target and features using the training data. However, such a model strongly tied to the training data faces significant challenges when adapting to a new tabular task, $\tilde{\mathcal{T}}$, with a different data schema $\tilde{\mathcal{X}}$ and prediction targets $\tilde{\mathcal{Y}}$.

Zero-shot Learning. In the realm of tabular data, zero-shot learning [13, 42] is typically characterized by the ability to predict outcomes for data samples from a previously unseen task, $\tilde{\mathcal{T}}$. This new task possesses an unknown data schema, $\tilde{\mathcal{X}}$, but it has meta-information $\mathcal{M}^{\tilde{\mathcal{T}}}$, which encapsulates the meaning of features and prediction targets. Formally, we define the input of a zero-shot learning task as a tuple $(\mathcal{M}^{\tilde{\mathcal{T}}}, \tilde{\boldsymbol{x}}^{new})$, where $\mathcal{M}^{\tilde{\mathcal{T}}}$ denotes the meta-information for the new task and $\tilde{\boldsymbol{x}}^{new}$ represents a novel data sample for which we aim to make predictions.

In-context Learning (Few-shot Learning). In-context learning resembles zero-shot learning but with an additional dimension: it includes a set of exemplars from the new task $\tilde{\mathcal{T}}$ as demonstrations, represented as $D^{\tilde{\mathcal{T}}} = \{(\tilde{\boldsymbol{x}}^j, \tilde{y}^j)\}_{j=1}^{N}$, where N signifies the quantity of in-context examples. We formally characterize the input of an in-context learning task as a tuple $(\mathcal{M}^{\tilde{\mathcal{T}}}, D^{\tilde{\mathcal{T}}}, \tilde{\boldsymbol{x}}^{new})$, where the meta-information $\mathcal{M}^{\tilde{\mathcal{T}}}$ can be omitted if the model does not utilize it, as in the case of TabPFN [14]. In-context learning can seamlessly transition into few-shot learning by allowing the model to undergo further fine-tuning on $D^{\tilde{\mathcal{T}}}$ before making predictions on $\tilde{\boldsymbol{x}}^{new}$.

3.2 Construction of Tabular Data in an Instruction-Oriented Language Format

The upper part of Figure 1 depicts the pipeline for our data construction. Our preliminary step is to collect a significant amount of tabular datasets, which cover a broad spectrum of predictive tasks across multiple domains. This process, with more details in Section 4.1, requires the acquisition of both features and labels, better including meta-information (optional). Then, for each specified tabular task, $\mathcal{T}$, and any accompanying meta information $\mathcal{M}^{\mathcal{T}}$, it is essential to transform the data samples from their original tabular

format into an instruction-oriented language format to support the subsequent GTL procedure.

This instruction-oriented format primarily consists of three components. The first component pertains to task instructions, detailing the background and the objective of this prediction task. The second component offers detailed descriptions of the features, encompassing both their values and associated meanings (when available). It's important to highlight that we have the ability to incorporate in-context examples within this component, preceding the current data sample. Additionally, we can introduce a prompt indicating the presence of in-context demonstrations for reference. The third component encompasses answer descriptions, typically prefaced with an answer instruction, followed by the prediction targets.

Adhering to these guidelines, we devise various templates that cater to different ways of expressing feature descriptions and varying requirements.

Initially, we formulate the `T-lang` template, which delineates tabular features in a manner akin to natural language, similar to previous research in feeding tabular data to LLMs [8, 13]. The major advantage of this template is that it emulates human language style, facilitating smoother knowledge transfer for LLMs, given their extensive training on human language data. A language-style description of a tabular data instance may induce improved performance. Detailed composition of this template is provided in Appendix A.3.1. However, it is important to note that this template falls short in efficiently integrating in-context examples, as it necessitates the conversion of each example into a description, resulting in repetitive descriptions for feature meanings.

To address this limitation, we introduce the `T-table` template, which employs a Markdown table format to encapsulate tabular feature values. This approach ensures that irrespective of the number of in-context examples added, we only need to specify a single table header containing brief tags for different features. Feature meanings are positioned ahead of this Markdown table. The `T-table` template efficiently handles tabular data, particularly in the in-context learning setup. However, it also poses challenges for the long-term association capabilities of LLMs, as they inherently need to link a feature value with its corresponding table column and subsequently associate it with the correct feature meaning explanations. Detailed composition of this template is provided in Appendix A.3.2.

The `T-anony` template, a derivative of the `T-table` template, omits all meta-information. This is designed to simulate a practical scenario where we possess tabular data samples, but lack knowledge of their background or feature meanings. This setup is more akin to TabPFN [14], which did not utilize semantic information about feature columns and task background. Detailed composition of this template is provided in Appendix A.3.3.

By employing these templates, we can adapt to an array of tabular tasks and data schemas across diverse domains. Additionally, we have the capability to utilize meta information associated with these tabular datasets when available, and also have mechanisms in place for situations where such information is absent. Furthermore, we can regulate the number of in-context examples incorporated, thereby effortlessly incorporating both zero-shot and in-context learning scenarios.

3.3 Learning and Adaptation

The bottom part of Figure 1 intuitively depicts the training and adaptation of GTL. Next, we formally introduce the GTL paradigm.

Notations for Tokenized Instruction-oriented Language Data. Below we mainly leverage the `T-lang` template to derive the notations for a tokenized instruction-oriented language sample. Recalling in the last section, an instruction-oriented language format for tabular data primarily incorporates three components, encompassing a task instruction, a feature description, and an answer prompt. Formally, we represent the tokenization results of the first part as: $\boldsymbol{t}^b = [t^b_1, \cdots, t^b_{|\boldsymbol{t}^b|}]$, where $|\cdot|$ is the operation to denote the length of a token sequence. Besides, we denote the i-th feature as $\boldsymbol{t}^{f_i} = [\boldsymbol{l}^{f_i}, \boldsymbol{t}^{x_i}, \boldsymbol{r}^{f_i}]$, where $\boldsymbol{l}^{f_i} = [l^{f_i}_1, \cdots, l^{f_i}_{|\boldsymbol{l}^{f_i}|}]$ includes all tokens of feature descriptions on the left of this feature value, $\boldsymbol{t}^{x_i} = [t^{x_i}_1, \cdots, t^{x_i}_{|\boldsymbol{t}^{x_i}|}]$ denotes the token sequence for the feature value x_i, and $\boldsymbol{r}^{f_i} = [r^{f_i}_1, \cdots, r^{f_i}_{|\boldsymbol{r}^{f_i}|}]$ includes the tokens for remaining feature descriptions on the right of feature values. By concatenating token sequences for all features, we have the overall feature sequence as $\boldsymbol{t}^f = [\boldsymbol{t}^{f_1}, \cdots, \boldsymbol{t}^{f_M}]$. Last, we represent the tokenized answer prompt as $\boldsymbol{t}^a = [\boldsymbol{l}^a, \boldsymbol{t}^y]$, where $\boldsymbol{l}^a = [l^a_1, \cdots, l^a_{|\boldsymbol{l}^a|}]$ denotes the answer prompt before the answer tokens $\boldsymbol{t}^y = [t^y_1, \cdots, t^y_{|\boldsymbol{t}^y|}]$. In this way, we can express a tabular instance $\boldsymbol{x}$ and the associated target y as a sequence of tokens:

$$[\boldsymbol{t}^b, \boldsymbol{t}^f, \boldsymbol{t}^a] = [\boldsymbol{t}^b, \boldsymbol{l}^{f_1}, \boldsymbol{t}^{x_1}, \boldsymbol{r}^{f_1}, \cdots, \boldsymbol{l}^{f_M}, \boldsymbol{t}^{x_M}, \boldsymbol{r}^{f_M}, \boldsymbol{l}^a, \boldsymbol{t}^y], \quad (1)$$

which systematically unifies task background ($\boldsymbol{t}^b$), feature meanings ($\{\boldsymbol{l}^{f_i}, \boldsymbol{r}^{f_i}\}_{i=1}^M$), and feature values ($\{\boldsymbol{t}^{x_i}\}_{i=1}^M$) and supports various prediction targets via a variable-length sequence ($\boldsymbol{t}^y$). Please note that, the above tokenization notations also apply for other templates, such as `T-table`, merely with a different way of specifying feature meanings. Besides, we need to mark the positions of feature value tokens and prediction target tokens of the current tabular data sample to support the subsequent GTL procedure.

The Objective of GTL. Based on the notations for tokenized data, we devise GTL as characterizing the following joint distribution:

$$p(\boldsymbol{x}, y) = p(\boldsymbol{t}^{\boldsymbol{x}}, \boldsymbol{t}^y | \boldsymbol{t}^m) = p(\boldsymbol{t}^y | \boldsymbol{t}^{\boldsymbol{x}}, \boldsymbol{t}^m) \prod_{i=1}^{M} p(\boldsymbol{t}^{x_i} | \boldsymbol{t}^{<x_i}), \quad (2)$$

where we introduce additional notations to ensure the concise formulation, using $\boldsymbol{t}^m = [\boldsymbol{t}^b, \boldsymbol{l}^{f_1}, \boldsymbol{r}^{f_1}, \cdots, \boldsymbol{l}^{f_M}, \boldsymbol{r}^{f_M}, \boldsymbol{l}^a]$ to denote all meta information, $\boldsymbol{t}^{\boldsymbol{x}} = [\boldsymbol{t}^{x_1}, \cdots, \boldsymbol{t}^{x_M}]$ to represent all tokens related to feature values, and $\boldsymbol{t}^{<x_i}$ to include all tokens ahead of $\boldsymbol{t}^{x_i}$ in equation 1. Here $p(\boldsymbol{x}, y)$ denotes the joint distribution in the initial feature and label spaces, while $p(\boldsymbol{t}^{\boldsymbol{x}}, \boldsymbol{t}^y | \boldsymbol{t}^m)$ represents the same joint distribution conditioned on all meta information using the text representation, which can further be decoupled autoregressively into $p(\boldsymbol{t}^y | \boldsymbol{t}^{\boldsymbol{x}}, \boldsymbol{t}^m) \prod_{i=1}^M p(\boldsymbol{t}^{x_i} | \boldsymbol{t}^{<x_i})$. It is thus straightforward to leverage LLMs, especially those using auto-regressive architectures, to characterize this decoupled formulation. The only modification needed is to mask out the losses on meta-information tokens. While GTL employs the next-token prediction loss akin to LLMs, it distinguishes itself from auto-regressive pre-training on

language data. Specifically, GTL explicitly stimulates LLMs to discern complex dependencies between prediction target tokens and feature tokens. It encourages the capture of intricate relationships between current features and in-context examples, and fosters the establishment of effective connections between diverse language instructions and numerical data.

Adaptation to Downstream Tasks. Given an LLM, we refer to its variant after the GTL process as LLM-GTL. When adapting to a new task, irrespective of the data schema or task type, LLM-GTL facilitates direct inference by simply specifying a prompt. For instance, if the objective is to optimally utilize prior knowledge for zero-shot inference on semantically-rich tasks, the `T-lang` template can be used to transform data samples, which are then fed into the LLM-GTL for output generation. In situations that necessitate the inclusion of more in-context examples and also require leveraging meta information about the tabular task, the `T-table` template emerges as the best practice. Furthermore, even in the absence of meta information, LLM-GTL can still offer predictions via the `T-anony` template, relying on the inherent statistical learning acquired during the GTL process.

4 EXPERIMENTS

This section presents extensive empirical investigations to address the following key research questions: 1) How does the application of GTL to an LLM impact its convergence and generalization behaviors on tabular deep learning? 2) What are the crucial scaling laws associated with GTL? 3) How do different text templates influence the performance? 4) How do GTL-enhanced LLMs compare against both traditional tabular models and contemporary LLMs?

4.1 Experimental Setups

Dataset Collection. We have curated a collection of 384 public tabular datasets from Kaggle[1], which includes 176 classification and 208 regression tasks spanning a wide range of industrial domains. For the purpose of holdout evaluation, we have randomly selected 44 tasks, consisting of 20 classification and 24 regression tasks, and carefully examined to ensure no overlap with the remaining tasks utilized for the continued pre-training with GTL. Further details pertaining to the statistics of the datasets and the covered domains can be found in Appendix A.2.

Configurations for Pre-training. For the 340 tabular datasets allocated for continued pre-training, we examine six different scenarios of zero-shot or in-context learning, with the number of in-context examples in $\{0, 4, 8, 16, 32, 64\}$. To further enrich the diversity of learning experiences, we have created up to four unique tasks for each dataset by judiciously selecting various columns as labels and the remainder as features. Each scenario is tested with all three text templates, resulting in a total of 14k cases. Typically, we randomly select 64 data samples for each case, obtaining 880k tokenized sequences for GTL. We impose a maximum sequence length of $4,096$ and discard any data samples that exceed this limit, yielding a total of 640k data samples for pre-training. We also sample a small subset of test samples, distinct from these pre-training samples, to assess in-domain generalization on these pre-training datasets across different instruction categories. We sample 16 per classification case and 4 per regression case (the latter requiring multi-step decoding which is highly time-consuming during evaluation). We have utilized the 7B and 13B versions of LLaMA 2 [35] as our base LLMs. The corresponding GTL-enhanced variants are denoted as LLaMA-7B-GTL and LLaMA-13B-GTL, respectively. Implementation details can be found in Appendix B.2.

Configurations for Holdout Evaluation. For the 44 holdout datasets, we adopt a similar approach to that of pre-training data, considering different in-context examples and text templates. To achieve a robust evaluation that mitigates the effects of randomness, we use three distinct random seeds to sample evaluation examples for each case, where one case is a combination of a dataset, a in-context configuration, and a text template. We increase the sample size to 64 for classification cases and 16 for regression cases. By observing performance variations across different seeds within the same case, we ascertain that this configuration balances the computational cost of running diverse experiments with the statistical significance of the results obtained. In total, we have around 88k data samples for holdout evaluation. The primary metrics used for evaluation are the Area Under the Receiver Operating Characteristic (AUROC) for classification tasks and the Normalized Mean Absolute Error (NMAE) for regression tasks.

Baselines. Our baseline includes both competitive tabular models and contemporary large language models (LLMs). For the tabular models, we have included classic tree-based models such as XGBoost [7], CatBoost [28], and LightGBM [18]. We have also considered the state-of-the-art in-context model, TabPFN [14], a competitive neural model, FTT [11], and a logistic regression (LR) baseline, known for its robustness in extremely few-shot scenarios. As for the LLMs, we have included the initial versions of LLaMA 2, denoted as LLaMA-7B and LLaMA-13B, and proprietary LLMs such as GPT-3.5 [5] and GPT-4 [26]. We assessed the proprietary models by invoking their APIs on our holdout data. To calculate AUROC using prediction probabilities, we instructed these models to output probabilities. For LLaMA and GTL models, due to our access to their networks, we instructed them to predict class indices directly and collected logits over the class tokens to acquire output probabilities. For the NMAE metric, all LLMs require multi-step decoding, for which we allowed a maximum of ten steps for LLaMA models, sufficient for all cases under consideration in this study.

4.2 Understanding GTL

We investigate the generalization behaviors during the optimization of GTL, as depicted in Figure 2. Notably, a significant gap emerges between the improvements of in-domain and out-of-domain generalization in the zero-shot scenario. Initially, both generalization abilities enhance swiftly, but as training progresses, in-domain generalization continues to improve while out-of-domain generalization plateaus. This plateau occurs due to the presence of domain-specific knowledge in holdout data, which is not covered by either the LLM's prior understanding or pre-train data. Consequently, early training steps primarily improve tabular data understanding. As training deepens, the model continues to learn patterns within

[1] https://www.kaggle.com/datasets

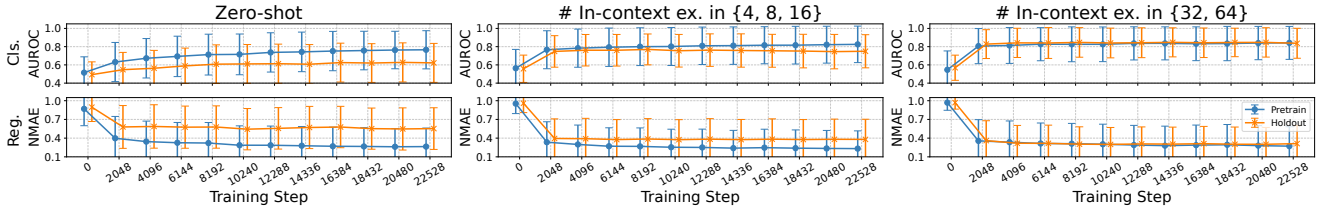

Figure 2: An examination of in-domain (on pre-train test sets) and out-of-domain (on holdout data) generalization as GTL optimization progresses, highlighting the zero-shot learning scenario (left), the in-context learning scenario with few in-context examples (middle), and with many examples (right).

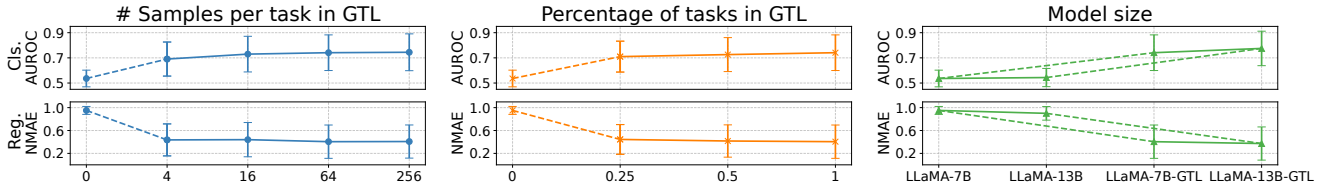

Figure 3: A study of scaling laws, focusing on the number of data samples per pre-training task (left), the percentage of pre-training tasks (middle), and the model size (right). Both AUROC and NMAE metrics, aggregated from all holdout datasets across different context examples, are represented. A dotted line connects a LLaMA model with its GTL variant for comparison.

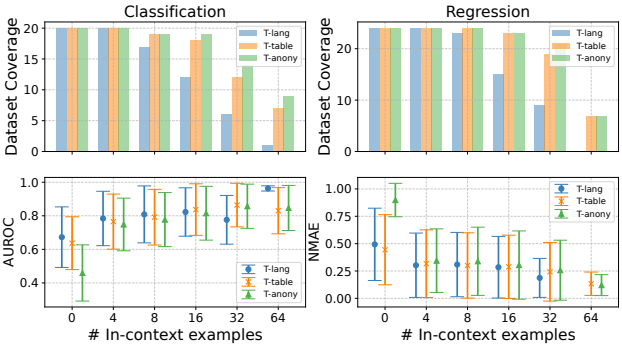

Figure 4: The influence of template variability on dataset coverage, given a maximum sequence length, and on generalization performance, considering different numbers of in-context examples on holdout datasets.

pre-train domains but acquires no further universal abilities for holdout datasets. However, the introduction of in-context examples narrows this gap. Specifically, when more than 32 in-context examples are included, the gap between in-domain and out-of-domain generalization nearly disappears, whereas they roughly converge after 8, 192 steps. This analysis suggests that early stopping of GTL optimization can be beneficial for generalization on unseen datasets and domains. Therefore, in subsequent experiments on scaling laws, we halt pre-training at step 8, 192.

The study of scaling laws seeks to identify key factors influencing generalization to holdout data, which could indicate potential avenues for future breakthroughs. We primarily focus on the impact of scaling the number of data samples per task, the number of tasks used for GTL, and the model size. As shown in Figure 3, scaling the number of data samples per tabular task improves generalization quickly when LLMs have limited exposure to tabular data. However, beyond a certain threshold, such as 64, further increases to 256 do not significantly enhance holdout data generalization. In contrast, scaling the number of tasks consistently improves generalization, suggesting that expanding task coverage beyond the 340 datasets used for pre-training here could further enhance GTL performance. Moreover, increasing the model size, for example from 7B to 13B, significantly boosts performance.

Additionally, we thoroughly compare the effects of different data templates on holdout generalization across various in-context scenarios. As Figure 4 depicts, the `T-lang` template, which best adheres to language conventions, excels in semantic-dominant scenarios, particularly zero-shot and few-shot classification tasks. However, due to its inefficient token usage, it becomes less effective as the number of in-context examples increase, with a significant drop-off observed at 16 examples. In contrast, regression tasks, which primarily involve fitting a numerical target based on different features, do not heavily rely on semantic information. Based on these findings, we propose a template usage strategy: `T-lang` for zero-shot classification cases, `T-table` for other scenarios when meta information is available, and `T-anony` when the origin of the tabular data is unknown.

4.3 Performance Comparisons with Competitive Tabular Models and LLMs

We further contrast our approach with competitive tabular models and contemporary LLMs, as illustrated in Table 1 and Table 2.

When compared to tabular models, our approach excels in numerous few-shot cases, including classification tasks with in-context examples ranging from 4 to 16, and regression tasks with examples from 4 to 32. These findings validate the effectiveness of the GTL

Table 1: Classification results. We display the mean, 25th percentile, and 75th percentile AUROC scores across all datasets for both zero-shot and in-context learning scenarios. Rank_z denotes the average rank in zero-shot settings, while Rank_i signifies the average rank in in-context settings.

	0 (Zero-shot)	4	8	16	32	64	Rank_z	Rank_i
LightGBM	–	$0.500_{[0.500,0.500]}$	$0.500_{[0.500,0.500]}$	$0.500_{[0.500,0.500]}$	$0.500_{[0.500,0.500]}$	$0.848_{[0.716,0.965]}$	–	9.754
XGBoost	–	$0.555_{[0.500,0.543]}$	$0.685_{[0.500,0.799]}$	$0.771_{[0.671,0.941]}$	$0.834_{[0.710,0.959]}$	$0.834_{[0.767,0.963]}$	–	6.174
CatBoost	–	$0.727_{[0.612,0.882]}$	$0.774_{[0.675,0.901]}$	$0.802_{[0.735,0.930]}$	$\mathbf{0.853_{[0.746,0.973]}}$	$0.844_{[0.778,0.956]}$	–	4.272
FTTransformer	–	$0.669_{[0.480,0.815]}$	$0.673_{[0.542,0.844]}$	$0.679_{[0.533,0.785]}$	$0.709_{[0.594,0.820]}$	$0.699_{[0.609,0.773]}$	–	7.564
XTab	–	$0.588_{[0.457,0.734]}$	$0.597_{[0.505,0.678]}$	$0.671_{[0.563,0.769]}$	$0.759_{[0.649,0.884]}$	$0.767_{[0.666,0.941]}$	–	7.554
LR	–	$0.715_{[0.542,0.886]}$	$0.743_{[0.630,0.889]}$	$0.800_{[0.725,0.925]}$	$0.850_{[0.759,0.965]}$	$0.813_{[0.700,0.954]}$	–	4.308
TabPFN	–	$0.722_{[0.604,0.884]}$	$0.758_{[0.634,0.872]}$	$0.803_{[0.733,0.933]}$	$0.852_{[0.747,0.970]}$	$\mathbf{0.850_{[0.787,0.946]}}$	–	4.236
LLaMA-7B	$0.504_{[0.457,0.585]}$	$0.545_{[0.451,0.622]}$	$0.525_{[0.457,0.585]}$	$0.494_{[0.429,0.556]}$	$0.493_{[0.430,0.545]}$	$0.500_{[0.461,0.557]}$	4.000	10.251
GPT-3.5	$0.539_{[0.456,0.632]}$	$0.525_{[0.490,0.576]}$	$0.484_{[0.466,0.541]}$	$0.536_{[0.485,0.571]}$	$0.509_{[0.455,0.552]}$	$0.520_{[0.483,0.549]}$	3.900	10.097
LLaMA-13B	$0.495_{[0.450,0.566]}$	$0.507_{[0.436,0.573]}$	$0.555_{[0.461,0.582]}$	$0.520_{[0.461,0.590]}$	$0.487_{[0.446,0.555]}$	$0.521_{[0.472,0.562]}$	4.550	9.990
GPT-4	$\mathbf{0.654_{[0.537,0.824]}}$	$0.642_{[0.563,0.785]}$	$0.721_{[0.596,0.887]}$	$0.713_{[0.573,0.832]}$	$0.711_{[0.570,0.866]}$	$0.658_{[0.531,0.753]}$	**2.450**	6.805
LLaMA-7B-GTL	$0.579_{[0.473,0.679]}$	$0.714_{[0.581,0.847]}$	$0.777_{[0.649,0.927]}$	$0.793_{[0.656,0.940]}$	$0.821_{[0.728,0.936]}$	$0.813_{[0.720,0.933]}$	3.550	5.164
LLaMA-13B-GTL	$0.641_{[0.565,0.752]}$	$\mathbf{0.767_{[0.663,0.919]}}$	$\mathbf{0.811_{[0.681,0.960]}}$	$\mathbf{0.823_{[0.752,0.955]}}$	$0.850_{[0.777,0.955]}$	$0.816_{[0.733,0.918]}$	2.500	**3.903**

Table 2: Regression results. We display the mean, 25th percentile, and 75th percentile NMAE (Normalized Mean Absolute Error) scores across all datasets for both zero-shot and in-context learning scenarios. NMAE values are truncated at a maximum of 1.0 to account for certain baselines' diminished performance due to inadequate training data. Rank_z denotes the average rank in zero-shot settings, while Rank_i signifies the average rank in in-context settings.

	0 (Zero-shot)	4	8	16	32	64	Rank_z	Rank_i
LightGBM	–	$0.422_{[0.220,0.534]}$	$0.414_{[0.199,0.540]}$	$0.436_{[0.226,0.554]}$	$0.439_{[0.191,0.550]}$	$0.281_{[0.091,0.252]}$	–	7.304
XGBoost	–	$0.376_{[0.154,0.475]}$	$0.350_{[0.134,0.512]}$	$0.348_{[0.116,0.507]}$	$0.284_{[0.054,0.400]}$	$\mathbf{0.141_{[0.029,0.161]}}$	–	5.507
CatBoost	–	$0.372_{[0.165,0.520]}$	$0.341_{[0.142,0.459]}$	$0.337_{[0.115,0.434]}$	$0.313_{[0.069,0.405]}$	$0.224_{[0.040,0.184]}$	–	4.996
XTab	–	$0.504_{[0.251,0.779]}$	$0.419_{[0.166,0.634]}$	$0.398_{[0.157,0.567]}$	$0.400_{[0.132,0.732]}$	$0.275_{[0.097,0.263]}$	–	7.386
FTTransformer	–	$0.414_{[0.206,0.514]}$	$0.407_{[0.201,0.509]}$	$0.433_{[0.221,0.572]}$	$0.424_{[0.189,0.610]}$	$0.355_{[0.180,0.362]}$	–	7.346
LR	–	$0.384_{[0.151,0.540]}$	$0.373_{[0.104,0.595]}$	$0.358_{[0.058,0.553]}$	$0.382_{[0.041,0.927]}$	$0.211_{[0.018,0.164]}$	–	5.621
LLaMA-7B	$0.938_{[0.977,1.000]}$	$0.902_{[0.945,1.000]}$	$0.954_{[1.000,1.000]}$	$0.995_{[1.000,1.000]}$	$0.987_{[1.000,1.000]}$	$0.956_{[0.988,1.000]}$	5.250	10.604
LLaMA-13B	$0.770_{[0.603,1.000]}$	$0.792_{[0.535,1.000]}$	$0.844_{[0.826,1.000]}$	$0.873_{[0.999,1.000]}$	$0.938_{[0.999,1.000]}$	$0.774_{[0.429,1.000]}$	4.167	9.932
GPT-3.5	$0.492_{[0.199,0.842]}$	$0.393_{[0.183,0.470]}$	$0.406_{[0.217,0.503]}$	$0.426_{[0.211,0.530]}$	$0.408_{[0.179,0.506]}$	$0.325_{[0.157,0.410]}$	3.083	7.114
GPT-4	$0.583_{[0.241,0.958]}$	$\mathbf{0.229_{[0.083,0.304]}}$	$\mathbf{0.233_{[0.072,0.297]}}$	$\mathbf{0.229_{[0.064,0.286]}}$	$\mathbf{0.214_{[0.054,0.268]}}$	$0.146_{[0.030,0.240]}$	3.250	**2.918**
LLaMA-7B-GTL	$0.523_{[0.223,0.957]}$	$0.316_{[0.102,0.417]}$	$0.299_{[0.082,0.387]}$	$0.311_{[0.081,0.513]}$	$0.291_{[0.068,0.511]}$	$0.229_{[0.031,0.315]}$	2.792	4.589
LLaMA-13B-GTL	$\mathbf{0.430_{[0.172,0.671]}}$	$0.288_{[0.079,0.403]}$	$0.271_{[0.059,0.398]}$	$0.295_{[0.063,0.415]}$	$0.260_{[0.050,0.355]}$	$0.165_{[0.035,0.263]}$	**2.083**	3.946

paradigm in integrating prior knowledge and tabular data understanding, a fusion that could be invaluable in addressing data-scarce challenges in many practical applications, such as rare disease classification or new material property prediction. A closer look at the classification results reveals TabPFN to be a robust baseline, outperforming us in average rank due to its superior handling of larger in-context examples. Regarding regression tasks, our untuned LLaMA-13B-GTL model frequently surpasses other models that have been tuned on in-context examples, also demonstrating the effectiveness of GTL. Besides, we note a rise in error with our model when 64 contexts are used, potentially due to the limited inclusion of such examples in pre-training regression tasks. We expect this limitation can be mitigated by scaling up the maximum sequence length, model size, and task quantity.

Our GTL-enhanced LLMs, potentially utilizing smaller model sizes, surpass GPT-4 in numerous classification scenarios and significantly narrow the performance gap in regression tasks. GPT-4, being proprietary with undisclosed pre-training data, raises potential concerns of data contamination since all tabular datasets used in our study are publicly available on Kaggle. Despite this potential issue, the exceptional performance exhibited by our LLaMA-GTL series, derived from open-source LLMs, affirms the effectiveness of the GTL paradigm in bolstering tabular deep learning capabilities for LLMs. Beyond the realm of tabular data, our findings suggest that open-source LLMs should integrate instruction-oriented tabular data into their pre-training corpus. This could amplify their data comprehension capabilities, especially considering the ubiquity of tabular data across various fields, and possibly spark new discoveries through the amalgamation of diverse data types. Finally, we conjecture that the superior performance of our GTL methodology over GPT-4 in classification tasks might be ascribed to different approaches in prediction probability computation, as we are limited to accessing GPT-4's prediction probabilities via instruction-based API calls.

5 CONCLUSION

This study introduces GTL, an generative paradigm that fuses LLM's instruction-following capabilities with tabular deep learning. The preliminary results demonstrate its effectiveness, particularly in tackling data scarcity issues, and our scaling analyses suggest avenues for further performance enhancements.

Limitations. Despite these encouraging outcomes, our study does have limitations. The context length constraint of LLMs restricts the number of in-context examples we can use, which can hinder learning over large-scale data. Besides, using a text format to represent tabular data is not token-usage efficient. Furthermore, the computational cost of LLMs is considerably higher than traditional techniques, raising financial and environmental concerns. Lastly, despite using 384 datasets in this study, the diversity and scale may not fully represent the potential and applicability of GTL across all tabular data fields.

Future Directions. In the meanwhile, we are optimistic about this research direction's future prospects. For example, the GTL paradigm opens doors for conversational tabular deep learning, where models could refine predictions through dialogue. GTL could also promote interpretable learning over tabular data, as LLMs can generate explanations with predictions, albeit with challenges in ensuring explanation faithfulness. There is potential for better integration between LLMs and traditional models, merging their strengths. Furthermore, our work provides a valuable asset for the LLM community, serving both as a benchmark for evaluating data comprehension capabilities and as a source to enrich existing pre-training corpora.

REFERENCES

[1] Anonymous. 2024. Making Pre-trained Language Models Great on Tabular Prediction. In *ICLR*.

[2] Sercan Ö Arik and Tomas Pfister. 2021. TabNet: Attentive interpretable tabular learning. In *AAAI*.

[3] Dara Bahri, Heinrich Jiang, Yi Tay, and Donald Metzler. 2022. SCARF: Self-Supervised Contrastive Learning using Random Feature Corruption. In *ICLR*.

[4] Sebastian Bordt, Harsha Nori, and Rich Caruana. 2023. Elephants Never Forget: Testing Language Models for Memorization of Tabular Data. In *NeurIPS 2023 Second Table Representation Learning Workshop*.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.

[6] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. 2021. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374* (2021).

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *KDD*.

[8] 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. In *NeurIPS*.

[9] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001).

[10] Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. PAL: Program-aided language models. In *ICML*.

[11] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting deep learning models for tabular data. In *NeurIPS*.

[12] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. 2022. Why do tree-based models still outperform deep learning on typical tabular data?. In *NeurIPS*.

[13] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. TabLLM: Few-shot classification of tabular data with large language models. In *AISTATS*.

[14] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2023. TabPFN: A Transformer That Solves Small Tabular Classification Problems in a Second. In *ICLR*.

[15] Shaohan Huang, Li Dong, Wenhui Wang, Yaru Hao, Saksham Singhal, Shuming Ma, Tengchao Lv, Lei Cui, Owais Khan Mohammed, Qiang Liu, et al. 2023. Language is not all you need: Aligning perception with language models. *arXiv preprint arXiv:2302.14045* (2023).

[16] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin. 2020. TabTransformer: Tabular data modeling using contextual embeddings. *arXiv preprint arXiv:2012.06678* (2020).

[17] Liran Katzir, Gal Elidan, and Ran El-Yaniv. 2021. Net-DNF: Effective deep modeling of tabular data. In *ICLR*.

[18] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems*.

[19] John D Kelleher, Brian Mac Namee, and Aoife D'arcy. 2020. *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. MIT press.

[20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* (2015).

[21] Roman Levin, Valeriia Cherepanova, Avi Schwarzschild, Arpit Bansal, C. Bayan Bruss, Tom Goldstein, Andrew Gordon Wilson, and Micah Goldblum. 2023. Transfer Learning with Deep Tabular Models. In *ICLR*.

[22] Ilya Loshchilov and Frank Hutter. 2018. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[23] Grégoire Mialon, Roberto Dessi, Maria Lomeli, Christoforos Nalmpantis, Ramakanth Pasunuru, Roberta Raileanu, Baptiste Roziere, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, Edouard Grave, Yann LeCun, and Thomas Scialom. 2023. Augmented Language Models: a Survey. *TMLR* (2023).

[24] Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. 2021. Transformers can do bayesian inference. *arXiv preprint arXiv:2112.10510* (2021).

[25] Jaehyun Nam, Jihoon Tack, Kyungmin Lee, Hankook Lee, and Jinwoo Shin. 2023. STUNT: Few-shot Tabular Learning with Self-generated Tasks from Unlabeled Tables. In *ICLR*.

[26] OpenAI. 2023. GPT-4 Technical Report. *arXiv preprint arXiv:2303.08774* (2023).

[27] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *NeurIPS*.

[28] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. In *NeurIPS*.

[29] Foster Provost and Tom Fawcett. 2013. *Data Science for Business: What you need to know about data mining and data-analytic thinking*. " O'Reilly Media, Inc.".

[30] Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Teven Le Scao, Stella Biderman, Leo Gao, Thomas Wolf, and Alexander M Rush. 2022. Multitask Prompted Training Enables Zero-Shot Task Generalization. In *ICLR*.

[31] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. *arXiv preprint arXiv:2302.04761* (2023).

[32] Ravid Shwartz-Ziv and Amitai Armon. 2022. Tabular data: Deep learning is not all you need. *Information Fusion* (2022).

[33] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C Bayan Bruss, and Tom Goldstein. 2021. SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training. *arXiv preprint arXiv:2106.01342* (2021).

[34] Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[35] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* (2023).

[36] Talip Ucar, Ehsan Hajiramezanali, and Lindsay Edwards. 2021. SubTab: Subsetting features of tabular data for self-supervised representation learning. In *NeurIPS*.

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

[38] Ruiyu Wang, Zifeng Wang, and Jimeng Sun. 2024. UniPredict: Large Language Models are Universal Tabular Classifiers. arXiv:2310.03266 [cs.LG]
[39] Zifeng Wang, Chufan Gao, Cao Xiao, and Jimeng Sun. 2023. MediTab: Scaling Medical Tabular Data Predictors via Data Consolidation, Enrichment, and Refinement. *arXiv preprint arXiv:2305.12081* (2023).
[40] Zifeng Wang and Jimeng Sun. 2022. TransTab: Learning transferable tabular transformers across tables. In *NeurIPS*.
[41] Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. 2022. An Explanation of In-context Learning as Implicit Bayesian Inference. In *ICLR*.
[42] Yazheng Yang, Yuqi Wang, Guang Liu, Ledell Wu, and Qi Liu. 2024. UniTabE: A Universal Pretraining Protocol for Tabular Foundation Model in Data Science. In *ICLR*.
[43] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *ICLR*.
[44] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar. 2020. VIME: Extending the success of self- and semi-supervised learning to tabular domain. In *NeurIPS*.
[45] Tianping Zhang, Shaowen Wang, Shuicheng Yan, Li Jian, and Qian Liu. 2023. Generative Table Pre-training Empowers Models for Tabular Prediction. In *EMNLP*.
[46] Bingzhao Zhu, Xingjian Shi, Nick Erickson, Mu Li, George Karypis, and Mahsa Shoaran. 2023. XTab: Cross-table Pretraining for Tabular Transformers. In *ICML*.

A DATA

A.1 Detailed Statistics of Collected Datasets

To empower Large Language Models (LLMs) to build foundational knowledge and abilities across a variety of domains through generative tabular learning, the compilation of a diverse and substantial collection of tabular datasets is pivotal. We have carefully curated a collection of 384 public datasets, comprised of high-quality, functional tabular classification datasets procured from Kaggle.

A.2 Domain Distribution

We have randomly divided all the acquired Kaggle datasets into pre-training and holdout sets, ensuring the preservation of domain distribution. The top 15 domain tags, along with the associated number of datasets for each domain, are displayed in Figure 5.

A.3 Template

A.3.1 The `T-lang` *Template.* The T-lang template preserves the semantic information of data samples by converting each feature into a sentence.

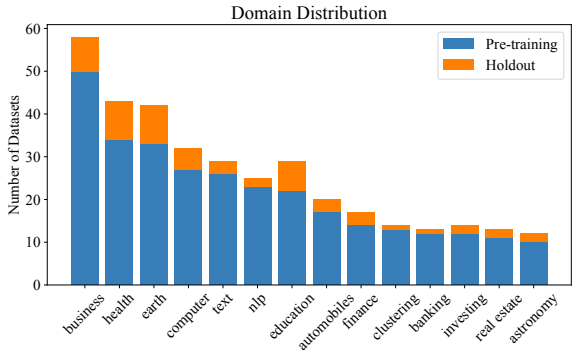

Figure 5: Domain distribution of pre-training and holdout datasets.

A.3.2 The `T-table` *Template.* The T-table template maintains the tabular format of the data samples. Meta information, such as feature description and label description, is added prior to the data and is not repeated with each example when in-context examples are introduced. An instance of the T-table template with in-context examples is showcased in List 1.

A.3.3 The `T-anony` *Template.* The T-anony template eliminates all meta information pertaining to the task, feature, and label. Similar to the T-table template, it organizes the tabular data in a markdown format.

B IMPLEMENTATION DETAILS

B.1 Baseline Models

Our experiments encompass two categories of baselines: large language models (LLMs) and traditional tabular models. For LLMs, we employ instruction following to predict the answer in both zero-shot and in-context scenarios. In the case of tabular models, we only compare results in few-shot settings. Specifically, for learning-based models such as XGBoost [7], CatBoost [28], LightGBM [18], FTT [11], and logistic regression (LR), we utilize in-context examples for training. Conversely, for TabPFN [14] and our GTL model, no parameter tuning is required; direct inference is achieved using the same examples through in-context learning. In order to conduct a fair comparison for all these baselines, we would sample category-balanced in-context samples for classification tasks.

B.1.1 LLMs Baseline. We assess the instruction following capabilities of several large language models, dividing them into two categories: black box models and white box models. Black box models include the GPT series, where we can only access the response rather than model parameters and prediction logits. In contrast, white box models like LLaMA, allow us to access the entire model and obtain logits for each class. The descriptions of each baseline model are as follows:

- **GPT-3.5** [5]: We employ the GPT-3.5-turbo version, which is the most capable GPT-3.5 model optimized for chat and designed for diverse tasks such as natural language understanding, translation, and summarization. To encourage the model to provide predictions and minimize instances where it refuses to make predictions, we incorporate an additional answer instruction prompt when querying GPT-3.5. This approach allows GPT-3.5 to predict the probabilities for each category, which can then be used to calculate AUROC.
- **GPT-4** [26]: A powerful baseline renowned for its top performance in numerous language tasks due to its larger model size and architectural refinements. We also fine-tune the answer instruction prompt for GPT-4. Compared to GPT-3.5, GPT-4 demonstrates superior performance in both zero-shot and in-context learning. The instruction prompt and the response format is the same with GPT-3.5.
- **LLaMA** [35]: We employ the LLaMA-2 7B and 13B version as our baseline and backbone for generative tabular learning.

B.1.2 Tradition Tabular Models Baseline. For traditional tabular models, to optimize the results for each model, we have utilized and finely tuned common pre-processing methods. These methods

Listing 1: Example of the T-table template with in-context examples

```
You are an expert in medical diagnosis.
Based on the features of the patient, please predict the diabetes diagnosis.
I will supply multiple instances with features and the corresponding label for reference.
Please refer to the table below for detailed descriptions of the features and label:
--- feature description ---
Age: The age of the individual in years
BMI: Body Mass Index of the individual
FBS: Fasting blood sugar level of the individual
HbA1c: Hemoglobin A1c level, a test to measure blood sugar level over the past 2 to 3 months
Gender: The gender of the individual, Male or Female
Blood Pressure: Blood pressure level of the individual, can be Normal or High
Family History: Whether the individual has a family history of diabetes or not
Smoking: Whether the individual smokes or not
Diet: The quality of the individual diet, can be Healthy or Poor
Exercise: Whether the individual exercises regularly or not
--- label description ---
Diagnosis: Whether the individual is diagnosed with diabetes or not
--- data ---
|Age|BMI|FBS|HbA1c|Gender|Blood Pressure|Family History|Smoking|Diet|Exercise|Diagnosis|
|48|47.0|200.0|9.2|Male|High|Yes|Yes|Poor|No|0|
|70|35.0|140.0|7.1|Female|Normal|No|No|Healthy|Regular|0|
|12|10.0|80.0|5.0|Male|Low|No|Yes|Poor|No|1|
|75|40.0|160.0|7.8|Female|High|Yes|Yes|Poor|No|1|
|30|20.0|100.0|5.7|Female|Normal|No|No|Healthy|Regular|<MASK>|
Please use the supplied data to predict the <MASK> Diagnosis. Diagnosed with diabetes[1] or not[0]?
Answer: 0
```

include z-score normalization for numerical features and labels, and one-hot encoding for categorical features. The performance of these baseline models, following the application of these pre-processing strategies, are included in our results.

- **Logistic Regression**: A linear model used to predict binary response probabilities based on predictor variables. Both Logistic Regression and its regression counterpart, Linear Regression, are straightforward and efficient, often serving as baseline models in classification and regression tasks respectively. To enhance model performance, we employ z-score normalization on numerical features and one-hot encoding on categorical features.
- **XGBoost** [7]: An advanced gradient-boosted decision tree algorithm renowned for its efficiency and high performance. Popular for handling diverse datasets and commonly used in machine learning competitions.
- **LightGBM** [18]: A gradient boosting framework employing tree-based learning algorithms, designed for efficiency and scalability, offering faster training speeds and lower memory usage than other techniques.
- **CatBoost**[28]: A high-performance gradient boosting library that handles categorical features directly, providing an efficient and accurate model. We implemented z-score normalization on numerical features and regression task labels.
- **FTTransformers**[11]: This model employs transformer-based architectures for tabular data, offering a dynamic and adaptable framework that can be fine-tuned for various tasks. In these deep learning-based models, we've found that the results largely depend on pre-processing strategies and the number of training epochs. Therefore, we selected the optimal strategy based on results from the holdout datasets. We implemented z-score normalization on numerical features and regression task labels.
- **TabPFN** [14] A deep learning model that combines Positional Feature-wise Networks with transformer-based architectures for tabular data, capturing both local and global feature interactions to enhance performance in various tasks.

B.2 Generative Tabular Learning (GTL)

Hyper-parameters. We employ the LLaMA-2 [35] 7B and 13B version models as the backbone for our experiments. For generative tabular learning, we utilize a fixed learning rate of 1e-5 and a batch size of 512, without incorporating any scheduler or warmup. The training procedure involves gradient updates with the optimizer AdamW [22]. We set the limitation of maximum token numbers to 4096, which ensures that all samples are within the acceptable range and prevents truncation.

Experimental Environments. The GTL-enhanced LLaMA model is implemented using PyTorch version 2.1.0 and executed on CUDA 12.1, running on NVIDIA Tesla A100 GPUs. As for GTL, pre-training is performed on a single node equipped with 8 A100 GPUs and the micro batch size is 4 for LLaMA-7B and 2 for LLaMA-13B. With an overall batch size of 512, the gradient accumulation steps amount to 16 for LLaMA-7B and 32 for LLaMA-13B. In LLaMA-13B-GTL, updating the gradient for 256 batches, which comprises 131,072 samples, takes approximately 26 hours.