IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 5 January 2024, accepted 10 January 2024, date of publication 16 January 2024, date of current version 23 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3354972

RESEARCH ARTICLE

LF-Transformer: Latent Factorizer Transformer for Tabular Learning

KWANGTEK NA[1], JU-HONG LEE[1], AND EUNCHAN KIM[2,3]
[1]Department of Electrical and Computer Engineering, Inha University, Incheon 22212, Republic of Korea
[2]Department of Intelligence and Information, Seoul National University, Seoul 08826, Republic of Korea
[3]College of Business Administration, Seoul National University, Seoul 08826, Republic of Korea

Corresponding author: Ju-Hong Lee (juhong@inha.ac.kr)

ABSTRACT The field of deep learning for tabular datasets has made significant strides in recent times. Previously, gradient boosting and decision tree algorithms had been the go-to options for processing such datasets due to their superior performance. However, deep learning has now reached a level of development where it can compete with these algorithms on equal footing. Accordingly, we propose latent factorizer transformer (LF-Transformer). Our proposing method, LF-Transformer, involves applying the transformer architecture to columns and rows of a given dataset to identify the attention latent factor matrix. This matrix is then used for prediction. The process is akin to matrix factorization, which involves breaking down the original matrix into a latent matrix and then reconstructing it again. Our experimental results indicate that the LF-Transformer approach outperforms general feature embedding methods, providing superior feature presentation. Additionally, the approach has demonstrated a relative superiority in regression and classification across various datasets that we have tested. In conclusion, the LF-Transformer presents a promising direction for deep learning approaches in tabular datasets. Its ability to identify latent factors and provide superior performance in regression and classification makes it a compelling alternative to traditional algorithms.

INDEX TERMS Tabular dataset, tabular learning, self-attention, row-wise attention, column-wise attention, matrix factorization.

I. INTRODUCTION

The prevailing opinion is that there is no field that deep learning cannot penetrate. Deep neural network models, including ensemble models and their derivatives, have been evaluated as efficient and effective in various domains including text, audio, and images [1], [2], [3], [4], [5]. Furthermore, the tabular dataset has been evaluated as an unconquered castle [6]. Certainly, deep learning cannot be used to solve all the problems. According to some claims, most tabular data models, which have recently received attention from the research community, have shown performances inferior to that of extreme gradient boosting (XGBoost) [7] and light gradient boosting machine (LightGBM) [8] with significantly less tuning [9]. It has even been argued that deep learning is not an absolute necessity for tabular data [9]. Several other studies [7], [10], [11] have supported the claims of Shwartz-Ziv and his colleague. These studies have supported the use and consideration of tree-based ensemble algorithms such as XGBoost and LightGBM for cases involving real-life tabular datasets. It was recently suggested that TabNet [6] is superior to the gradient boosted decision tree (GBDT) algorithms for tabular datasets. However, at least as a result of implementing and experimenting with TabNet, the performance presented in the original study [6] was not properly reproduced, and it can be seen through the experimental results in [12].

In this competitive state-of-the-art algorithm development on tabular datasets, we discuss the process of improvement and development of a new model called the latent factorizer transformer (LF-Transformer) based on the feature tokenizer transformer (FT-Transformer) [12] and matrix factorization [13] in this study. The codes and datasets related to our proposed model (LF-Transformer) and research are publicly available on

The associate editor coordinating the review of this manuscript and approving it for publication was Ines Domingues.

our GitHub project (URL: https://github.com/kwangtekNa/LF-Transformer/ accessed on 5 January 2024.).

The LF-Transformer we propose, which combines the matrix factorization idea with the transformer idea, has the following contributions.

Contribution 1: Column and row transformer enables capturing not only the attention relationship between features (columns) but also the attention relationship between data (rows).

Contribution 2: Our idea of reconstructing the data matrix by embedding columns and rows into the latent space by applying the matrix factorization concept has shown good results in feature representation.

Contribution 3: We designed a new architecture called LF-Transformer and had a relative superiority in regression and classification in many of the datasets we tested.

II. RELATED WORKS

Traditional machine learning methods such as gradient-boosted decision trees (GBDTs) and tree-based ensemble models [14], [15] are still widely used for tabular data modeling [9]. However, several attempts have been made to use deep learning with tabular data. These attempts such as TabNet [6], oblivious decision ensembles (NODE) [16], and disjunctive normal form-based method (Net-DNF) [17] have shown better performances than that of GBDTs. However, since the aforementioned studies did not employ standard benchmarks and used different datasets, significant model and parameter tuning will be required when using these models while ensuring that the various problems associated with deep neural networks (such as lack of locality, missing input data, data type problems, etc.) are avoided. Thus, it is evident that extensive work must be done with regard to fine tuning when using these models. In this sense, we briefly outline the key ideas of tabular models that are relevant to our study. We then present a review as well as a brief explanation of our study motivation. Finally, we present our methodology (LF-Transformer) combining the idea of FT-Transformer and matrix factorization.

A. TABULAR DATA LEARNING

NODE: The NODE model [16] primarily ensures that the error gradient is effectively backpropagated. Differentiable oblivious decision trees are set up to perform this task, and similar to traditional tree models, their performance on data segmentation and training for a selection function is decided based on a threshold. Only one function is used at each level, resulting in a balanced tree for differentiation. The final model is presented in the form of a differentiable ensemble.

TabNet: TabNet [6] is a representative and popular model that combines deep learning with attentive mechanism for tabular dataset presented by Google AI. This model has been proven to perform well on multiple datasets. TabNet involves a sequential decision step wherein an encoder encodes each corresponding feature using sparse learning and selects a relevant feature for each row using an attention mask. In encoding, a small set of features is forcibly selected, and the sparsemax layer is used. A typical advantage is that there is no need to select all or none of the features during the learning process, and instead of using a threshold value for each feature, the learnable mask can deliver a softer decision. This problem can also be mitigated through the use of a feature selection method.

FT-Transformer: The FT-Transformer has been proposed by [12]. The authors apply the transformer architecture [18] to perform regression and classification on a tabular dataset. The FT-Transformer embeds both categorical and numerical features in a d-dimensional vector space. In addition, the final representation of classification token (CLS token) is used for final prediction by inserting the CLS token into the embedding vector [12].

SAINT: As far as we know, the self-attention and intersample attention transformer (SAINT) [19] is the first model to apply row attention to tabular datasets, which is referred to as inter-sample attention in the paper. SAINT performs self-attention between columns and then applies inter-sample attention sequentially to the output of self-attention. As per the authors, the application of inter-sample attention results in the following effect: In instances where a particular feature is absent or contains noise in a given row, SAINT is empowered to utilize intersample attention for borrowing the relevant features from other data samples that are similar in the batch.

III. MOTIVATION

Matrix factorization [13] is a method of approximating the original data matrix by mapping each column and row of the data matrix to the latent feature space, and by multiplying them again. It is mainly used in recommendation systems or information retrieval. When considering a movie recommendation system, each row is assigned to a user and each column is assigned to a movie title. At this moment, matrix factorization decomposes this matrix which consists of users (rows) $\times$ movie titles (columns), and maps each to a latent feature space. Its performance varies depending on the number of dimensions of the latent feature space determined by the researcher. Latent vectors' each dimension of users and movie titles is mapped to the latent feature space, and can be analyzed through latent semantic analysis (LSA). In this kind of movie recommender system, each dimension of the latent space can be interpreted as the degree of sci-fi genre propensity, the degree of romance propensity, and etc. The matrix factorization can grasp the characteristics of data that cannot be confirmed in the original data, and also has an effect of removing noise. We applied this idea of matrix factorization to transformer. The purpose of the column transformer is to generate column attention latent features using the attention information between columns, and the row transformer uses the attention information between rows to create row attention latent features and multiplies them again to identify information that cannot be confirmed in the original data.

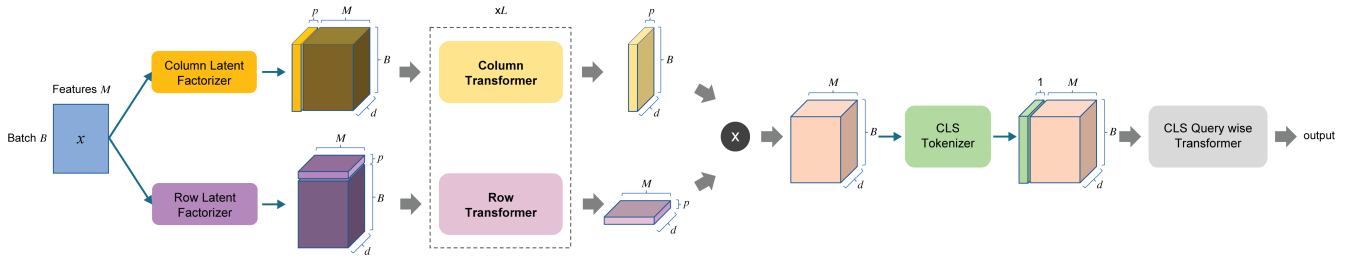


FIGURE 1. Architecture of LF-Transformer.
Note: First, the original input data is embedded and inserted into a d-dimensional space, and a latent factor is inserted to learn column and row information. Afterwards, it goes through the column transformer and row transformer respectively to obtain the column latent matrix with column information and row latent matrix with row information. These two matrices are multiplied again to produce the final embedded matrix, and the CLS token is inserted into it to obtain the final result.

IV. PROPOSED MODEL

As depicted in Figure 1, the structure of our proposed model applies the concepts of FT-Transformer and matrix factorization. A transformer is applied to each column and row, resulting in the generation of column attention latent factor matrix and row attention latent factor matrix. These matrices are multiplied together to create the final embedding matrix. The embedding matrix created at this time is a matrix created after mapping column information and row information to latent space, and removes noise, which has the effect of clarifying. A detailed analysis of this is covered in the analysis section. Then, after inserting the CLS token to be used for prediction into the latent embedding matrix, the final output is obtained through the CLS query-wise transformer.

A. LATENT FACTORIZER

The latent factorizer module embeds the input feature in the d-dimensional space and adds a latent factor so that the final representation vector of the input feature appears in the Latent space. If f is the function embedding the input features x which is one data point in the batch. it can be expressed as follows.

$$T_j = b_j + f_j(x_j), \quad f_j : X_j \to R^d \tag{1}$$

Here, x_j represents j-th, that is the input feature, b_j represents the bias, T_j represents the d-dimensional embedding vector of x_j. We refer to the study Gorishniy et al. [12], the numerical feature, $f^{(num)}$ can be expressed as an element-wise product of the weight vector, $W_j^{(num)} \in R^d$, and the categorical feature, $f^{(cat)}$ and $W_j^{(cat)}$ are selected by each corresponding vector from the look-up table $W_j^{(cat)} \in R^{s_j \times d}$.

$$T_j^{(num)} = b_j^{(num)} + x_j^{(num)} W_j^{(num)} \quad \in R^d \tag{2}$$

$$T_j^{(cat)} = b_j^{(cat)} + e_j^T W_j^{(cat)} \quad \in R^d \tag{3}$$

$$T = stack[T_1^{(num)}, \ldots, T_{k^{(num)}}^{(num)}, T_1^{(cat)}, \ldots, T_{k^{(cat)}}^{(cat)}]$$
$$T \in R^{k \times d} \tag{4}$$

Afterwards, in order to generate the latent factor matrix, we insert the latent factor F into the embedding vector T.

$$T = stack[F, T_1^{(num)}, \ldots, T_{k^{(num)}}^{(num)}, T_1^{(cat)}, \ldots, T_{k^{(cat)}}^{(cat)}]$$
$$T \in R^{(k+p) \times d}, \quad F \in R^{p \times d} \tag{5}$$

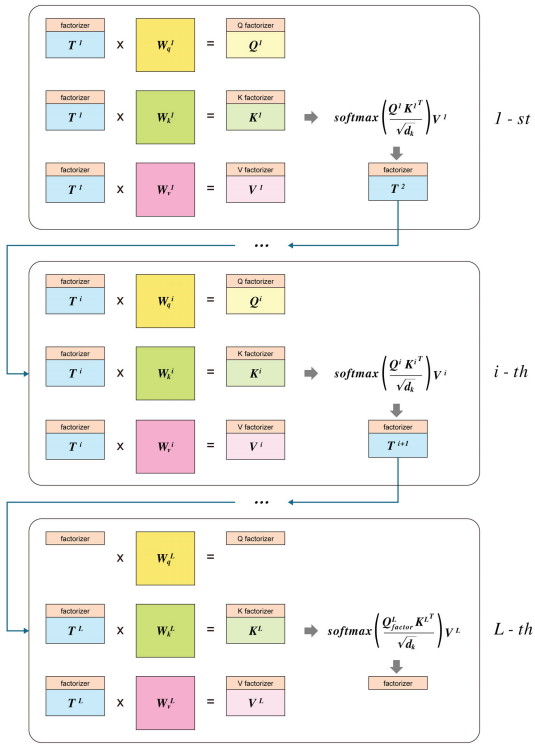
In $F \in R^{p \times d}$, the dimension, p is the dimension of the latent factor which is hyper parameter being embedded in the latent space.

B. COLUMN AND ROW TRANSFORMER

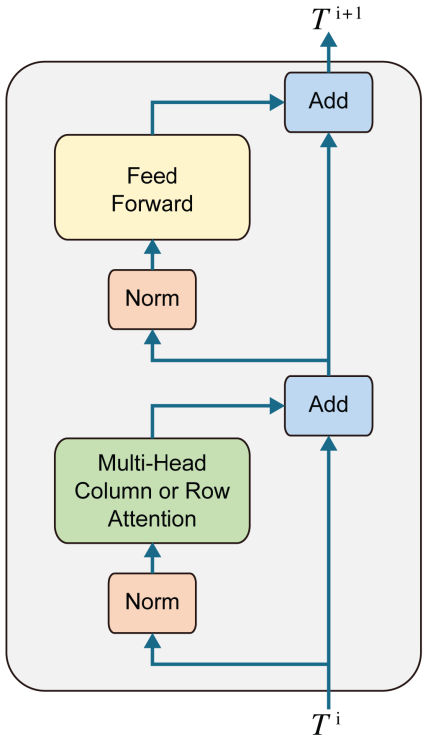
The transformer layer we utilize is based on the pre-normalization (PreNorm) transformer of the study Gorishniy et al. [12]. In (a) of Figure 2, it consists of a total of the L number of transformer layers and operates like a general transformer up to the $L - 1_{th}$ layer, but only the query latent factor is input as the query input in the last layer, L_{th} Layer. Therefore, the final output is a latent factor matrix that has learned from the column information or row information, and the column is formed as the $B \times p \times d$ dimension, the row is composed of $p \times M \times d$ dimension. Here, B is the batch size, M is $k^{(num)} + k^{(cat)}$ the number of features, d is the dimension of the feature embedding, and p is the dimension of the latent factor. Column transformer plays a role in identifying which features play an important role by utilizing the attention technique between features. As shown in (b) of Figure 2, the self-attention used in the previous study [12] is identical to the self-attention of the column transformer in our proposed algorithm. Row transformer's self-attention utilizes attention between each instance within batch data. The row transformer's self-attention allows it to learn by borrowing information from other instances in the batch if a specific instance is noisy or blur. More detailed content is covered in the analysis section. The input of column transformer is the output of the column factorizer, which is $B \times (M + p) \times d$. The input of the row transformer is the output of row factorizer, that is $(B + p) \times M \times d$.

C. CLS TOKENIZER AND CLS QUERY-WISE TRANSFORMER

CLS Tokenizer works the same as column latent factorizer. However, a CLS token is inserted, not a latent factorizer, and



(a) Process for Generating Latent Factors



(b) Column and Row Transformer

FIGURE 2. Column and Row Transformer.
Note: As shown in (a), it consists of a total of *L* number of layers for transformer, and only query factorizer enters the query in the last L_{th} layer. For the column and row transformer, the PreNorm transformer [12], [20] is used as in (b).

the dimension of the token is always one-dimensional. CLS query-wise transformer works the same as L_{th} layer of column transformer. However, the CLS Token for prediction, not the latent factorizer, is inserted into the latent embedding matrix. Therefore, the transformer's self-attention is applied to the CLS query token, storing information for the final prediction.

V. EXPERIMENTS

A. DATASETS AND COMPUTING RESOURCES

We used a diverse set of eleven public datasets which is the same dataset of the previous study Gorishniy et al. [12]. However, considering the training time, we reduced the size of large data through random sampling. All datasets are consisting of train set, validation set, test set and all algorithms use the same datasets. Table 1 shows the source of datasets and Table 2 describes the details of the datasets. In order to conduct the experiment, we employed particular hardware specifications. These specifications consisted of an NVIDIA Tesla V100 16GB GPU, an Intel Xeon 2.30GHz CPU, and 64GB of RAM. The software utilized for the experiment was Python version 3.8.8, along with PyTorch version 1.7.1 and CUDA version 11.1.

B. DATA PREPROCESSING

We preprocessed the same way of the previous study [12]. Basically, the quantile transformation was applied, and standard normalization was applied to the Helena dataset and the ALOI dataset. For Epsilon dataset, As discussed in the prior study [12], we concluded that preprocessing adversely affects performance, so raw features were utilized without preprocessing.

TABLE 1. Sources of datasets.

CA	California Housing (real estate data) [21]
AD	Adult (income estimation) [22]
HE	Helena (anonymized dataset) [23]
JA	Jannis (anonymized dataset) [23]
HI	Higgs (simulated physical particles) [24]
AL	ALOI (image dataset) [25]
EP	Epsilon (simulated physics experiments) [26]
YE	Year (audio features) [27]
CO	Covertype (forest characteristics) [28]
YA	Yahoo (search queries) [29]
MI	Microsoft (search queries) [30]

C. HYPERPARAMETER TUNING

We utilized Bayesian optimization for hyperparameter tuning. In Table 3, it can be observed that an early stopping was applied with a patience of 16, and the total number of epochs was not separately indicated. The Hidden factor is used to calculate the output dimension of a fully connected layer inside a transformer. For example, when the input dimension of the fully connected layer is n, the output dimension would be int (n × Hidden factor).

TABLE 2. Properties of datasets.

	CA	AD	HE	JA	HI	AL	EP	YE	CO	YA	MI
Train	13000	26000	40000	55000	64000	12000	64000	100000	70000	23000	120000
Validation	3000	6500	10000	13000	15000	3500	16000	25000	23000	5000	40000
Test	4000	16000	15000	15000	19000	4000	20000	25000	23000	7000	40000
Numerical	8	6	27	54	288	128	2000	90	54	699	136
Categorical	0	6	0	0	0	0	0	0	0	0	0
Metric	RMSE	ACC	ACC	ACC	ACC	ACC	ACC	RMSE	ACC	RMSE	RMSE
Class	-	2	100	4	2	1000	2	-	7	-	-

TABLE 3. Hyperparameter of LF-transformer.

	CA	AD	HE	JA	HI	AL	EP	YE	CO	YA	MI
Activation	reglu	reglu	reglu	reglu	reglu	reglu	reglu	reglu	reglu	reglu	reglu
Attention dropout	0.451886	0.290285	0.021247	0.419491	0.402993	0.04885	0.15	0.395004	0.223646	0.10893	0.236703
Hidden factor	2.342426	2.037608	2.007966	1.061031	1.695314	1.333333	1.333333	1.333333	1.333333	1.333333	1.333333
Embedding dim	272	352	512	288	216	472	96	384	408	72	104
Hidden layer dropout	0.146239	0.160582	0.182515	0.463288	0.079468	0.218969	0	0.294022	0.19588	0.257506	0.087825
Initialization	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming	kaiming
Num of heads	8	8	8	8	8	8	8	8	8	8	8
Num of repetitions in a transformer.	3	3	2	4	3	3	1	1	2	2	2
Prenormalization	FALSE	TRUE	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Residual dropout	0	0	0	0	0.025131	0	0.15	0	0	0	0
Latent factor dim	1	1	1	2	1	2	1	1	1	2	1
Batch size	250	500	500	500	500	500	500	1000	500	125	1000
Learning rate	9.23E-05	2.67E-05	1.42E-05	0.000109	0.000342	4.45E-05	0.0001	6.89E-05	3E-05	0.000188	4.69E-05
Optimizer	admaw	admaw	admaw	admaw	admaw	admaw	admaw	admaw	admaw	admaw	admaw
Patience	16	16	16	16	16	16	16	16	16	16	16
Weight decay	2.24E-06	1.88E-05	5.16E-05	2.9E-05	2.74E-05	2.38E-05	0.000025	5.05E-05	1.04E-05	9.15E-05	0.001

TABLE 4. Results of singular model.

	CA	AD	HE	JA	HI	AL	EP	YE	CO	YA	MI	AVG RANKING
TabNet	0.4989	0.8514	0.3727	0.7244	0.7208	0.8030	0.8856	9.1803	0.9076	0.8162	0.7478	8.64
SNN	0.5032	0.8557	0.3720	0.7192	0.7199	0.8389	0.8932	9.2048	0.9099	0.8022	0.7512	8.36
MLP	0.5043	0.8566	0.3811	0.7206	0.7248	0.8378	0.8874	9.0809	0.9102	0.7861	0.7468	5.91
ResNet	0.4855	0.8551	**0.3928**	0.7281	0.7272	0.8678	0.8960	9.0674	0.9182	0.7910	**0.7401**	3.73
AutoInt	0.4801	0.8571	0.3668	0.7255	0.7224	0.8255	0.8882	9.1072	0.8819	0.8005	0.7500	6.82
DCN-V2	0.4844	0.8563	0.3826	0.7157	0.7212	0.8445	0.8921	9.1433	0.9128	0.7906	0.7489	6.18
GrowNet	0.4871	0.8560	-	-	0.7238	-	0.8739	9.1289	-	0.7931	0.7507	7.56
NODE	0.4651	0.8570	0.3483	0.7259	0.7248	0.8113	0.8959	**8.9976**	0.9130	**0.7812**	0.7452	4.36
SAINT	0.4880	0.8524	0.3811	0.7151	0.7105	0.7760	-	9.6830	0.92417	0.79842	0.7870	8.80
FT-Transformer	0.4599	0.8575	0.3883	**0.7333**	**0.7293**	**0.8732**	0.8953	9.0562	0.9320	0.7882	0.7425	2.18
LF-Transformer	**0.4588**	**0.8581**	0.3865	0.7323	0.7257	0.8690	**0.8962**	9.0423	**0.9325**	0.7882	0.7423	**2.00**

D. RESULTS

As can be seen in Table 4, FT-Transformer and NODE performed well on several datasets. However, our proposed model shows the best performance in most of the other datasets. As discussed in the case of NODE in the previous study [12], a direct comparison is difficult because it has a complex structure in the form of an ensemble rather than a single model. Table 5 shows the ensemble results of resnet, FT-Transformer, and our proposed algorithm. The ensemble algorithms utilized the average results from each of the 15 individually trained models. As shown in the table, the LF-Transformer demonstrates superior performance compared to both individual models and a single model. It was confirmed that the ensemble of NODE showed lower performance than the FT-Transformer in the prior study [12], so it was not tested separately. Table 6 shows the performance comparison of the LF-Transformer ensemble model with GBDT algorithms.

VI. ANALYSIS

The mnist dataset was used to analyze the effects of column transformer, row transformer, and latent factorizer. The mnist dataset is a handwriting image dataset, and although it is far from the tabular dataset, it was chosen because it is very popular and easy to visualize the results.

A. FEATURE ATTENTION MAP

We confirmed the attention map through the method proposed by [31] and [32]. In order to see the attention map, the attention

TABLE 5. Results of ensemble model.

	CA	AD	HE	JA	HI	AL	EP	YE	CO	YA	MI	AVG RANKING
ResNet	0.4773	0.8566	**0.3964**	0.7343	0.7319	0.8750	**0.8970**	9.0238	0.9223	0.7854	**0.7365**	2.45
FT-Transformer	0.4497	0.8587	0.3947	0.7392	0.7325	**0.8883**	0.8964	9.0048	0.9388	0.7834	0.7408	2.18
LF-Transformer	**0.4466**	**0.8592**	0.3960	**0.7396**	**0.7331**	0.8871	0.8968	**8.9976**	**0.9391**	**0.7832**	0.7405	**1.36**

TABLE 6. Comparison of GBDT and LF-transformer ensemble.

	CA	AD	HE	JA	HI	AL	EP	YE	CO	YA	MI
catboost	**0.3902**	0.8715	0.3843	0.7231	0.7272	0.8470	0.8769	9.1062	0.9224	0.7802	0.7389
XGBoost	0.4323	**0.8721**	0.3719	0.7231	0.7251	0.7948	0.8762	9.1529	0.9255	**0.7787**	**0.7380**
LF-Transformer	0.4466	0.8592	**0.3960**	**0.7396**	**0.7331**	**0.8871**	**0.8968**	**8.9976**	**0.9391**	0.7832	0.7405

TABLE 7. Effects of latent factor dimensions on real datasets.

Factor Dim	1	2	10	15	25	30	35	45
ALOI (ACC)	0.8678	0.8690	0.8665	0.8649	0.8650	0.8645	0.8649	0.8640
Yahoo (RMSE)	0.7889	0.7882	0.7903	0.7893	0.7902	0.7889	0.7890	0.7898

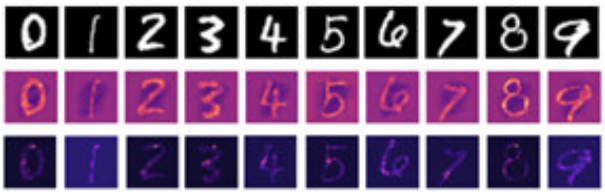
score, $softmax((QK^T)/\sqrt{d})$ value has to be checked. The attention score of the row transformer is a three dimensional (3D) tensor, which is $784 \times (B + p) \times (B + p)$. After selecting factor p from the second component of tensor, it was visualized by removing factor p from the $784 \times (B + p)$. The attention score of the column transformer is a 3D tensor, $B \times (784 + p) \times (784 + p)$. To secure the attention map, if factor p is chosen from $(784 + p)$, the second component of tensor, it becomes a two dimensional (2D) matrix of $B \times (784 + p)$, and we exclude the factor p with the selection ofvf $B \times 784$ for visualization. Looking at the attention map of the Row Transformer in (a) of Figure 3, the features with numbers show a strong attention, and the area around the number features shows very low attention. Parts that are far from the number features appear relatively bright and show some degree of attention, but they are relatively very low compared to the attention applied to the number features.

B. BATCH ATTENTION MAP

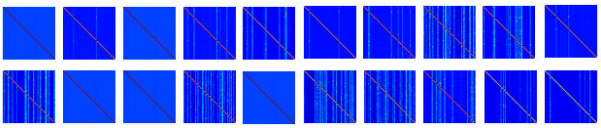
To check the row transformer's batch attention map, 20 features were randomly sampled in the dimension of mnist data, $28 \times 28 = 748$ in (b) of Figure 3. Although it is not known exactly what part each feature is responsible for in the image, features 15, 75, 362, 479, and 533 are identified as features that do not have numerical features, so it can be seen that attention is not captured at all. On the other hand, it can be seen that features such as 626 and 544 are helping to borrow information by paying attention to other samples on the batch.

C. EFFECT OF LATENT FACTOR DIMENSIONALITY

In Figure 4, when the latent factor is one-dimensional, it is confirmed that each number is mixed. However, as the number



(a) Column and Row Transformer Feature Attention Map

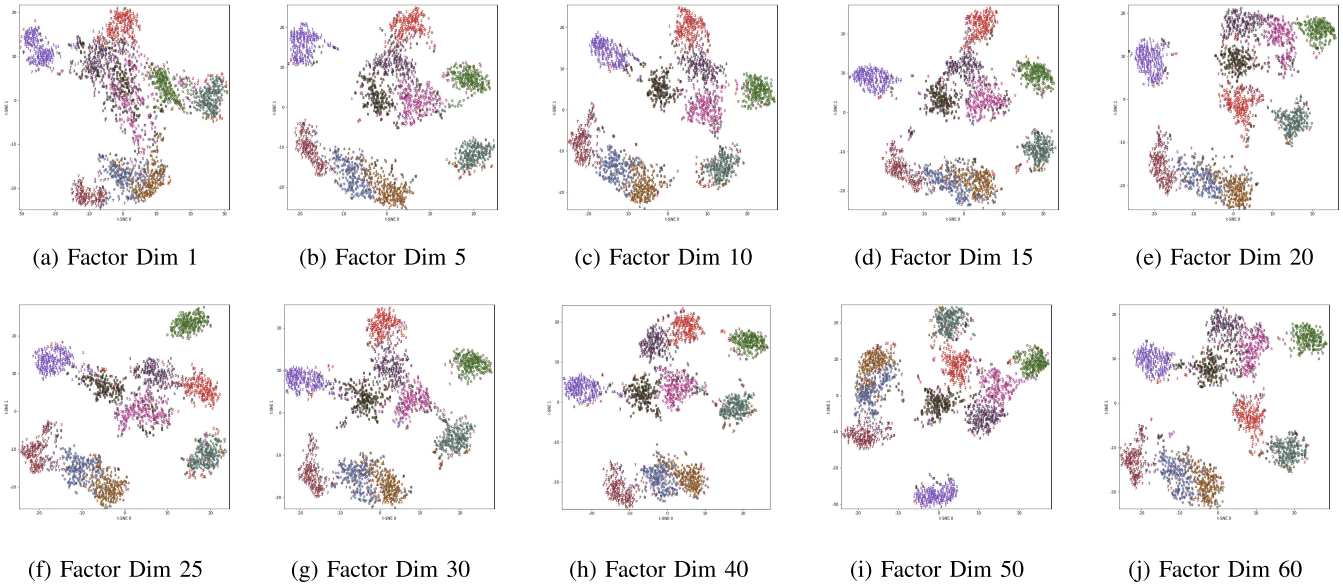


(b) Row Transformer Batch Attention Map

FIGURE 3. Transformer Attention Map.
Note: In (a), the row and column transformer feature attention map is shown. The first line is the original input, the second is the row attention feature map, and the last is the column attention feature map. In (b), the row transformer batch attention map from the top left to the right, 15_{th}, 70_{th}, 75_{th}, 124_{th}, 151_{th}, 174_{th}, 203_{th}, 209_{th}, 329_{th}, 334_{th}, 348_{th}, 263_{th}, 479_{th}, 487_{th}, 544_{th}, 544_{th}, 626_{th}, 680_{th}, 710_{th}, and 713_{th} features are shown.

of dimension of the latent factor increases, it is noticeable that the class of each number is clearly distinguished. It is obvious that the latent factor is a hyper parameter, and the optimal value must be obtained through hyper parameter optimization.

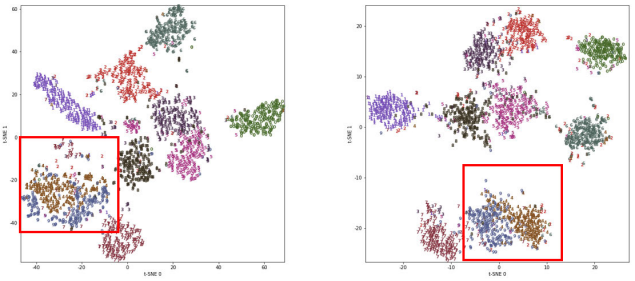
Table 7 confirms that the performance varies according to the latent factor dimension for the ALOI dataset and the yahoo dataset. The ALOI dataset's accuracy is better when the score is closer to 1 and the Yahoo dataset's root mean square error (RMSE) is better when the score is closer to 0. Both datasets



(a) Factor Dim 1 (b) Factor Dim 5 (c) Factor Dim 10 (d) Factor Dim 15 (e) Factor Dim 20

(f) Factor Dim 25 (g) Factor Dim 30 (h) Factor Dim 40 (i) Factor Dim 50 (j) Factor Dim 60

FIGURE 4. T-sne Results of Latent Matrix.
Note: The t-sne result of latent embedding matrix according to the dimension of latent factor for mnist data. Latent factor in dimensions 1, 5, 10, 15, 20, 25, 30, 40, 50, and 60 from the top left to the right. It can be seen that the feature representation improves as the latent dimension increases.



(a) Column latent factorizer only (b) Our model

FIGURE 5. Comparison of T-sne Results.
Note: The t-sne result of embedding matrix with 40-dimnnsional latent factor. a) applies only the column latent factorizer. (b) is our proposed model, and both Column and Row latent factorizers have been applied. It can be seen that the final embedding matrix performs better in feature representation.

show better performance when the dimension of latent factors is 2.

D. DIFFERENCE BETWEEN ORIGINAL AND LATENT EMBEDDING

In (a) of Figure 5, the t-sne result of the embedding matrix applied with the column latent factorizer looks okay in terms of separation. The distinction between 4 (brown) and 9 (blue) in the bottom left is not clear. However, as shown in (b) of Figure 5, the t-sne result of the embedding matrix to which the Latent Factor 40 dimension is applied shows a relatively clear distinction between 4 and 9.

Through this, the effect can be confirmed like matrix factorization, which restores the original matrix by embedding features in the latent space. The performance of clustering is evaluated based on two important criteria: how tightly the points are packed within each cluster, and how far apart the clusters are from each other. It can be observed that our model outperforms in both of these metrics.

VII. CONCLUSION

A. LIMITATIONS AND FUTURE WORK

In addition to the common limitations of tabular data-based deep learning ensemble algorithms, this study also intends to suggest additional research directions by identifying the following limitations.

First, our proposed model, LF-Transformer, does not fully provide explainable insights. As discussed in the analysis (section VI), it is possible to identify which features and data influenced the results through the attention map, but it is difficult to see that it provides all possible explanations. Therefore, additional research on the application of additional explainable techniques is needed. Second, although the superiority of LF-Transformer was demonstrated through various experiments in terms of performance, it was confirmed that our proposed model may not be optimal in several experiments. Besides the tabular data used in the experiment, LF-Transformer may not be a globally optical solution, so adjustments such as additional parameter tuning are required to properly utilize it. In addition, like other most of the tabular data-based algorithm improvement studies, continuous efforts are required to develop a globally optimal solution through unending research and experiments. Third, the LF-Transformer requires a significant amount of memory, because it creates separate data tensors for columns and rows in the model. This means that the model's complexity is high and may not be optimal in terms of time and cost

efficiency. Therefore, future research should consider metrics that factor in time and cost efficiency. Last but not least, since LF-Transformer applies a row-wise transformer, the batch size must always be the same. This means that the entire dataset must be divided by the batch size, and the dataset that is not divisible by the batch size must be partially deleted or enlarged to match the size. For example, if the number of rows in an entire dataset is 4,120 and a batch size is 200, only 120 rows are entered in the last batch of the last iteration, so in this case, the total number of rows should be reduced by 4,000 or increased up to 4,200. Future research should aim to solve this problem and create an environment in which the model can run well.

B. CONCLUDING REMARK

In this study, as we were Influenced and inspired by previous studies on FT-Transformer and matrix factorization, we proposed LF-Transformer, of creating the final latent embedding matrix by generating an attention latent factor matrix with a transformer. It was confirmed that the data representation of our proposed method is rather superior. Further, it was found that when this latent embedding matrix is used for classification or regression problems based on tabular datasets, it shows relatively superior performance compared to other current state-algorithms. We hope that our study results and our model would serve as basis and go-to option for further developments on tabular data-based deep learning.

REFERENCES

[1] T. B. Brown, "Language models are few-shot learners," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 1877–1901.

[2] Y. Zhang, J. Qin, D. S. Park, W. Han, C.-C. Chiu, R. Pang, Q. V. Le, and Y. Wu, "Pushing the limits of semi-supervised learning for automatic speech recognition," 2020, *arXiv:2010.10504*.

[3] B. Choi, Y. Lee, Y. Kyung, and E. Kim, "ALBERT with knowledge graph encoder utilizing semantic similarity for commonsense question answering," *Intell. Autom. Soft Comput.*, vol. 36, no. 1, pp. 71–82, 2023.

[4] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention mask transformer for universal image segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 1280–1289.

[5] E. Kim, Y. Lee, J. Choi, B. Yoo, K. J. Chae, and C. H. Lee, "Machine learning-based prediction of relative regional air volume change from healthy human lung CTS," *KSII Trans. Internet Inf. Syst.*, vol. 17, no. 2, pp. 576–590, 2023.

[6] S. Arik and T. Pfister, "TabNet: Attentive interpretable tabular learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 8, May 2021, pp. 6679–6687.

[7] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[8] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[9] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Inf. Fusion*, vol. 81, pp. 84–90, May 2022.

[10] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: Unbiased boosting with categorical features," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, 2018.

[11] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[12] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds. Red Hook, NY, USA: Curran Associates, 2021, pp. 18932–18943.

[13] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.

[14] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Mach. Learn.*, vol. 40, no. 2, pp. 139–157, Aug. 2000.

[15] M. Zięba, S. K. Tomczak, and J. M. Tomczak, "Ensemble boosted trees with synthetic features generation in application to bankruptcy prediction," *Expert Syst. Appl.*, vol. 58, pp. 93–101, Oct. 2016.

[16] S. Popov, S. Morozov, and A. Babenko, "Neural oblivious decision ensembles for deep learning on tabular data," 2019, *arXiv:1909.06312*.

[17] L. Katzir, G. Elidan, and R. El-Yaniv, "Net-DNF: Effective deep modeling of tabular data," in *Proc. Int. Conf. Learn. Represent.*, 2020.

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

[19] G. Somepalli, M. Goldblum, A. Schwarzschild, C. B. Bruss, and T. Goldstein, "SAINT: Improved neural networks for tabular data via row attention and contrastive pre-training," 2021, *arXiv:2106.01342*.

[20] Q. Wang, B. Li, T. Xiao, J. Zhu, C. Li, D. F. Wong, and L. S. Chao, "Learning deep transformer models for machine translation," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1810–1822.

[21] R. Kelley Pace and R. Barry, "Sparse spatial autoregressions," *Statist. Probab. Lett.*, vol. 33, no. 3, pp. 291–297, May 1997.

[22] R. Kohavi, "Scaling up the accuracy of Naive-Bayes classifiers: A decision-tree hybrid," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, vol. 96, 1996, pp. 202–207.

[23] I. Guyon, L. Sun-Hosoya, M. Boullé, H. J. Escalante, S. Escalera, Z. Liu, D. Jajetic, B. Ray, M. Saeed, and M. Sebag, "Analysis of the AutoML challenge series," *Automated Mach. Learn.*, pp. 177–219, May 2019.

[24] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature Commun.*, vol. 5, no. 1, pp. 1–9, Jul. 2014.

[25] J.-M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders, "The Amsterdam library of object images," *Int. J. Comput. Vis.*, vol. 61, no. 1, pp. 103–112, Jan. 2005.

[26] G.-X. Yuan, C.-H. Ho, and C.-J. Lin, "An improved GLMNET for l1-regularized logistic regression," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2011, pp. 33–41.

[27] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proc. 12th Int. Soc. Music Inf. Retr. Conf. (ISMIR)*, 2011.

[28] J. A. Blackard and D. J. Dean, "Comparative accuracies of artificial neural networks and discriminant analysis in predicting forest cover types from cartographic variables," *Comput. Electron. Agricult.*, vol. 24, no. 3, pp. 131–151, Dec. 1999.

[29] O. Chapelle and Y. Chang, "Yahoo! learning to rank challenge overview," in *Proc. Learn. Rank Challenge*, 2011, pp. 1–24.

[30] T. Qin and T.-Y. Liu, "Introducing LETOR 4.0 datasets," 2013, *arXiv:1306.2597*.

[31] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," 2020, *arXiv:2005.00928*.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2020, *arXiv:2010.11929*.

[33] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo, "OpenML: Networked science in machine learning," *ACM SIGKDD Explor. Newslett.*, vol. 15, no. 2, pp. 49–60, Jun. 2014.

[34] J. Ho, N. Kalchbrenner, D. Weissenborn, and T. Salimans, "Axial attention in multidimensional transformers," 2019, *arXiv:1912.12180*.

[35] R. M. Rao, J. Liu, R. Verkuil, J. Meier, J. Canny, P. Abbeel, T. Sercu, and A. Rives, "MSA transformer," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 8844–8856.
[36] H. Iida, D. Thai, V. Manjunatha, and M. Iyyer, "TABBIE: Pretrained representations of tabular data," 2021, *arXiv:2105.02584*.

KWANGTEK NA received the B.Sc. degree in civil engineering and the M.S. degree in computer science and engineering from Inha University, South Korea, in 2013 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering. He is also researching machine learning with Hanwha Group. His research interests include statistical machine learning, reinforcement learning, and recommender systems.

JU-HONG LEE received the B.S. and M.S. degrees in computer engineering from Seoul National University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree in computer science from KAIST, Daejeon, South Korea, in 2001. He is currently a Professor of computer science and engineering with Inha University, Incheon, South Korea. He is also the CEO of Qhedge Company Ltd. He has developed the credit scoring system and hedge investment algorithms for portfolio management, index tracking, and arbitrage trading. His research interests include machine learning, data mining, financial engineering.

EUNCHAN KIM received the B.A. degree in economics from the University of Minnesota, Twin Cities, in 2012, and the M.S. degree in management (concentration: information systems) and the Ph.D. degree in engineering (concentration: intelligence and information systems) from Seoul National University, in 2017 and 2023, respectively. He is currently a Senior Researcher with Hanwha Group and a Lecturer with the College of Business Administration, Seoul National University. He is also a Visiting Scholar with both Seoul National University Hospital and Jeonbuk National University Hospital. His research interests include information systems, artificial intelligence (AI), applications of AI, big data, and engineering management.