Leveraging Structural Information in Tree Ensembles for Table Representation Learning

Nikhil Pattisapu, Siva Rajesh Kasa, Sumegh Roychowdhury, Karan Gupta, Anish Bhanushali, Prasanna Srinivasa Murthy
{npattisa,kasasiva,sumegr,karaniis}@amazon.com
Amazon, India

Abstract

Tabular data is one of the most common data formats found in the web and used in domains like finance, banking, e-commerce and medical. Although deep neural networks (DNNs) have demonstrated outstanding performance on homogeneous data such as visual, audio, and textual data, tree ensemble methods such as Gradient Boosted Decision Trees (GBDTs) are often the go-to choice for supervised machine learning problems involving heterogeneous tabular data. However, a major limitation of these methods lies in the difficulty of plugging-in other modalities (like text, images), as is achievable with deep learning (DL) models. To bridge this gap, researchers have put forth a multitude of DL approaches tailored specifically for tabular data. In this work, we propose a new **path embedding-based method** to harness the structural information from tree ensembles to improve tabular data representation. Our approach not only demonstrates superior performance compared to existing DL models for tabular classification tasks but also outperforms competitive baselines when combined with textual data in multimodal tabular transformers.

ACM Reference Format:
Nikhil Pattisapu, Siva Rajesh Kasa, Sumegh Roychowdhury, Karan Gupta, Anish Bhanushali, Prasanna Srinivasa Murthy. 2024. Leveraging Structural Information in Tree Ensembles for Table Representation Learning. In . ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Deep Learning (DL) methods have demonstrated unprecedented performance on predictive and generative tasks involving homogeneous data such as text, audio and video. In contrast, their performance lags behind the classic tree ensemble methods such as XGBoost [6], CatBoost [26] on several tabular datasets comprising of heterogeneous data involving sparse categorical (ordinal, nominal) and dense numeric features [23]. Researchers have identified several reasons for this phenomenon. Firstly, deep learning (DL) models are susceptible to the nature and quality of training data. Tabular datasets often contain missing values, outliers, erroneous data, label imbalance, skewed or heavy tailed feature distributions and the number of labeled examples are often small compared to the dimensionality of the feature vectors [23]. While most of these characteristics are handled well by the tree-based methods, DL models often struggle with it. Secondly, tabular datasets have either missing or complex spatial dependencies which means there is little or no spatial correlation between features in these datasets. Therefore, the inductive biases present in DL models such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) which are well-suited for homogeneous datasets are often unsuitable for tabular data [4, 14]. Thirdly, modeling homogeneous data includes an implicit representation learning and involves minimal preprocessing. In contrast, DL methods for tabular data are highly sensitive to the data preparation and preprocessing steps such as feature imputation, feature normalization and the methods employed for handling categorical features. Lastly, tabular datasets' target variable could be dependent on single feature. For instance, a single categorical feature change could cause the predicted class to flip; whereas in homogeneous data such as images, it would take a coordinated change in many pixels to change the class.

Why use DL ? - Nevertheless, researchers have continued to invest in developing DL models for tabular data, driven by their adaptability to multimodal settings, where tabular data complements input modalities like text & images, which wouldn't be possible in tree-based models like GBDT. Also unlike GBDTs, DL models support iterative training, i.e. when new batch of data becomes available, they can be fine-tuned on the new data, without having to retrain the entire model from scratch. These factors have lead to the resurgence in DL methods for tabular data [11, 13, 16, 17]. However, developing methods to represent tabular data in higher dimensions—while effectively capturing diverse feature distributions and outliers—remains a significant challenge. Additionally, there are numerous design options for integrating these representations with existing text-based deep learning embeddings to efficiently capture cross-modal interactions between textual and tabular data (further discussed in Section 2). In order to address these challenges, we propose an approach that combines the strengths of both paradigms (GBDT & DL). We propose a new path embedding-based method to harness the structural information from tree ensembles to improve tabular data representation. This representation learning strategy also enables seamless integration with textual modality, thereby improving performance on multimodal classification tasks. In summary, our main contributions can be outlined as follows:

- We introduce a novel method `TreeTransformer` that harnesses the structural information derived from trained tree-ensemble (GBDT here) models, enabling the generation of a homogeneous representation for tabular datasets.
- We conduct a comprehensive comparison of Deep Learning Methods for Tabular data across 41 publicly available

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

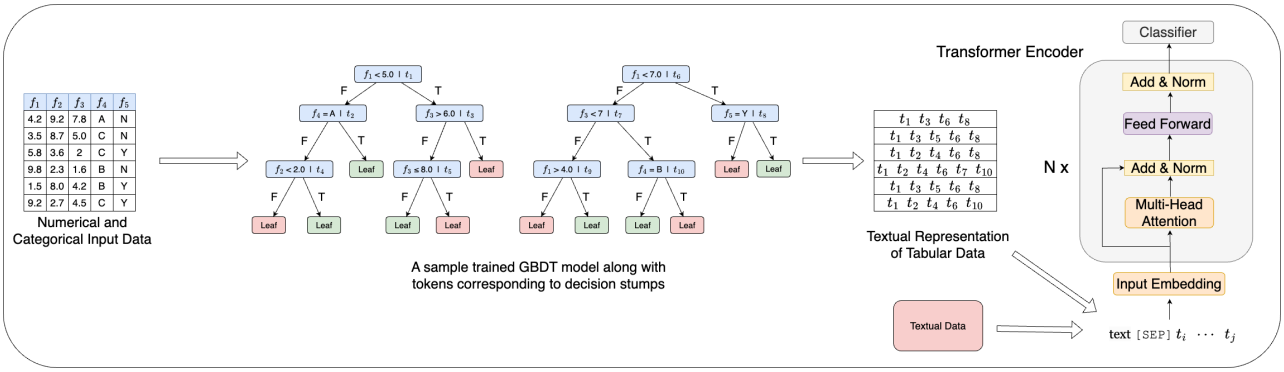

Figure 1: [Best viewed in color] `TreeTransformer` architecture.

OpenML datasets [1]. Through our evaluation, we demonstrate that our proposed method outperforms prior state-of-the-art DL models, including those that utilize representations obtained from GBDTs.

- We also show that using `TreeTransformer` can improve performance over other popular cross-modality combination techniques on 5 multimodal classification tasks [27] involving both text and numeric/categorical data.

2 Related Works

Tabular Data Approaches: A large group of prior approaches propose using customized popular deep learning architectures such as MLP [9, 11, 13], ResNets [13, 19], Transformers [11, 13, 17, 28], Convolutional Neural Networks [32] and Graph Convolutional Networks [8] for tabular learning. [17, 29] were among the first to employ a transformer encoder for this task. We suggest referring to [5] for more details. Improving upon these, another line of work, more similar to ours, employ both decision tree ensembles and DL models for tabular data modeling. [11] propose an approach where they first train a decision tree and extract (feature, value) tuples from the decision stumps. They use these tuples to split the range of every numerical feature into disjoint set of multiple intervals/bins. Then each of these bins are assigned trainable embeddings which are aggregated and passed to neural architectures like MLP or Transformer. Next `DeepGBM` [18] used an ensemble of GBDT and DL methods for tabular data classification specifically by passing the leaf nodes of trained GBDT model as input to their DL model, but do not leverage the internal (non-leaf) nodes. Another approach `DeepTLF` develop a novel knowledge distillations procedure, called *TreeDrivenEncoder*, which given an input feature vector x, visits the inner nodes of every tree of the trained GBDT model and exploits the decision tree's boolean functions to construct a binary feature vector. The resultant representation is fed to a MLP. They show that this approach outperfoms several competetive baselines such as `DeepGBM` [18], TabTransformer [17] and `NODE` [25]. [20] extend the `DeepTLF` framework, and utilize pretrained tree ensembles to transform raw variables into binarized embeddings.

We choose the best approaches from the above mentioned works as baselines based on their performance on TabZilla Benchmarks [23]. Another parallel line of work includes `TabPFN` [16] and `TabR` [12] which utilize data-lookup during inference. However their quadratic runtime and memory costs make them unsuitable for large datasets. Hence, we don't consider these methods in our study.

Multimodal Approaches: With the rise of multimodal transfer learning [7, 24] in DL, researchers have introduced various methods to fuse multiple modalities such as text, images, and audio [30] to enhance downstream performance. These methods can generally be categorized into two main approaches: early fusion and late fusion, often incorporating cross-modal interaction mechanisms (e.g., co-attention [22] or hierarchical attention [21]) at different stages. In late fusion [15], interactions between modalities occur near the final layers, whereas early fusion enables these interactions across all intermediate layers. Although less explored due to the inherently heterogeneous nature of tabular data, recent studies have begun investigating early fusion techniques [2, 27, 31] for modeling text + tabular data. In this work, our focus is to propose a simple but novel architecture-agnostic approach for encoding tabular data to get homogeneous representations which can be plugged into any of these above state-of-the-art multimodal architectures to obtain performance gains.

3 Proposed Approach

Algorithm 1: Tabular Encoding Scheme used in `TreeTransformer`

Input: List of decision trees `trees` from trained GBDT
Output: String of traversed node tokens separated by [*SEP*]

```
Initialize result ← [];
for each tree T in trees do
    Initialize an empty list traversed_nodes;
    Set current_node ← T.root_node;
    while current_node is not a leaf do
        result ← result + current_node.token;
        if current_node.condition is true then
            Set current_node ← current_node.right_child;
        else
            Set current_node ← current_node.left_child;
    result ← result + [SEP];
return result;
```

Figure 1 depicts our approach in a nutshell. We first train a GBDT model (here XGBoost [6]) using numerical and categorical features that is subsequently used to obtain the homogeneous tabular representation corresponding to any input data. Each decision stump in the trained GBDT model is assigned a unique token. For a given input data point we run the inference on the trained GBDT model while tracing the tokens encountered during traversal following *Algorithm 1*. The resultant sequence of discrete tokens is used as the textual representation for the input sample. [*SEP*] tokens are added so as to help the DL model discriminate between tokens corresponding to different trees. The resultant sequence length scales with $O(T * log(N))$ where T is the number of trees in ensemble and N is the total nodes. The hyperparameters are tuned using Optuna library to ensure maximum length doesn't exceed 128. In case of modeling tabular data, these textual tokens are mapped to vectors using a trainable embedding matrix lookup operation. The resultant embeddings are passed to a BERT-styled transformer encoder block with a classification head which is trained using binary cross-entropy loss. We found that our model performance is more sensitive to the tree-based hyperparameters than the DL ones.

Our approach can be seamlessly extended to the text + tabular multimodal use case, where the input consists of a concatenation of tokens from textual fields and the special tokens outlined in *Algorithm 1*. In the text + tabular use case, leveraging pretrained models like BERT allows us to utilize their deep language understanding capabilities. To seamlessly integrate the special tokens required for handling tabular fields, we extend BERT's embedding matrix to include these new tokens as part of the vocabulary. This ensures that the model can process both the standard textual input and the additional tokens representing tabular features. The new tokens are initialized with random embeddings, allowing the model to adapt during fine-tuning without disrupting its pretrained knowledge. This approach enables a unified representation of text and tabular data while retaining the benefits of transfer learning.

Unlike several previous approaches discussed earlier our approach results in a target-aware tree representation (since GBDT is trained using the same ground truth label) which inherently possesses the benefits of feature selection. Since we directly leverage the learned tree ensembles here, this also mitigates the usual challenges of encoding tabular data using DL models due to irregular feature distributions, missing values, etc. Our approach is inspired from both `DeepTLF` [3] and `TreeToToken` [20] approaches discussed in Section 2. We explicitly state the difference here. Firstly, while both `DeepTLF` and `TreeToToken` utilize the nodes of trained GBDT models, neither of these methods encode the nodes using trainable embeddings. In contrast, our approach employs a trainable embedding matrix that is randomly initialized, with each node corresponding to a unique embedding. According to experiments conducted in [11], embedding binary features has exhibited superior performance compared to using binarized raw feature vectors directly in the model. Secondly, to achieve a homogeneous representation of the input data, both `DeepTLF` and `TreeToToken` consider tree ensembles as collections of nodes, **disregarding the structural information** inherent in these trained trees, such as the depth of the node, the sequence of nodes from root to leaf, and the relationship of nodes within the tree. Whereas our proposed method leverages this information. Another advantage of our encoding method is that since it relies on the path from the root node to the leaf node, the input length scales with $O(log(N))$. In contrast, the input length of `DeepTLF` or `TreeToToken` scales with $O(N)$. As a result, our method requires less time for training and inference when dealing with transformer models, whose time complexity grows quadratically with the length of the input sequence.

4 Results and Performance Comparison

Table 1: Performance comparison with baselines on 41 *GBDT-friendly* benchmark datasets for binary classification, reporting the rank and AUC of each approach averaged across 10 folds.

		Rank ↓				AUC ↑		Std. AUC	
Algorithm	**Category**	min	max	mean	med.	mean	med	mean	med
XGBoost [6]	GBDT	1	5	1.60	1	0.89	0.92	0.03	0.01
ResNet [13]	DL	1	4	3.7	4	0.85	0.87	0.04	0.02
SAINT [28]	DL	1	7	3.58	3	0.83	0.86	0.04	0.02
FTTransformer [13]	DL	1	7	4.28	5	0.86	0.90	0.04	0.02
TreeToToken [20]	Tree-rep + DL	1	7	3.58	3	0.83	0.86	0.05	0.02
DeepTLF [3]	Tree-rep + DL	1	7	5.36	6	0.83	0.85	0.04	0.02
TreeTransformer	Tree-rep + DL	1	6	3.07	3	0.88	0.91	0.04	0.02

Table 2: Performance comparison with baselines on 11 *hard* benchmark datasets for binary classification, reporting the rank and AUC of each approach averaged across 10 folds.

		Rank ↓				AUC ↑		Std. AUC	
Algorithm	**Category**	min	max	mean	med.	mean	med	mean	med
XGBoost [6]	GBDT	1	3	1.27	1	0.88	0.92	0.03	0.03
ResNet [13]	DL	2	7	4.18	4	0.85	0.85	0.03	0.03
SAINT [28]	DL	1	7	4.00	3	0.84	0.85	0.03	0.03
FTTransformer [13]	DL	1	6	3.20	3	0.86	0.89	0.03	0.03
TreeToToken [20]	Tree-rep + DL	3	7	5.27	6	0.84	0.86	0.03	0.03
DeepTLF [3]	Tree-rep + DL	4	7	6.27	7	0.83	0.85	0.04	0.03
TreeTransformer	Tree-rep + DL	2	5	3.27	3	0.87	0.90	0.03	0.03

Table 3: Performance comparison on various datasets, reporting the AUC for different approaches averaged across 10 folds.

	imdb	**jigsaw**	**kick**	**airbnb**	**channel**
All Text	0.8043	0.9618	0.7776	0.7922	0.6764
Early Fusion	0.7952	0.9623	0.7611	0.7859	0.6069
Late Fusion	0.8134	0.9626	0.7614	0.790	0.6073
`TreeTransformer`	**0.8213**	**0.9692**	**0.7863**	**0.8019**	**0.7485**

Tabular Data Results: For reporting results, we use 41 binary classification *GBDT-friendly* datasets from the TabZilla Benchmark [23] where DL models usually underperform traditional tree-based approaches like GBDT owing to their irregular feature distributions. This choice is made because for DL-friendly datasets where prior DL approaches outperform GBDTs it's a solved problem already. Table 1, shows that our proposed approach `TreeTransformer` significantly outperforms prior tree-based DL approaches `TreeToToken` and `DeepTLF` in terms of both mean rank and mean AUC of each approach across all datasets. Furthermore, it surpasses other non-tree DL baselines such as `SAINT`, `FTTransformer` and `ResNet`. This validates our initial hypothesis that leveraging the tree structure and paths, while keeping trainable node embeddings (which have been overlooked in previous approaches), aids our model in learning superior tree representations. We notice that the tree-based DL baselines `TreeToToken` and `DeepTLF` typically exhibit the lowest performance while our proposed approach performs the best. This underscores the fact that with the right technique, representations

extracted from GBDTs are useful for tabular tasks. The TabZilla benchmarks also identify 36 datasets from a mix of both DL-friendly and GBDT-friendly datasets as *hard* due to the consistently low performance of most algorithms. Among these, we focus only on the binary classification tasks and compare against baseline methods in Table 2. `TreeTransformer`, consistently achieves the highest mean AUC, outperforming other deep learning baselines. However, in terms of mean rank, *FTTransformer*, a pure DL approach slightly edges out on the DL-friendly datasets by a very narrow margin (in the third/fourth decimal place), improving its rank. Overall, our approach helps narrow the performance gap between XGBoost and DL models on both GBDT/DL-friendly datasets. We next demonstrate the benefits of applying DL techniques to tabular data in multimodal scenarios, where tree-based models often lag behind pre-trained DL methods [27].

Multimodal Results: For our analysis, we selected five datasets from the collection of binary and multiclass classification datasets in [27] having more than five numerical features (for the learned GBDT model to provide meaningful representation). As outlined in Section 2, we compared our proposed approach with popular modality-fusion techniques - (a) *Early Fusion* - similar to xVal [10], we maintain a single trainable embedding for numerical features, scaling it by their corresponding magnitudes, and appending it to the textual and categorical data before feeding it into the transformer encoder stack. (b) *Late Fusion* - numerical and categorical features were appended to the [CLS] token embedding from the transformer and then passed through classifier and (c) *All Text* - numerical and categorical features were converted to strings and were directly appended to the textual data. As shown in Table 3, our proposed method outperformed all baselines on all 5 datasets.

Conclusion: We find that our novel path-based tabular encoding approach surpasses previous state-of-the-art techniques for tabular encoding, including the tree+DL methods. Furthermore, integrating our architecture-agnostic approach with DL models outperforms other widely-used techniques for combining modalities, showing the usefulness of tree-based representations. Future work could easily extend this approach to audio and visual modalities.

References

[1] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Pieter Gijsbers, Frank Hutter, Michel Lang, Rafael G Mantovani, Jan N van Rijn, and Joaquin Vanschoren. 2017. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731* (2017).

[2] Thomas Bonnier. 2024. Revisiting Multimodal Transformers for Tabular Data with Text Fields. In *Findings of the Association for Computational Linguistics ACL 2024*. 1481–1500.

[3] Vadim Borisov, Klaus Broelemann, Enkelejda Kasneci, and Gjergji Kasneci. 2023. DeepTLF: robust deep neural networks for heterogeneous tabular data. *International Journal of Data Science and Analytics* 16, 1 (2023), 85–100.

[4] Vadim Borisov, Tobias Leemann, Kathrin Seßler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2022. Deep neural networks and tabular data: A survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[5] Vadim Borisov, Tobias Leemann, Kathrin Sessler, Johannes Haug, Martin Pawelczyk, and Gjergji Kasneci. 2022. Deep Neural Networks and Tabular Data: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* (2022), 1–21. https://doi.org/10.1109/tnnls.2022.3229161

[6] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2* 1, 4 (2015), 1–4.

[7] Guilherme Lourenço de Toledo and Ricardo Marcondes Marcacini. 2022. Transfer learning with joint fine-tuning for multimodal sentiment analysis. *arXiv preprint arXiv:2210.05790* (2022).

[8] Lun Du, Fei Gao, Xu Chen, Ran Jia, Junshan Wang, Jiang Zhang, Shi Han, and Dongmei Zhang. 2021. TabularNet: A neural network architecture for understanding semantic structures of tabular data. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. 322–331.

[9] James Fiedler. 2021. Simple modifications to improve tabular neural networks. *arXiv preprint arXiv:2108.03214* (2021).

[10] Siavash Golkar, Mariel Pettee, Michael Eickenberg, Alberto Bietti, Miles Cranmer, Geraud Krawezik, Francois Lanusse, Michael McCabe, Ruben Ohana, Liam Parker, et al. 2023. xval: A continuous number encoding for large language models. *arXiv preprint arXiv:2310.02989* (2023).

[11] Yury Gorishniy, Ivan Rubachev, and Artem Babenko. 2022. On Embeddings for Numerical Features in Tabular Deep Learning. In *NeurIPS*.

[12] Yury Gorishniy, Ivan Rubachev, Nikolay Kartashev, Daniil Shlenskii, Akim Kotelnikov, and Artem Babenko. [n. d.]. TabR: Tabular Deep Learning Meets Nearest Neighbors. ([n. d.]).

[13] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. 2021. Revisiting Deep Learning Models for Tabular Data. In *NeurIPS*.

[14] Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux. 2022. Why do tree-based models still outperform deep learning on typical tabular data? *Advances in neural information processing systems* 35 (2022), 507–520.

[15] Ken Gu and Akshay Budhkar. 2021. A package for learning on tabular and text data with transformers. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*. 69–73.

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

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

[18] Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu. 2019. 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*. 384–394.

[19] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. 2017. Self-normalizing neural networks. *Advances in neural information processing systems* 30 (2017).

[20] Xuan Li, Yun Wang, and Bo Li. 2023. Tree-Regularized Tabular Embeddings. In *NeurIPS 2023 Second Table Representation Learning Workshop*. https://openreview.net/forum?id=dQLDxIPsU4

[21] Junyang Lin, An Yang, Yichang Zhang, Jie Liu, Jingren Zhou, and Hongxia Yang. 2020. Interbert: Vision-and-language interaction for multi-modal pretraining. *arXiv preprint arXiv:2003.13198* (2020).

[22] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems* 32 (2019).

[23] Duncan McElfresh, Sujay Khandagale, Jonathan Valverde, Vishak Prasad C, Benjamin Feuer, Chinmay Hegde, Ganesh Ramakrishnan, Micah Goldblum, and Colin White. 2023. When Do Neural Nets Outperform Boosted Trees on Tabular Data? arXiv:2305.02997 [cs.LG]

[24] Ricardo Montalvo-Lezama, Berenice Montalvo-Lezama, and Gibran Fuentes-Pineda. 2022. Improving Transfer Learning with a Dual Image and Video Transformer for Multi-label Movie Trailer Genre Classification. *arXiv preprint arXiv:2210.07983* (2022).

[25] Sergei Popov, Stanislav Morozov, and Artem Babenko. 2019. Neural oblivious decision ensembles for deep learning on tabular data. *arXiv preprint arXiv:1909.06312* (2019).

[26] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. *Advances in neural information processing systems* 31 (2018).

[27] Xingjian Shi, Jonas Mueller, Nick Erickson, Mu Li, and Alexander J Smola. 2021. Benchmarking multimodal automl for tabular data with text fields. *arXiv preprint arXiv:2111.02705* (2021).

[28] 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).

[29] Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. 2019. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 1161–1170.

[30] Peng Xu, Xiatian Zhu, and David A Clifton. 2023. Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 10 (2023), 12113–12132.

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

[32] Yitan Zhu, Thomas Brettin, Fangfang Xia, Alexander Partin, Maulik Shukla, Hyunseung Yoo, Yvonne A Evrard, James H Doroshow, and Rick L Stevens. 2021. Converting tabular data into images for deep learning with convolutional neural networks. *Scientific reports* 11, 1 (2021), 11325.