Communication Papers of the 19th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 53–59 DOI: 10.15439/2024F1407 ISSN 2300-5963 ACSIS, Vol. 41

Gradient Boosting Trees and Large Language Models for Tabular Data Few-Shot Learning

Carlos Huertas
Amazon Research
Email: carlohue@amazon.com

***Abstract*—Large Language Models (LLM) have brought numerous of new applications to Machine Learning (ML). In the context of tabular data (TD), recent studies show that TabLLM is a very powerful mechanism for few-shot-learning (FSL) applications, even if gradient boosting decisions trees (GBDT) have historically dominated the TD field. In this work we demonstrate that although LLMs are a viable alternative, the evidence suggests that baselines used to gauge performance can be improved. We replicated public benchmarks and our methodology improves LightGBM by 290%, this is mainly driven by forcing node splitting with few samples, a critical step in FSL with GBDT. Our results show an advantage to TabLLM for 8 or fewer shots, but as the number of samples increases GBDT provides competitive performance at a fraction of runtime. For other real-life applications with vast number of samples, we found FSL still useful to improve model diversity, and when combined with ExtraTrees it provides strong resilience to overfitting, our proposal was validated in a ML competition setting ranking first place.**

I. Introduction

TABULAR data in real-world applications is the most common type of data [1], this continues to be true since relational databases are still pretty common in all sort of domains from social to natural sciences [2]–[6]. Deep Learning (DL), or in general, Neural Network based architectures have shown tremendous potential in tasks like Natural Language Processing (NLP) with developments like transformers [7] and large-scale pre-trained models like DeBERTa [8] have pushed the state-of-the-art (SOTA) and gave DL a top spot in performance. The same can be observed for Computer Vision (CV) with developments like convolutional neural networks (CNN) opening the door for more advanced designs like EfficientNets [9] and more recently Vision Transformers (ViT) have found their way into CV as well [10] with Next-ViT [11] aiming to bridge the gap that still separates ViT from CNN in terms of efficiency in the latency/accuracy trade-off.

Despite all the success from DL, tabular data continues to be omnipresent [12], [13], and to the best of our knowledge, we have not found a consistent DL-based approach that can outperform Gradient Boosted Decision Trees (GBDT) [14]–[16] over a *wide variety* of tasks and conditions, even though it is possible to find specific niche setups where this happens [17]–[19].

Recently, the introduction of Large Language Models (LLM) [20] demonstrated a whole new level of performance for several tasks [21], [22], from traditional NLP to even code generation [23]. The concept of revisiting the qualities of DL-based techniques, in particular LLM for tabular data surged again [6], due to some of the key properties over GBDT [24], such as: representation learning, sequential processing and generalization. Even though DL provides some advantages, if maximum performance is desired, GBDT continues to be the SOTA [25] even with amazing advances in DL, some of the most notable attempts to outperform GBDT with DL methods include: Wide&Deep [26], DeepFM [27], SDTR [28], DeepGBM [29], TabNN [30], BGNN [31], TabNet [32], TransTab [33], TabTransformer [34], SAINT [35] and NPT [36], none of them providing enough evidence to actually be able to beat GBDT over a wide variety of tasks, most of the time, it has been demonstrated the claimed improvements are only present in very specific cases or datasets [17].

There are however, some situations where LLM based solutions seem to have an edge [6], this is when data is limited, and LLM have the capacity to perform both zero-shot (ZSL) and few-shot learning (FSL) [37]. While there is no doubt current SOTA in GBDT will show random-performance for zero-shot learning, recent studies [38] show that even under a few-shot schema, LLM can outperform Xgboost [14], one of the most popular GBDT algorithms.

In this work, we will further explore the performance of GBDT under a FSL schema in order to provide strong baselines. Since previous studies [17] have demonstrated bias in claims of DL outperforming GBDT in other tasks, we look to enhance experiments to confirm SOTA results in the new trend of results regarding FSL and the superiority of LLM over GBDT.

II. Related Work

The main concept behind ZSL or FSL by definition implies the evaluated classifier has either (a) never seen the data samples before (ZSL), or only a few samples (FSL), however, this can only be proven true if we were to train a model (LLM for the purpose of this research) from scratch. Any sort of pre-trained architecture could, in theory, already seen the dataset, hence showing incredible performance. This particular problem has been studied before [39], where both GPT-3.5 and GPT-4 are proven to have seen common datasets in the past, like *Adult Income* and *FICO* [40], in some cases, even proven LLM have literally memorized the datasets verbatim [41] as samples can be extracted out. With this in mind, the fair

TABLE I
GPT-3/4 VS TRADITIONAL ALGORITHMS FOR FEW-SHOT-LEARNING PERFORMANCE (AUC)

Algorithm	Kaggle Titanic	OpenML Diabetes	Adult Income	FICO	Spaceship Titanic	Pneumonia
GPT-4	**0.98**	0.75	0.82	0.68	0.69	0.81
GPT-3.5	0.82	0.74	0.79	0.65	0.63	0.54
GBDT (Xgboost)	0.84	0.75	**0.87**	**0.72**	**0.80**	**0.90**
Logistic Regression	0.79	**0.78**	0.85	**0.72**	0.77	**0.90**

evaluation of LLM vs GBDT under a truly FSL schema is very challenging, while we can guarantee GBDT has never seen the data, the same cannot be said for many LLM applications.

The results from Bordt et al. [39], using a 20-shot-learning are shown in Table I, in this work authors study LLM memorization.

Although LLM results are far from bad, the performance still shows gaps to match GBDT. On top of this, GBDT is a much simpler and faster model, essentially being a more efficient and more powerful option. For the Kaggle Titanic dataset, the power of GPT-4 might look impressive, until authors have proven this is due to memorization and not any particular useful learning. This problem is not particular to tabular data, as LLM have been proved to do so as well for other domains [42]. Nonetheless, authors have found that there is some learning happening, for datasets with no memorization LLM can still provide some performance, especially in very few shot-learning, which leads to the work of Hegselmann et al. [43], where LLM are shown to actually outperform GBDT.

In such work, authors present TabLLM, a very innovative solution to use LLMs for few-shot classification on tabular data, in principle, first running a serialization-step, to turn tabular into a natural language representation. An extensive analysis is done to benchmark multiple serialization techniques. Surprisingly, one of the simplest approaches resulted to be very effective, *“Text Template”* is a compact representation of the form: *“The <column name> is <value>”*. This followed by a task-specific prompt, that can later be fined-tuned for FSL.

TabLLM has been benchmarked for both binary and multiclass problems, from datasets identified in key literature for this task [19], [25], [44]. For simplicity, we will focus on the binary tasks, as to ensure all tasks are of the same objective, and metrics are comparable, e.g. AUC. A summary of their benchmarking results is presented in Table II. For full details refer to Table 12, 13 and 14 in [43].

The results show NN-based solutions, both TabPFN [47] and TabLLM [43], substantially outperform LightGBM for FSL, the improved performance by these techniques is such that the minimal delta observed comes in the *Bank* dataset where TabLLM shows an average (over 4 to 64 FSL) advantage of 163% relative improvement $[(0.642 - 0.5)/(0.554 - 0.5)]$ vs the GBDT solution. On the other extreme, the superiority of TabLLM goes further to outperform LightGBM for as much as 745% $[(0.686 - 0.5)/(0.522 - 0.5)]$ for the *Credit-g* experiment.

In the next section, we present our analysis regarding the extreme underperformance from LightGBM, and our recommendations to establish a fair baseline for a FSL application. Increasing its performance to a more competitive level, and hoping this serves as reference for future benchmarks in the field.

III. PROPOSED SOLUTION

The process of FSL might have slightly different interpretations depending on the field, but the core concept remains, the usage of only a few samples to train a model. This concept holds for the tabular data use-case. Knowing this, is imperative to understand how algorithms like LightGBM work in order to build an effective FSL solution. The LightGBM algorithm is a boosting approach using decisions trees (DT) to learn a function from the input space X^s to the gradient space G [15], the splitting criteria is reviewed below.

Given a training set with n i.i.d. instances $\{x_1, ..., x_n\}$, where each x_i is a vector with dimension s in space X^s. For each boosting iteration, the negative gradients of the loss function with respect to the output of the model are denoted as $\{g_1, ..., g_n\}$. The DT model splits each node to maximize information gain, which is measured by the variance *after* splitting. For a training set O on a fixed node, the gain of splitting (V) feature j at point d is defined as:

$$V_{j|O}(d) = \frac{1}{n_O}\left(\frac{\left(\Sigma_{\{x_i \in : x_{ij} \leq d\}} g_i\right)^2}{n^j_{l|O}(d)} + \frac{\left(\Sigma_{\{x_i \in : x_{ij} > d\}} g_i\right)^2}{n^j_{r|O}(d)}\right) \quad (1)$$

The problem however arises in practice since the optimization is constrained so that the left $n^j_{l|O}(d)$ and right $n^j_{r|O}(d)$ nodes have a minimum sample size. A segment of LightGBM implementation is shown in Algorithm 1.

The *minimum samples per leaf* then becomes a blocker for FSL, causing the algorithm to stall. Unable to perform any split until training samples exceeds the *min_samples_leaf* parameter. Although previous works [43] have explored parameter tuning based on literature recommendations [12], [19], this is not being addressed, and as a result LightGBM shows *random-guess* performance (e.g. 0.5 AUC) in most experiments, since the default value for *min_samples_leaf* is set to 20.

In this work we propose a LightGBM configuration specifically for FSL applications. We identified key parameters needed as shown in Table III.

The most important parameter for FSL is, without a doubt, *min_data_in_leaf*, as otherwise optimization cannot happen.

TABLE II
TabLLM Experiments Results: LightGBM (GBDT) vs NN-Based (AUC)

Dataset	Method	4-shot	8-shot	16-shot	32-shot	64-shot	Average
Bank [44]	LightGBM	0.50	0.50	0.50	0.50	0.77	0.554
	TabPFN	0.59	0.66	0.69	0.76	0.82	**0.704**
	TabLLM	0.59	0.64	0.65	0.64	0.69	0.642
Blood [44]	LightGBM	0.50	0.50	0.50	0.50	0.69	0.538
	TabPFN	0.52	0.64	0.67	0.70	0.73	**0.652**
	TabLLM	0.58	0.66	0.66	0.68	0.68	0.652
Credit-g [44]	LightGBM	0.50	0.50	0.50	0.50	0.61	0.522
	TabPFN	0.58	0.59	0.64	0.69	0.70	0.640
	TabLLM	0.69	0.66	0.66	0.72	0.70	**0.686**
Diabetes [45]	LightGBM	0.50	0.50	0.50	0.50	0.79	0.558
	TabPFN	0.61	0.67	0.71	0.77	0.82	**0.716**
	TabLLM	0.61	0.63	0.69	0.68	0.73	0.668
Heart [46]	LightGBM	0.50	0.50	0.50	0.50	0.91	0.582
	TabPFN	0.84	0.88	0.87	0.91	0.92	**0.884**
	TabLLM	0.76	0.83	0.87	0.87	0.91	0.848
Income [12]	LightGBM	0.50	0.50	0.50	0.50	0.78	0.556
	TabPFN	0.73	0.71	0.76	0.80	0.82	0.764
	TabLLM	0.84	0.84	0.84	0.84	0.84	**0.840**

TABLE III
Proposed Parameters for FSL Applications in LightGBM

Parameter	Description	Default	Recommended
extra_trees	use extremely randomized trees	false	True
num_leaves	max number of leaves in one tree	31	4
eta	shrinkage rate	0.1	0.05
min_data_in_leaf	**minimal number of data in one leaf**	**20**	**1**
feature_fraction	subset of features on each tree	1.0	0.5
bagging_fraction	select part of data without resampling	1.0	0.5
bagging_freq	frequency for bagging	0	1
min_data_per_group	number of data per categorical group	100	1
cat_l2	L2 regularization in categorical split	10	0
cat_smooth	reduce noise-effect in categoricals	10	0
max_cat_to_onehot	one-vs-other algorithm control	4	100
min_data_in_bin	minimal number of data inside one bin	3	3

The same concept applies to any other parameter that relies on counting of samples, such as *min_data_per_group*. In general, it is required to minimize the restrictions here, this is however a very bad practice for Non-FSL applications, leading to overfitting, and should be used with care in any other types of problems.

Due to the partition mechanism of DT, small sample-size will generate a very constrained histogram, and a greedy partition threshold is not desirable, to enhance this, the usage of extremely randomized trees is required to ensure partition splits are over represented in the tree structure.

In the next section we provide experimental results to demonstrate the ability of LightGBM to do few-shot learning.

IV. Experiments

Our experiment design covers two folds. First, we replicate previous work [43], but apply our recommended methodology to enable efficient FSL for LightGBM. Second, we bring a practical application to incorporate FSL into larger-scale data, this serves as reference that even if samples are vast, FSL can provide benefits.

A. TabLLM Experiment Replication

Both TabPFN and TabLLM show similar performance in average. Only a marginal improvement of 1% in favor of TabPFN, however, both of those solutions outperform LightGBM over 343% in average, with extreme cases such as Credit-g where the relative performance of TabLLM is 745% better. While we were able to validate these numbers are correct, our results show this extreme underperformance is driven due to incorrect parameters.

We have replicated the binary problems. For the sake of simplicity, our LightGBM does not include hyperparameter tuning and instead executed with our fixed recommended parameters as shown in Table III. This leads to intentional underoptimization to disregard the effect of better tuning in the results. We found LightGBM much more competitive as seen in Table IV.

Algorithm 1 LightGBM: feature_histogram Implementation

```
is_splittable_ = false;

//...
const auto grad = GET_GRAD(data_, t);
const auto hess = GET_HESS(data_, t);

sum_left_gradient += grad;
sum_left_hessian += hess;

left_count += cnt;

if (left_count < min_data_in_leaf) {
        continue;
}

right_count = num_data - left_count;
if (right_count < min_data_in_leaf) {
        break;
}
//...

is_splittable_ = true;
```

Our methodology improved the performance of LightGBM by 290%, essentially reducing both TabLLM and TabPFN claimed advantage by 84.5%.

LightGBM can outperform or meet TabLLM for 64-shot performance in 5 out of 6 datasets, only missing for Income dataset, where TabLLM performance is constant regardless the number of shots. This is an interesting problem to review for memorization.

For extreme low FSL, like 4 and 8 shot, we found LightGBM to be competitive, yet falling generally behind, this can further be improved with parameter tuning, but gaps are large to close still. Over 16-shots there is considerable performance parity and as the shots increase LightGBM consistently starts to take over. When enough samples are available, no performance advantages were found from TabLLM or TabPFN, yet both solutions are considerably slower to LightGBM.

B. FedCSIS 2024 Data Science Challenge

To further review performance and applications of FSL, we applied our findings to the FedCSIS 2024 Data Science Challenge hosted in the KnowledgePit platform, a web system designed for ML competitions helping to bring collaboration between industry and academia [48].

The challenge: *Predicting Stock Trends*, provides stock-tickers and their performance as measured by 116 financial-markers, such as: *Dividend Payout Ratio*, *Gross Profit Margin*, and *Price to Total Revenue per Share*. The information is provided for current Trailing Twelve Months (TTM), these are static features, named *I1* to *I58*. Another set, known as relative-features, named *dI1* to *dI58* provide the relative 1-yr change for such indicators.

This is a competition event that promotes an objective evaluation of performance. Participants were asked to predict the optimal investment strategy of securities among 3 actions: *buy*, *hold* or *sell*. An in-depth review of the competition is detailed in [49].

Initial Model: In order to establish a baseline we started our simplest possible solution directly with DT, this due to its usual superiority over other algorithms for tabular data that has not been deeply feature engineered [50]. A LightGBM regression model using all features as-is and the original discrete target *"Class"* achieves 0.8439 mean absolute error (MAE). The first insight came from feature importance, which suggests the relative (*dI**) variables far dominate the static set (*I**) as seen in Table V, taking 4 out of the top 5 spots. This inspired further review to enhance generalization given the limited data size.

Sample and Feature Selection: Following Occam's Razor principle, we challenged the value of the static features (*I**). When using all variables it's possible to get 0.6018 AUC, an alternate variant for diversification would be to use relative-features (*dI**) only, this proves to be quite competitive, retaining 95% of predictive power (0.5963 AUC), with a 50% reduction of features. This is important since the large feature mismatch promotes orthogonal decisions boundary for subsequent ensembling techniques.

Another diversification technique comes from instance sampling. We studied the sample-size vs performance in the same binary case to determine the right number of shots to use, ideally the smaller the better for diversification in further stages. Results are provided in Table VI where we can observe even after a 40% sample size reduction (6864 to 4118) there is zero impact in performance, and reducing further brings minimal degradation, this provides an ideal framework for FSL, as the ability to use few samples allows for stacking level-0 models with non-overlapping samples.

Stacking Level-0 Models: Based on previous insights, we determined that FSL is a viable strategy to enable multiple orthogonal models. Although previous analysis was done in a binary setting, these new models are built with the *Perform* target in the dataset. Unlike the discrete buy/hold/sell, this continuous representation allows the model to understand the impact of each action, e.g. not all *"buys"* are equal, since they provide different levels of financial gain/loss. Using a 3k shot-approach per model we forced diversification in the sample space. In order to improve generalization, we used the learnings that ExtraTrees outperforms GBDT in most FSL settings. We did not create any feature engineering, but our *Base Feature Set* is a concatenation of existing features over multiple years for stock-tickers that are present more than once in the dataset, only relative features (*dI**) are used. The details of each model and their respective performance is shown in Table VII. Note that because we switched to *Perform* as target, MAE is no longer optimal, so we optimized for the mean squared error (MSE) instead.

Final Blend: Our Level-1 Meta model is fed with the five different L0 configurations. *MLPRegressor* from *sklearn* was selected for simplicity, architecture is 2 hidden-layers of 10

TABLE IV
UNTUNED LIGHTGBM IMPROVED BASELINE PERFORMANCE (AUC)

Dataset	**Method**	**4-shot**	**8-shot**	**16-shot**	**32-shot**	**64-shot**	**Average**
Bank	Our LightGBM	0.54	0.62	0.65	0.70	0.77	**0.656**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.77	0.554
Blood	Our LightGBM	0.50	0.63	0.67	0.70	0.71	**0.642**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.69	0.538
Credit-g	Our LightGBM	0.60	0.64	0.62	0.65	0.70	**0.642**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.61	0.522
Diabetes	Our LightGBM	0.50	0.62	0.65	0.71	0.78	**0.652**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.79	0.558
Heart	Our LightGBM	0.78	0.85	0.88	0.90	0.91	**0.864**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.91	0.582
Income	Our LightGBM	0.60	0.68	0.77	0.81	0.83	**0.738**
	LightGBM [43]	0.50	0.50	0.50	0.50	0.78	0.556

TABLE V
COMPETITION: TOP FINANCIAL INDICATORS AS DETERMINED BY LGBM BASELINE MODEL

Feature	**Description**	**Importance**
dI58	1-year Absolute Change of Price to Cash Flow from Operations per Share	1.000
I57	Cash Flow from Operations Pct of Capital Expenditures	0.725
dI52	1-year Absolute Change of Cash Ratio	0.675
dI43	1-year Absolute Change of Dividend Yield - Common - Net - Issue - %	0.613
dI56	1-year Absolute Change of Book Value Percentage of Market Capitalization	0.537
I5	Excess Cash Margin - %	0.536
dI57	1-year Absolute Change of Cash Flow from Operations Pct of Capital Expenditures	0.521
Group	Industry sector	0.520
I24	Accounts Receivable Turnover	0.471
dI17	1-year Absolute Change of Debt - Total to EBITDA	0.404
dI44	1-year Absolute Change of PE Growth Ratio	0.377

TABLE VI
COMPETITION: SAMPLE SIZE EFFECT IN PERFORMANCE

Sample Size	**AUC**
6,864	0.6018
6,178	0.6098
5,491	0.6027
4,118	0.6055
1,373	0.5835
686	0.5887

and 5 neurons with ReLU activation [51]. Optimization is still using *Perform* target, with a 10% validation sample size and adam optimizer [52]. Early stopping is based on R2 score with 64 max epochs.

Since the competition requires discrete actions (buy/hold/sell) instead of expected performance, we optimize the performance-to-action thresholds by ensuring the same action-distribution between train and test. This solution has ranked 1^{st} place in the event, with a MAE score of 0.772, which represents a 3.66% and 7.12% relative improvement against 2^{nd} and 10^{th} place respectively.

V. CONCLUSIONS

When the merit of a proposal is measured by its relative performance to a baseline, the baseline itself is equally, or even more important than the proposal. It is trivial to show a solution is good by simply selecting a weak reference point to compare with. Efforts invested in a new proposal can also be applied to improve a baseline. In this work we have improved LightGBM FSL performance found in literature by 290%. Improvements of this magnitude are unusual with just parameter optimization.

Our results show GBDT can perform few-shot-learning (FSL) with surprising performance with as little as 8-shots. And when data is available, FSL can be used to force diversification between individual models in ensemble or stacking architectures.

While global optimum is too expensive to reach, its imperative to learn the inner caveats of algorithms to exploit their strengths to reasonable levels. Our solution in FedCSIS

TABLE VII
LEVEL-0 MODELS FOR FEDCSIS24: STOCK PREDICTION COMPETITION

Model	Features	Target	MSE
ExtraTrees	Base Feature Set	Original	0.020039
GBDT	Base Feature Set	Original	0.020051
ExtraTrees	Base Feature Set	Quantile(0.5%,99.5%)	0.019609
ExtraTrees	Base with Categorical Removed	Original	0.020088
ExtraTrees	Base with static features added back	Original	0.020055

competition shows the importance of understanding your algorithms to maximize performance, both the FSL approach for diversity and ExtraTrees to fight overfitting proved to be very successful in our experiments to achieve 1^{st} place.

REFERENCES

[1] Ravid Shwartz-Ziv and Amitai Armon, "Tabular data: Deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, 2022.

[2] Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang, "Deep & cross network for ad click predictions," 2017.

[3] Jinsung Yoon, Yao Zhang, James Jordon, and Mihaela van der Schaar, "Vime: Extending the success of self- and semi-supervised learning to tabular domain," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, Eds. 2020, vol. 33, pp. 11033–11043, Curran Associates, Inc.

[4] Yixuan Zhang, Jialiang Tong, Ziyi Wang, and Fengqiang Gao, "Customer transaction fraud detection using xgboost model," in *2020 International Conference on Computer Engineering and Application (ICCEA)*, 2020, pp. 554–558.

[5] Zifeng Wang and Suzhen Li, "Data-driven risk assessment on urban pipeline network based on a cluster model," *Reliability Engineering & System Safety*, vol. 196, pp. 106781, 2020.

[6] Xi Fang, Weijie Xu, Fiona Anting Tan, Jiani Zhang, Ziqing Hu, Yanjun Qi, Scott Nickleach, Diego Socolinsky, Srinivasan Sengamedu, and Christos Faloutsos, "Large language models(llms) on tabular data: Prediction, generation, and understanding – a survey," 2024.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017.

[8] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen, "Deberta: Decoding-enhanced BERT with disentangled attention," *CoRR*, vol. abs/2006.03654, 2020.

[9] Mingxing Tan and Quoc V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. 2021, OpenReview.net.

[11] Jiashi Li, Xin Xia, Wei Li, Huixia Li, Xing Wang, Xuefeng Xiao, Rui Wang, Min Zheng, and Xin Pan, "Next-vit: Next generation vision transformer for efficient deployment in realistic industrial scenarios," 2022.

[12] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci, "Deep neural networks and tabular data: A survey," *CoRR*, vol. abs/2110.01889, 2021.

[13] Dugang Liu, Pengxiang Cheng, Hong Zhu, Xing Tang, Yanyu Chen, Xiaoting Wang, Weike Pan, Zhong Ming, and Xiuqiang He, "Diwift: Discovering instance-wise influential features for tabular data," 2022.

[14] Tianqi Chen and Carlos Guestrin, "Xgboost: A scalable tree boosting system," *CoRR*, vol. abs/1603.02754, 2016.

[15] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[16] Anna Veronika Dorogush, Andrey Gulin, Gleb Gusev, Nikita Kazeev, Liudmila Ostroumova Prokhorenkova, and Aleksandr Vorobev, "Fighting biases with dynamic boosting," *CoRR*, vol. abs/1706.09516, 2017.

[17] Ravid Shwartz-Ziv and Amitai Armon, "Tabular data: Deep learning is not all you need," 2021.

[18] Tomaso Poggio, Andrzej Banburski, and Qianli Liao, "Theoretical issues in deep networks," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30039–30045, 2020.

[19] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux, "Why do tree-based models still outperform deep learning on tabular data?," 2022.

[20] Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian, "A comprehensive overview of large language models," 2024.

[21] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou, "Chain-of-thought prompting elicits reasoning in large language models," 2023.

[22] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang, "Sparks of artificial general intelligence: Early experiments with gpt-4," 2023.

[23] Xue Jiang, Yihong Dong, Lecheng Wang, Zheng Fang, Qiwei Shang, Ge Li, Zhi Jin, and Wenpin Jiao, "Self-planning code generation with large language models," 2023.

[24] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep Learning*, MIT Press, Cambridge, MA, USA, 2016, http://www.deeplearningbook.org.

[25] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci, "Deep neural networks and tabular data: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, p. 1–21, 2024.

[26] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah, "Wide and deep learning for recommender systems," in *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, New York, NY, USA, 2016, DLRS 2016, p. 7–10, Association for Computing Machinery.

[27] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, Xiuqiang He, and Zhenhua Dong, "Deepfm an end-to-end wide and deep learning framework for ctr prediction," 2018.

[28] Haoran Luo, Fan Cheng, Heng Yu, and Yuqi Yi, "Sdtr: Soft decision tree regressor for tabular data," *IEEE Access*, vol. 9, pp. 55999–56011, 2021.

[29] Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu, "Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2019, KDD '19, p. 384–394, Association for Computing Machinery.

[30] Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu, "TabNN: A universal neural network solution for tabular data," 2019.

[31] Sergei Ivanov and Liudmila Prokhorenkova, "Boost then convolve: Gradient boosting meets graph neural networks," 2021.

[32] Sercan O. Arik and Tomas Pfister, "Tabnet: Attentive interpretable tabular learning," 2019.

[33] Zifeng Wang and Jimeng Sun, "Transtab: Learning transferable tabular transformers across tables," 2022.

[34] Xin Huang, Ashish Khetan, Milan Cvitkovic, and Zohar Karnin, "Tabtransformer: Tabular data modeling using contextual embeddings," 2020.

[35] Gowthami Somepalli, Micah Goldblum, Avi Schwarzschild, C. Bayan Bruss, and Tom Goldstein, "Saint: Improved neural networks for tabular data via row attention and contrastive pre-training," 2021.

[36] Jannik Kossen, Neil Band, Clare Lyle, Aidan N. Gomez, Tom Rainforth, and Yarin Gal, "Self-attention between datapoints: Going beyond individual input-output pairs in deep learning," 2022.

[37] Omurhan A. Soysal and Mehmet Serdar Guzel, "An introduction to zero-shot learning: An essential review," in *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, 2020, pp. 1–4.

[38] Ruiyu Wang, Zifeng Wang, and Jimeng Sun, "Unipredict: Large language models are universal tabular classifiers," 2024.

[39] Sebastian Bordt, Harsha Nori, and Rich Caruana, "Elephants never forget: Testing language models for memorization of tabular data," 2024.

[40] Chaofan Chen, Kangcheng Lin, Cynthia Rudin, Yaron Shaposhnik, Sijia Wang, and Tong Wang, "An interpretable model with globally consistent explanations for credit risk," 2018.

[41] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, Alina Oprea, and Colin Raffel, "Extracting training data from large language models," 2021.

[42] Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang, "Quantifying memorization across neural language models," 2023.

[43] Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag, "Tabllm: Few-shot classification of tabular data with large language models," 2023.

[44] Arlind Kadra, Marius Lindauer, Frank Hutter, and Josif Grabocka, "Well-tuned simple nets excel on tabular datasets," 2021.

[45] Jack Smith, J. Everhart, W. Dickson, W. Knowler, and Richard Johannes, "Using the adap learning algorithm to forcast the onset of diabetes mellitus," *Proceedings - Annual Symposium on Computer Applications in Medical Care*, vol. 10, 11 1988.

[46] Steinbrunn William Pfisterer Matthias Janosi, Andras and Robert Detrano, "Heart Disease," UCI Machine Learning Repository, 1988, DOI: https://doi.org/10.24432/C52P4X.

[47] Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter, "Tabpfn: A transformer that solves small tabular classification problems in a second," 2023.

[48] Sebastian Stawicki Andrzej Janusz, Dominik Slezak and Mariusz Rosiak, "Data-driven risk assessment on urban pipeline network based on a cluster model," *Proceedings of the 24th International Workshop on Concurrency, Specification and Programming*, 2015.

[49] Ivana T. Dragovic Ana M. Poledica Milica M. Zukanovic Andrzej Janusz Dominik Slezak Aleksandar M. Rakicevic, Pavle D. Milosevic, "Predicting stock trends using common financial indicators: A summary of fedcsis 2024 data science challenge held on knowledgepit.ai platform," *Proceedings of FedCSIS 2024*, 2024.

[50] C. Huertas and Q. Zhao, "On the irrelevance of machine learning algorithms and the importance of relativity," in *2023 IEEE International Conference on Multimedia and Expo Workshops (ICMEW)*, Los Alamitos, CA, USA, jul 2023, pp. 16–21, IEEE Computer Society.

[51] Abien Fred Agarap, "Deep learning using rectified linear units (relu)," 2019.

[52] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," 2017.