CATN: Cross Attentive Tree-Aware Network for Multivariate Time Series Forecasting

Hui He[1], **Qi Zhang**[2,3], **Simeng Bai**[1], **Kun Yi**[1], **Zhendong Niu**[1,4*]

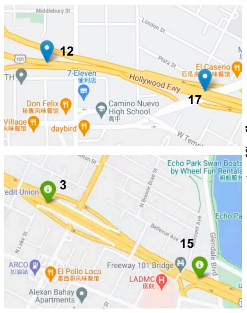
[1]Beijing Institute of Technology
[2]University of Technology Sydney
[3]DeepBlue Academy of Sciences
[4]University of Pittsburgh
{hehui617, zhangqi_cs, baisimeng, yikun, zniu}@bit.edu.cn

Abstract

Modeling complex hierarchical and grouped feature interaction in the multivariate time series data is indispensable to comprehending the data dynamics and predicting the future condition. The implicit feature interaction and high-dimensional data make multivariate forecasting very challenging. Many existing works did not put more emphasis on exploring explicit correlation among multiple time-series data, and complicated models are designed to capture long- and short-range patterns with the aid of attention mechanisms. In this work, we think that a pre-defined graph or a general learning method is difficult due to its irregular structure. Hence, we present CATN, an end-to-end model of Cross Attentive Tree-aware Network to jointly capture the inter-series correlation and intra-series temporal patterns. We first construct a tree structure to learn hierarchical and grouped correlation and design an embedding approach that can pass a dynamic message to generalize implicit but interpretable cross features among multiple time series. Next in the temporal aspect, we propose a multi-level dependency learning mechanism including global&local learning and cross attention mechanism, which can combine long-range dependencies, short-range dependencies as well as cross dependencies at different time steps. The extensive experiments on different datasets from real-world show the effectiveness and robustness of the method we proposed when compared with existing state-of-the-art methods.

Introduction

Time series forecasting is an essential component in countless domains including medical monitoring, e-learning, energy and smart grid management, economics and finance, sensor network analysis and epidemic propagation (Faloutsos et al. 2019; Wan and Niu 2020). Of all the time-series data mining tasks, it is one of the most tremendous applications of data-driven methods and arguably the most challenging one. In the above scenarios, the prevailing approaches into the application today have been developed in the background of extremely substantial time-series data on historical behavior to forecast with complicated models structured and adjusted by domain experts. However, with the increasing dimensions of online, time-stamped activities strain the prediction capacity to measure collective behavior of important evolutions (Boroujeni et al. 2018; Salinas et al. 2019; Gao, Duru, and Yuen 2021).



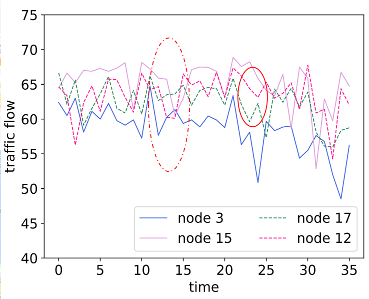


Figure 1: Intuitive example: in left part, we select four sensors with their index 3, 15 (green marker), 17 and 12 (blue marker) from real-world dataset on Google Map. The right half shows the traffic flow of these four sensors in the raw data from 1:15 A.M. to 10:15 A.M..

Existing forecasting models are designed focusing on exploring intra-series temporal patterns, some short sequences of consecutive characters with blocks occurring in the same time interval or varying time delays, in hundreds or thousands of related time series (Zhuang, Li, and Wong 2014; Huang et al. 2019; Wu et al. 2020; Sen, Yu, and Dhillon 2019; Yang et al. 2020; Cao et al. 2020). However, many high-dimension and real-world collections of time series exhibit complex hierarchical and grouped correlation (Mancuso, Piccialli, and Sudoso 2020; Rangapuram et al. 2021). As an intuitive example, the occupancy rates of national roads can be disaggregated into different provincial roads, and the occupancy rates of provincial roads can be separated into county roads. As shown in Figure 1, sensor 12 and sensor 17 are in similar branches and their traffic flow trends are also similar (solid ellipse). Sensor 15 is located in the main road and sensor 3 is located in the branch of sensor 15, so the traffic flow of sensor 15 is mostly higher than that of sensor 3 (dotted ellipse). Hence, implicit or rare feature interaction of inter-series data needs to be modeled explicitly.

There are some prior works on improving the latent spatial correlation between variables. Graph neural networks (GNNs) have obtained great achievements in multivariate time series forecasting viewed from a graph perspective to capture inter-series correlation. However, the limit of these

[*]Corresponding author

graph models strongly depends on the predefined graph structure to extract spatio-temporal features (Yu, Yin, and Zhu 2018). Instead of a predefined graph structure, graph learning module is designed to extract the uni-directed relations among variables, where the nodes in a graph are out-of-order, which makes it difficult to learn the topological relationship in the graph (Wu et al. 2020; Bai et al. 2020).

Hence, the major challenge for exploring the hierarchical and grouped correlation between multiple variables is enhancing the multi-step prediction capacity to meet the increasing dimension demand, which requires (a) extraordinary modeling ability to extract dynamic correlation of inter-series data and (b) efficient balance ability to capture global and local dependency of intra-series data. Motivated by (Lyu et al. 2020; Wang et al. 2018b; Cheng et al. 2018), we employ an ordered hierarchical tree structure where both structure and content information can be preserved to discover hierarchical feature interaction between inter-series data at different time steps to address the issue (a). We dynamically update node and edge representation after each prediction to enhance node/edge characteristics. Meanwhile, we propose a novel multiple-level dependency learning mechanism including global&local learning and cross attention mechanism to enhance the intra-series feature discriminability for requirement (b). Our contributions are summarized as:

• We propose CATN that can capture the intra-series temporal dependency and dynamic inter-series correlation simultaneously. To our knowledge, this is the first work to introduce a tree-based neural network into multivariate time-series forecasting.

• We introduce a tree structure to dynamically reason the hierarchical and grouped interaction and design an embedding method to generate implicit but interpretable inter-series features from the structure's perspective.

• We design a multiple-level dependency learning mechanism including global&local learning approach that considers both long- and short-term dependency and cross attention mechanism that integrates intra-series dependency at crossed time steps, which enhance the feature discriminability largely of different nodes at the same level.

• We carry out extensive experiments on our end-to-end model and obtain state-of-the-art performance on various time series forecasting baselines, which proves its effectiveness and robustness.

Related Work

Correlated Time Series Forecasting

Traditional methods cannot model complex patterns or dependencies lying in real-world data, although they are simple and interpretable. In recent years, a lot of works in time series forecasting based on deep learning methods have gained popularity. A majority of such studies (Sen, Yu, and Dhillon 2019; Du et al. 2021; Zhuang et al. 2020; Li et al. 2018) rely on recurrent neural network (RNN). Moreover, temporal convolutions have been used for time-series forecasting to extract local features. A combination of 2D convolution to capture short-term patterns and recurrent structures to capture long-term patterns between multiple time-series have been leveraged (Lai et al. 2018). However, scaling this model beyond a few thousand time series is difficult because of the growing size of the input layer. Shih et al. designed an LSTM-based model where hidden states were fed into a set of CNN filters to model frequency information and then calculated weights by a scoring function to select related time series, and finally create the forecasting values (Shih, Sun, and Lee 2019). Xu et al. utilized a tensorized LSTM with adaptive shared memory to learn global features and a multitask 1dCNN to learn local features (Xu et al. 2020).

However, the combination of capturing local dependencies by CNNs and maintaining long-term dependencies by RNNs based on intra-series data ignores the inter-series dependencies among multiple variables. The Conv-LSTM structure can not meet the requirement (a) (given in Introduction). Bai et al. studied node-specific patterns instead of a pre-defined graph and then used a learnable node embedding dictionaries to infer spatial dependencies between different series (Bai et al. 2020). However, learning general graphs is difficult due to their erratic structures (Lyu et al. 2020). The Tree is an ordered graph with a distinct hierarchical structure. Tree structures have been used in image classification (Lyu et al. 2020), text classification (Cheng et al. 2018), explainable recommendation (Wang et al. 2018b) et al.

Cross Attention Mechanism

Cross attention mechanism has been applied for few-shot classification (Hou et al. 2019), object detection (Lin et al. 2021) and video-text matching (Lee et al. 2021) or text-text matching (Hao et al. 2017), and recommendation (Hu et al. 2018; Wang et al. 2018a; Zhu et al. 2021). Lin et al. modeled the bidirectional correspondences between target and query image based on transformer. This architecture exploited the similarity between two streams (query-to-target and target-to-query) (Hou et al. 2019). Lin et al. proposed a cross attention module including two sub-modules: language guide vision and vision guide language for referring expression comprehension (Lin et al. 2021). Hao et al. extracted i vital entities from answers and constructed four bidirectional streams (answer-entity$i$-to-question attention and question-to-answer-entity$i$ attention) (Hao et al. 2017). Lee et al. devised a cross-correlation to generate both of audio and visual features attended by each other (Lee et al. 2021). The most similar to our cross attention mechanism is the cross attention module of (Hou et al. 2019), which computes cross attention between feature maps of two images (support and query). However, our cross attention mechanism is proposed for multivariate time series forecasting. Owing to the easy-to-interpret cross attentive tree-aware network, the overall prediction process is fully transparent and self-explainable.

Preliminaries

A time series $\mathrm{x}^i = \{x_1^i, x_2^i, ..., x_T^i\}$ records the observed values of variate i at time stamp T. A multivariate time series is denoted as $\mathcal{X} = \{\mathrm{x}^1, \mathrm{x}^2, ..., \mathrm{x}^{d_x} \mid \mathrm{x}_T \in \mathbb{R}^{d_x}\}$. d_x describes the variate dimension and the time interval between two consecutive observation values is fixed.

Problem Statement Given the fixed input time window T_x, we have the input $\mathcal{X} = \{\mathrm{x}^1, \mathrm{x}^2, ..., \mathrm{x}^{d_x} \mid \mathrm{x}^{d_x} \in$

$\mathbb{R}^{T_x}\}$, where d_x represents the number of univariate time series. The output is the corresponding sequence $\mathcal{Y} = \{\mathrm{y}^1, \mathrm{y}^2, ..., \mathrm{y}^{d_y} \mid \mathrm{y}^{d_y} \in \mathbb{R}^{T_y}\}$. For implementing the rolling forecasting, we aim at predicting $\mathcal{Y}_{T+h}$ based on $\mathcal{X}$. $h \in \mathbb{R}^+$ is the envisaged time horizon later than the current time stamp. T_y is the output window whose length may differ from T_x. We formulate that for the forecasting target $\mathcal{Y}_{T+h} \in \mathbb{R}^{d_y}$, the input matrix is $\mathcal{X} = \{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_{T_x} \mid \mathrm{x}_{T_x} \in \mathbb{R}^{d_x}\}$.

Hierarchical Clustering (HC) Given a set of n univariate time series $\mathcal{V} = \{v_1, v_2, ..., v_n \,|v_i \in \mathbb{R}\} = \mathcal{X}^T$. The goal of HC is to find a tree $\mathcal{T}$ that contains them as leaves. The non-leaf nodes of $\mathcal{T}$ correspond to the clusters of variables at different levels. Therefore, we call the non-leaf nodes of $\mathcal{T}$ as clusters, while also refering to them as sets of all variables in their subtrees. A matrix $S \in \mathbb{R}^{n\times n}$ includes the pairwise similarities between the variables. The nodes and the pairwise similarities are represented by a tree $\mathcal{T} = (\mathcal{V}, S)$.

Methodology

The architecture of CATN is shown in Figure 2 and more details are described in the following sections.

Tree Construction

We believe that there is a potential hierarchical and grouped correlation between multiple variables. Agglomerative hierarchical clustering performs clustering in a greedy manner, that is, initially, there are n clusters, each consisting of a single feature point, each step needs to merge two clusters that are the most similar. The distance measure affects the final clustering results, and high dimensional variables require relatively large time consumption. There is a trade-off between distance measure accuracy and time complexity.

The representations of different clusters depend on the results of similarity measure or the distance function. Inspired by (Maxim et al. 2020), we use the weights of data in different dimensions $w_i = z_i/(\frac{1}{n}\sum_{i=1}^{n} z_i)$. The w-weighted euclidean distance is defined as distance measures for feature vectors: $d(\mathrm{z}, \mathrm{p}) := \mathrm{w}\,\|\mathrm{z} - \mathrm{p}\|_2^2$. The points that have a minimal dissimilarity are merged into a single cluster according to the w-weighted euclidean distance criterion. For inter-cluster distance, single/complete-linkage clustering computes the distance between two clusters by computing the distance between their closest/farthest data points:

$$s/c - L(Z, P) = min/max\{d(z_i, p_j)\,|z_i \in Z, p_j \in P\}$$

where $Z = \{z_i\}_{i=1}^{|Z|}$ and $P = \{p_i\}_{i=1}^{|P|}$ are two clusters.

However, the closest or the farthest pairwise distance cannot sufficiently represent their clusters. All-pairs linkage clustering computes the distance between the means of clusters, but this method is sensitive to clustering of non-convex shapes. Average-linkage criterion considers the average of the inter-cluster dissimilarity $a - L_{\alpha,\beta} = \sum_{z_i\in Z}\sum_{p_j\in P}\frac{d(z_i,p_j)}{|Z||P|}$. Average-linkage has the highest time complexity, i.e., $O(n^3)$, among these algorithms.

Inspired by (Emmendorfer and de Paula Canuto 2021), we propose a median linkage criterion to compute the distance of two clusters Z and P by selected pairs of points as below:

$$m - L_{\alpha,\beta}(Z, P) = \frac{1}{|S_{\alpha(N),\beta(N)}|}\sum_{(z,p)\in S_{\alpha(N),\beta(N)}} d(z, p)$$

where $S_{\alpha(N),\beta(N)}$ is a subset of all possible pairs $(z, p), z \in Z, p \in P$. We define $S_{\alpha(N),\beta(N)}$ as:

$$\begin{aligned} S_{\alpha(N),\beta(N)} =& \{(z, p), z \in Z, p \in P | r_{(\alpha(N))} \leq \\ & d(z, p) \leq r_{(\beta(N))}\} \end{aligned} \tag{1}$$

Here, $m_{(1)} \leq m_{(2)} \leq ... \leq m_{(N)}$ is a rank-ordered set M of the distances $d(z, p)$ between all $N = |Z||P|$ possible pairs $(z, p), z \in Z, p \in P$. The element $m_{(1)}$ and $m_{(N)}$ correspond to the pair with the minimum distance and the maximum distance respectively. To obtain the median $\tilde{m}$ of M, the parameters $\alpha(N)$ and $\beta(N)$ are set as 1 and $\frac{N}{2}$.

We construct a tree $\mathcal{T}$ which contains variables as leaves to build hierarchical relationships between variables. Measuring the quality of a tree $\mathcal{T}$ that represents the hierarchical correlation between features is achieved by evaluating the performance of label prediction tasks. The representation of $\mathcal{T}$ will be introduced in the next subsection.

Tree Embedding

We believe that there is a hierarchical and grouped correlation between multiple variables - as we shown in the introduction. The strength of tree-based methods lies in their effective structure and interpretability. Assume that the embeddings of the univariable i are the leaf nodes at the very bottom layer in $\mathcal{T}$. A subset of multiple variates that have potentially relevant is grouped together via one parent node, where the parent node is the embedding of the corresponding group. For the data matrix $\mathrm{X} \in \mathbb{R}^{T\times n}$, we have obtained a projection function $f : \mathrm{X} \to \mathcal{T}\{V, E\}$, where V represents the set of nodes, E represents the set of edges. The function is aimed to transfer matrix to tree space (Lyu et al. 2020). The node set V can be partitioned into two subsets, the non-leaf nodes V_I and leaf nodes V_L, $V = V_I \cup V_L$. The edge set E can be divided into two categories: $e_l \in E_L$ and $e_r \in E_R$. Tree $\mathcal{T}$ can be regarded as an effective tool to deliver high-order cross features from the original features.

Node Embedding To further encode high-order features, we map every leaf node $v_l \in V_L$ into a updatable dense embedding tensor $u \in \mathbb{R}^d$ where d is the dimensionality.

Time Embedding In order to capture the time dependence between sequences, hierarchical timestamps (day, week, month, year) and agnostic (A.M. and P.M., holidays) are needed to be considered. Time stamps can be used as part of a sequence to participate in prediction. We extract t hierarchical time stamps as $E_t \in \mathbb{R}^{r\times t}$ and then map to $\mathbb{R}^{r\times t\times d_t}$, where r represents the total of hierarchical time stamps and d_t represents the dimension of time embedding. After concatenating, the final node representation is $[u : E_t]$.

Edge Embedding For ease of notation, we use one-hot encoding to represent left edge and right edge initially. To facilitate embedding edge into the whole network for gradient backward, we finally project each edge into a learnable dense embedding vector e_l and e_r.

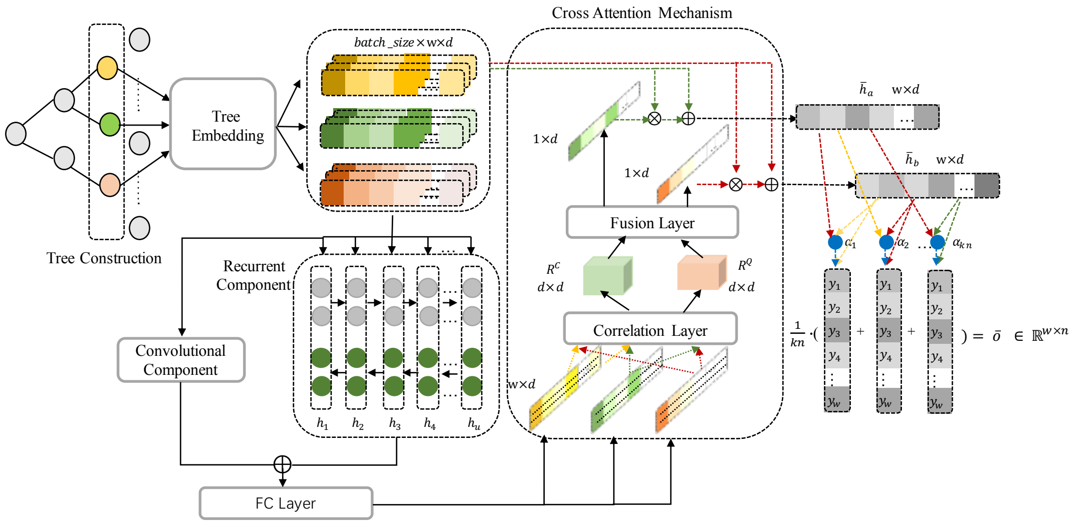


Figure 2: Architecture for the CATN designed for multivariate time-series forecasting. For the sake of simplicity, only a set of cross processes (red and green) are shown.

Non-leaf nodes $v_i \in V_I$ could be represented initially as :

$$v_i = \varphi(u_l) \cdot e_l + \varphi(u_r) \cdot e_r \tag{2}$$

where $\varphi(X)$ is an operation that fills in all missing values from X. The v_i depicts all latent representations of high-order features and will be used for prediction.

Updating Non-leaf Nodes Hierarchical correlation learning can explore dependency between parent nodes and children nodes. To make weight coefficients comparable, we perform a normalization on the left-child node and right-child node,

$$\Theta^{(lr)} = \frac{exp(v_i)}{\sum_{j=l,r} exp(e_j)} \tag{3}$$

Then we update the non-leaf nodes v_i with the $\theta^{(lr)}$, resulting in $v_{i_{up}} = \theta^{(lr)} \cdot v_i$. The output of each node in interaction layer we selected is a tensor $E_d \in \mathbb{R}^{w\times d}$ and will be delivered in the next subsection.

Global&Local Learning

We use a combination of convolutional and recurrent component to mining local and global information.

Convolutional Component At the kth level of the tree, we search up $E_d \in \mathbb{R}^{w\times d}$ of each node, which is randomly initialized and learned during training. Here, w means the window of time series. Local temporal convolution can capture local temporal patterns in the time dimension. Motivated by (Chang et al. 2018; Lai et al. 2018; Shih, Sun, and Lee 2019; Zhang et al. 2021; Shi et al. 2021), the first layer of global&local learning is a convolutional layer without pooling. We use n_c filters of width w_c and length l_c, where l_c is equal to d. All filters sweep through the output of tree module $E_d \in \mathbb{R}^{w\times d}$ and produce:

$$o_c = \frac{1}{n_c}\sum_{k=1}^{n_c} RELU(W_k * E_d + b_k) \tag{4}$$

where $*$ represents the convolution operation and the vector o_c is the output. After zero-padding, each vector o_c with length d is put on the left of E_d and then generates the output of this component with size $w \times 1$.

Recurrent Component Similarly, we get $E_d \in \mathbb{R}^{w\times d}$ and then the embeddings of kn nodes are fed into kn LSTMs. LSTM is a recurrent neural network and has been proven to be effective in many time series forecasting tasks such as (Du et al. 2020). For considering the information of before and after current moment, we employ bidirectional LSTM, which consists of both forward and backward networks. Thus we could obtain two hidden state sequences, one from the forward one $(\overrightarrow{h_i^1}, \overrightarrow{h_i^2}, ..., \overrightarrow{h_i^u})$ and the other from the backward one $(\overleftarrow{h_i^1}, \overleftarrow{h_i^2}, ..., \overleftarrow{h_i^u})$. After concatenating, we obtained $\{\overrightarrow{h_i}; \overleftarrow{h_i} : h_i = \sigma(o_i) \cdot tanh(c_i)\}$, where the hidden unit u of forward and backward LSTM is $\frac{d}{2}$. We denote the hidden state of multiple LSTMs as $\{h_1, h_2, ..., h_{kn}\}$, where kn denotes the number of nodes of the kth layer. To be further enhanced with the local information, the output is concatenated with o_c and then fed into cross attention module.

Cross Attention Mechanism

For calculating efficiently, we construct the contrast set $\mathcal{C} = \{h_a\}_{a=1}^{kn}$ and the query set $\mathcal{Q} = \{h_b\}_{b=a+1}^{kn}$ inspired by (Hou et al. 2019). We match arbitrary two non-leaf nodes at the kth level of the tree from $\mathcal{C}$ and $\mathcal{Q}$ as $\mathcal{S} = \{(h_a, h_b)| \, a \in [1, kn], b \in (a, kn]\}, crad(\mathcal{S}) = C_{kn}^2$, where $h_i \in \mathbb{R}^{w\times d}$ denotes the hidden state of the ith LSTM. The cross attention measures the extent of attention by the relatedness between h_a and h_b, help to re-read the vital information of the target and enhance the feature discriminability.

Inspired by (Hou et al. 2019; Hao et al. 2017), different from existing methods which extract the contrast and query features independently, we present a cross attention module that considers the dynamic relevance between $\mathcal{C}$ and $\mathcal{Q}$ in time pattern. We get the input $h_a \in \mathbb{R}^{w\times d}$ and $h_b \in \mathbb{R}^{w\times d}$ from $\mathcal{C}$ and $\mathcal{Q}$, respectively, where $h_a = [c_1, c_2, ..., c_d]$ and $h_b = [q_1, q_2, ..., q_d]$. Then we design a correlation layer to calculate a correlation map between c_i and q_j at staggered moments. The semantic relevance between c_i and q_j is computed by cosine distance in the correlation layer to get the correlation map $R \in \mathbb{R}^{d\times d}$ as:

$$R_{ij} = (\frac{c_i}{\|c_i\|_2})^T(\frac{q_j}{\|q_j\|_2}),\, i,j \in [1,d] \tag{5}$$

Moreover, two correlation maps based on R are defined: h_a correlation map $R^c = [r_1^c, r_2^c, ..., r_d^c]$ and h_b correlation map $R^q = [r_1^q, r_2^q, ..., r_d^q]$, where $r_i^c \in \mathbb{R}^d$ represents the correlation between local vector h_a at a certain time c_i and global vectors $\{q_i\}_{i=1}^d$, and $r_i^q \in \mathbb{R}^d$ represents the correlation between local vector of h_b at a certain time q_i and global vectors $\{c_i\}_{i=1}^d$. By this means, R^c and R^q capture all local correlations within a time window between input h_a and h_b.

To obtain the contrast and query attention maps, we use a fusion layer. We take the contrast attention map as an example. We refer to the contrast correlation map R^c as input of the fusion layer. $W \in \mathbb{R}^{d\times 1}$ is an intermediate matrix and b is the offset. For expressing each local correlation vector $\{r_i^c\}$ of R^c into an attention scalar, we use $f(\cdot)$ to normalize the attention scalar. The contrast attention at the ith position:

$$\alpha_i^c = \frac{exp(\omega_i^c)}{\sum_{j=1}^d exp(\omega_j^c)} \tag{6}$$

$$\omega_i^c = f(W^T r_i^c + b) \tag{7}$$

where $\alpha_i^c \in \mathbb{R}^{1\times d}$ and $\alpha^c \in \mathbb{R}^{w\times d}$. The function $f(\cdot)$ aggregates the correlations between the local class feature c_i and all local query features $\{q_j\}_{j=1}^d$ at the i-th position.

Furthermore, we get the query attention map $\alpha^q \in \mathbb{R}^{d\times d}$. we reshape α^c and α^q into $\mathbb{R}^{1\times d}$. Then we elementwisely weight the initial feature maps h_a and h_b into $1 + \alpha^c$ and $1 + \alpha^q$ via a residual attention mechanism. We finally form more discriminative feature map pairs $\bar{\mathcal{S}} = \{(\bar{h}_a, \bar{h}_b)| a \in [1, kn], b \in (a, kn]\}$. $\bar{o}$ calculates an average outputs based on attention again to mitigate the impact of mapping by:

$$\bar{o} = \frac{1}{crad(\mathcal{S})} \sum_{a=1}^{kn} \frac{exp(\omega_{h_a})}{\sum_{b=a}^{kn} exp(\omega_{h_b})} \bar{h}_a \tag{8}$$

here, $crad(\mathcal{S})$ denotes the size of $\mathcal{S}$, the calculation of ω_{h_a} and ω_{h_b} is similar to Eq.(7).

Learning Objective

We choose the MSE loss function and optimize the loss for multi-step forecasting together, and the loss is propagated back from the LSTM′outputs across the entire model.

$$\mathcal{L}_{CATN} = \frac{1}{T_y} \sum_{i=1}^{T_y} (Y[:, \tau] - \widehat{Y}[:, \tau])^2 \tag{9}$$

where $\tau = t_i + 1 : t_i + h$.

Experiments

Datasets, Metrics and Baselines

We employ four real-world datasets , including:

Traffic [1] Data is collected from the California Department of Transportation and describes the occupancy rate of different lanes on San Francisco highway from January, 1st, 2008 until to March, 30st, 2009. This dataset contains 963 time-series and 10560 time-points, where each observation is between 0 and 1. The sampling interval is 10 minutes.

Electricity [2] The dataset consists of 370 time-series and 25968 time-points from electricity consumption of 370 clients. All values are recorded in kW every 15 minutes.

PeMSD7(M) [3] PeMSD7(M) dataset is organized from Caltrans Performance Measurement System (PeMS) and describes the detectors spanning the freeway system across all major metropolitan areas of California. It contains 228 time-series and 11232 time-points at 5 minutes interval.

METR-LA [4] It includes traffic information from 207 sensors in the highway of Los Angeles County. It contains 207 time series and 34272 time-points at 5 minutes interval.

We employ five typical metrics, i.e., mean absolute error (MAE), root mean square error (RMSE), mean absolute percentage error (MAPE), symmetric mean absolute percentage error (SMAPE) and weighted absolute percentage error (WAPE), for time series forecasting evaluation.

We select eight methods as baselines, including: 1) Vector Auto-Regression (VAR): is an advanced statistical method that captures the linear correlation; 2) LSTNet (Lai et al. 2018): is often used to discover the short-term local dependency patterns and the long-term pattern. 3) TPA-LSTM (Shih, Sun, and Lee 2019): selects relevant time series using attention mechanism and transforms “time domain information” into “frequency domain information”; 4) MT-Net (Chang et al. 2018): is a memory time-series neural network consists of a large memory component that can capture long-term dependencies while incorporating multiple variable dimension; 5) DSANet (Huang et al. 2019): captures complicated nonlinear dependencies between time steps and between multiple time series; 6) MTGNN (Wu et al. 2020): as a graph neural network model, is often used to learn the spatial and temporal dependencies among time series; 7) DeepGLO (Sen, Yu, and Dhillon 2019): combines local and global features into temporal convolution network; 8) Informer (Zhou et al. 2021): addresses long sequence forecasting problem based on Transformer.

Overall Comparison

In Table 1, the overall prediction performances measured by three widely use indicators MAE, RMSE and MAPE, where horizon are $\{3, 6, 9, 12\}$ against other benchmarks respectively. We can observe that CATN does better than the baselines based on convolution and recurrent operation with different self-attention mechanism, as it is aided by both global&local and cross factors. Traditional VAR is not good at dealing with the long sequence time-series forecasting problem. LSTNet performs better on the electricity dataset with horizon 3 and horizon 6, but CATN shows a more stable

[1]https://archive.ics.uci.edu/ml/datasets/PEMS-SF

[2]https://archive.ics.uci.edu/ml/datasets/ElectricityLoadDiagrams20112014

[3]https://dot.ca.gov/programs/traffic-operations/mpr/pems-source

[4]http://ceur-ws.org/Vol-2750/paper9.pdf

Models	Metrics	traffic				electricity				Improvements
		3	6	9	12	3	6	9	12	
VAR	MAE	0.0252	0.0481	0.0642	0.102	0.081	0.0872	0.0963	0.111	
	RMSE	0.0374	0.0679	0.0982	0.1939	0.2408	0.2557	0.2766	0.2787	
	MAPE	0.5658	1.302	2.2563	2.5818	0.3285	0.4521	0.5252	0.5496	
LSTNet	MAE	0.0193	0.0198	0.0225	0.0229	0.0755	0.0841	0.0868	0.0839	
	RMSE	0.0847	0.0481	0.0509	0.0517	**0.2351**	**0.2445**	0.2629	0.2695	
	MAPE	0.1831	0.1897	0.237	0.2455	0.3239	0.3668	0.3753	0.4006	*
TPA-LSTM	MAE	0.0187	0.0192	0.0223	0.0237	0.0617	0.0678	0.0743	0.0739	*
	RMSE	0.031	0.0315	0.0398	0.0498	0.2481	0.2683	0.2761	0.2718	*
	MAPE	0.4681	0.7099	0.4813	0.5671	0.2696	0.3172	0.3236	0.3488	
MTNet	MAE	0.3086	0.3108	0.3071	0.3128	0.0834	0.0835	0.093	0.084	
	RMSE	0.412	0.4154	0.4112	0.4225	0.2723	0.2724	0.2843	0.273	
	MAPE	1.6268	1.6544	1.7265	1.5982	3.4273	3.4374	3.7312	3.4838	
DSANet	MAE	0.0227	0.0203	0.0235	0.0252	0.068	0.0655	0.0707	0.0711	
	RMSE	0.0322	0.0341	0.0412	0.0475	0.247	0.266	0.292	0.268	
	MAPE	0.181	0.164	0.191	0.191	0.22	0.264	0.263	0.246	
MTGNN	MAE	0.0242	0.0467	0.0289	0.03	0.0646	0.0824	0.0872	0.0767	
	RMSE	0.0523	0.0789	0.0587	0.0605	0.295	0.317	0.3229	0.3128	
	MAPE	0.2307	0.6161	0.2891	0.3231	0.2423	0.3318	0.3778	0.3409	
CATN	MAE	**0.018**	**0.0183**	**0.0193**	**0.0178**	**0.061**	**0.0613**	**0.061**	**0.0626**	11.7%
	RMSE	**0.0309**	**0.0312**	**0.0319**	**0.0308**	0.2601	0.2605	**0.2621**	**0.2649**	3.62%
	MAPE	**0.156**	**0.1589**	**0.1682**	**0.155**	**0.1907**	**0.1921**	**0.1931**	**0.2017**	39.03%

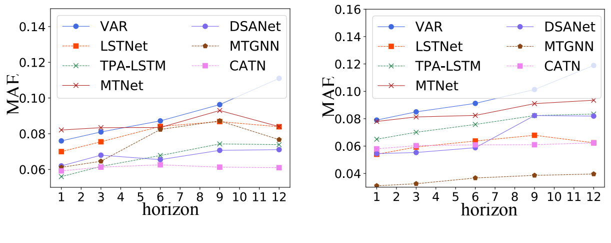
Models	Datasets	WAPE / MAPE / SMAPE			
		3	6	9	12
DeepGLO	PeMSD7(M)	0.0818 / 0.1024 / 0.0988	0.1378 / 0.1635 / 0.1651	0.0554 / 0.063 / 0.0592	0.102 / 0.1506 / 0.1245
	METR-LA	0.1506 / 0.2157 / 0.196	0.0806 / 0.068 / 0.07	0.0917 / 0.0755 / 0.0697	0.091 / 0.0749 / 0.0695
Informer	PeMSD7(M)	0.1371 / 0.1808 / 0.1543	0.1404 / 0.191 / 0.1587	0.1685 / 0.2208 / 0.186	0.1421 / 0.1972 / 0.1606
	METR-LA	0.137 / 0.2794 / 0.1693	0.1333 / 0.2682 / 0.165	0.1488 / 0.2798 / 0.1796	0.148 / 0.2877 / 0.1795
MTGNN	PeMSD7(M)	0.1771 / 0.2891 / 0.2542	0.0654 / 0.0782 / 0.0698	0.0801 / 0.0988 / 0.0836	0.0886 / 0.1129 / 0.0927
	METR-LA	0.0972 / 0.1028 / 0.1109	0.1012 / 0.1231 / 0.1276	0.1304 / 0.1465 / 0.1478	0.1504 / 0.191 / 0.1681
CATN	PeMSD7(M)	**0.0444 / 0.0486 / 0.0462**	**0.0439 / 0.0481 / 0.0457**	**0.0439 / 0.048 / 0.0457**	**0.0458 / 0.05 / 0.0476**
	METR-LA	**0.0516 / 0.0626 / 0.0556**	**0.0527 / 0.0639 / 0.0568**	**0.053 / 0.0641 / 0.057**	**0.054 / 0.0604 / 0.0576**

Table 1: Prediction results of different methods on the traffic and electricity dataset, where bold values are the best performance (smaller value means better performance). Below the main table, the WAPE/MAPE/SMAPE comparison with three baselines are reported, on the PeMSD7(M) and METR-LA dataset.

prediction ability. CATN brings 11.7%/3.62%/39.03% improvements on MAE/RMSE/MAPE to the existing best results (* in Table 1) for three datasets. CATN balances short- and long-term prediction well and achieves the best performance for almost all horizons. Specially, we find CATN perform better than the model based on GCN (MTGNN). Although MTGNN models multivariate time series data without a pre-defined graph, out-of-order graph learning is hard. CATN can learn the dynamic hierarchical and grouped correlation although there is no spatial relationship embedded from the real world. Below the main table, the comparison with two baselines are reported on PeMSD7(M) and METR-LA with the other three metrics WAPE/MAPE/SMAPE for traffic flow forecasting. We see that CATN performs best on all metrics with $>30\%$ improvements. Figure 3 further shows the different-window forecasting performance at each horizon on the electricity dataset. CATN uses fewer historical data but shows better performance.

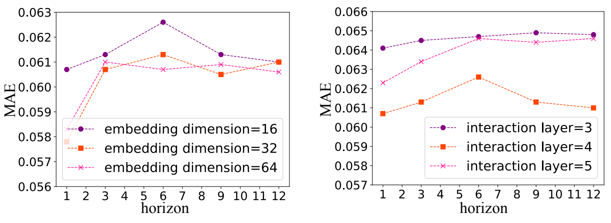
Parameter Sensitivity

We perform the sensitivity analysis of the proposed CATN model on electricity dataset. **Input Window**: In Figure 3, when obtaining short input sequences such as 1 hour , CATN shows the best performance of all the baselines. When the input window is set as 2.5 hours, MTGNN can achieve a lower MAE value. Because longer historical data contains more information to learn more accurate topological relationships between nodes. **Embedding Dimension**: In order to better align the aggregated node and edge information, we set their dimensions to the same. The embedding dimension is a key parameter in CATN because it influences the quality of the learned tree. We select three values as $\{16, 32, 64\}$ and the results are shown as Figure 4(a). We can see that excessively small or large embedding dimensions will reduce the performance. The larger embedding dimension will contain more hierarchical and grouped information, but at the same time, it will bring burdens to the upward aggrega-



(a) 1 hour (b) 2.5 hour

Figure 3: Prediction performance comparison of different windows at each horizon.



(a) Embedding Dimension (b) Interaction Layer

Figure 4: The parameter's sensitivity of CATN.

tion of the message. When the embedding dimension is set as 32, our model achieves the best performance. **Interaction Layer**: Figure 4(b) shows that selecting all nodes of the third interaction layer and inputting them into GLL gives worse results. It is possible because the node information is too centralized and loses the original information distribution. Over-smoothing problem in information aggregation makes the poor performance of the third interaction layer. The fourth and the fifth interaction layer show similar results. It is noted that a deep interaction layer can lead to the doubling of GLL parameters.

Ablation Study

To better evaluate the tree construction and embedding (TEC), global&local learning (GLL) and cross attention mechanism (CAM), we also implement additional experiments on electricity with ablation consideration. We select setting as batch_size = 64, input_window = 12, horizon = 3 in all ablation experiments. Table 2 shows the results obtained on electricity dataset. The results demonstrate the effectiveness of key components that contribute to the improved outcomes of our CATN model. In **w/o TEC**, we input data into GLL+CAM directly and output module projects the hidden features to the original dimension to get the final results. As shown in Table 2, the RMSE of **w/o TEC** is significantly improved (+18.03%). As verified by **w/o TEC**, mining hierarchical and grouped correlations between different series can lead to better performance. In **w/o GLL**, we use a pure recurrent module ignoring local information learning instead of GLL. We prove that capturing global and local temporal patterns jointly based on intra-series data can bring benefits. Moreover, **w/o CAM** indicates weak improvements on all metrics. When using the CAM, our model is able to emphasize relevant information and extract more discriminative features compared to extracting independently.

Metrics	**CATN**	w/o TEC	w/o GLL	w/o CAM
MAE	0.061	0.0649	0.0613	0.0615
RMSE	0.2601	0.3173	0.2603	0.2603
MAPE	0.1907	0.216	0.191	0.1934
SMAPE	0.174	0.1966	0.1755	0.1767

Table 2: Results of ablation study on the electricity dataset

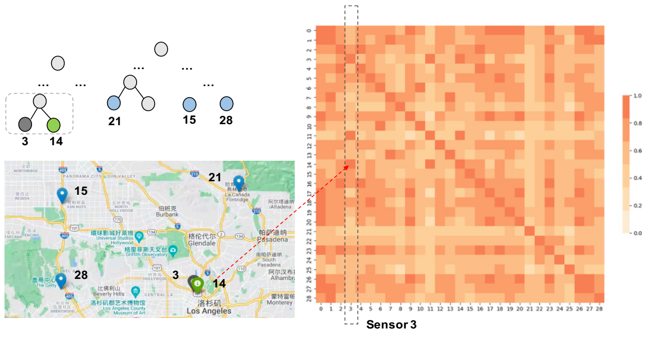


Figure 5: Visualization of computed correlation matrix in TEC on the METR-LA dataset

Analysis

To investigate the validity of hierarchical clustering in our tree construction, we execute a case study in the traffic forecasting scenarios. We select 5 sensors from METR-LA (Li et al. 2018) on Google Map and show their corresponding position in the tree (the upper left part in Figure 5). Sensor 3 and sensor 4 belong to the same parent node. Sensor 21, 15 and 28 are at the same level of the tree but belong to different parent nodes. Furthermore, the correlation matrix learned from the training data in a window is shown clearly (the right part of Figure 5). Each column represents a sensor in the real world. Column i denotes the correlation of sensor i and other sensors and self-correlation are set as 1 by experience. As shown in the figure, the values of sensor 3 and sensor 14 are high and it indicates they are closely related. This is attributable that sensor 14 is located on the road of 101, while sensor 3 is located at the intersection near sensor 14. Hence, our model has both excellent prediction performance and reasonable interpretability.

Conclusion

In this paper, we studied the hierarchical and grouped correlation mining problem of multivariate time-series data and proposed CATN for multi-step forecasting. To the best of our knowledge, this is the first work to learn inter-series hierarchical and grouped correlation via a tree-based deep learning approach. We designed a multi-level learning mechanism to capture long-, short-range and cross temporal patterns for intra-series data. Our CATN demonstrates extraordinary performance on four datasets from real world and applies to time-series data without a predefined topology. It is noted that CATN is applicable for many real-world collections, especially for those with explicit hierarchical and grouped correlation, e.g., including but not limited to stock, solar, covid-19, weather, etc.

Acknowledgments

This work is supported in part by National Key R&D Program of China (2019YFB1406302).

References

Bai, L.; Yao, L.; Li, C.; Wang, X.; and Wang, C. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. In *NeurIPS*.

Boroujeni, F. R.; Wang, S.; Li, Z.; West, N.; Stantic, B.; Yao, L.; and Long, G. 2018. Trace Ratio Optimization With Feature Correlation Mining for Multiclass Discriminant Analysis. In *AAAI*, 2746–2753. AAAI Press.

Cao, D.; Wang, Y.; Duan, J.; Zhang, C.; Zhu, X.; Huang, C.; Tong, Y.; Xu, B.; Bai, J.; Tong, J.; and Zhang, Q. 2020. Spectral Temporal Graph Neural Network for Multivariate Time-series Forecasting. In *NeurIPS*.

Chang, Y.; Sun, F.; Wu, Y.; and Lin, S. 2018. A Memory-Network Based Solution for Multivariate Time-Series Forecasting. *CoRR*, abs/1809.02105.

Cheng, Z.; Yuan, C.; Li, J.; and Yang, H. 2018. TreeNet: Learning Sentence Representations with Unconstrained Tree Structure. In *IJCAI*, 4005–4011. ijcai.org.

Du, S.; Li, T.; Yang, Y.; and Horng, S. 2020. Multivariate time series forecasting via attention-based encoder-decoder framework. *Neurocomputing*, 388: 269–279.

Du, Y.; Wang, J.; Feng, W.; Pan, S.; Qin, T.; Xu, R.; and Wang, C. 2021. AdaRNN: Adaptive Learning and Forecasting of Time Series. *CoRR*, abs/2108.04443.

Emmendorfer, L. R.; and de Paula Canuto, A. M. 2021. A generalized average linkage criterion for Hierarchical Agglomerative Clustering. *Appl. Soft Comput.*, 100: 106990.

Faloutsos, C.; Flunkert, V.; Gasthaus, J.; Januschowski, T.; and Wang, Y. 2019. Forecasting Big Time Series: Theory and Practice. In *KDD*, 3209–3210. ACM.

Gao, R.; Duru, O.; and Yuen, K. F. 2021. High-dimensional lag structure optimization of fuzzy time series. *Expert Syst. Appl.*, 173: 114698.

Hao, Y.; Zhang, Y.; Liu, K.; He, S.; Liu, Z.; Wu, H.; and Zhao, J. 2017. An End-to-End Model for Question Answering over Knowledge Base with Cross-Attention Combining Global Knowledge. In *ACL (1)*, 221–231. Association for Computational Linguistics.

Hou, R.; Chang, H.; Ma, B.; Shan, S.; and Chen, X. 2019. Cross Attention Network for Few-shot Classification. In *NeurIPS*, 4005–4016.

Hu, L.; Jian, S.; Cao, L.; and Chen, Q. 2018. Interpretable Recommendation via Attraction Modeling: Learning Multilevel Attractiveness over Multimodal Movie Contents. In *IJCAI*, 3400–3406. ijcai.org.

Huang, S.; Wang, D.; Wu, X.; and Tang, A. 2019. DSANet: Dual Self-Attention Network for Multivariate Time Series Forecasting. In *CIKM*, 2129–2132. ACM.

Lai, G.; Chang, W.; Yang, Y.; and Liu, H. 2018. Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks. In *SIGIR*, 95–104. ACM.

Lee, J.; Jain, M.; Park, H.; and Yun, S. 2021. Cross-Attentional Audio-Visual Fusion for Weakly-Supervised Action Localization. In *ICLR*. OpenReview.net.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *ICLR (Poster)*. OpenReview.net.

Lin, W.; Deng, Y.; Gao, Y.; Wang, N.; Zhou, J.; Liu, L.; Zhang, L.; and Wang, P. 2021. CAT: Cross-Attention Transformer for One-Shot Object Detection. *CoRR*, abs/2104.14984.

Lyu, Y.; Li, M.; Huang, X.; Guler, U.; Schaumont, P.; and Zhang, Z. 2020. TreeRNN: Topology-Preserving Deep Graph Embedding and Learning. In *ICPR*, 7493–7499. IEEE.

Mancuso, P.; Piccialli, V.; and Sudoso, A. M. 2020. A machine learning approach for forecasting hierarchical time series. *CoRR*, abs/2006.00630.

Maxim, L. G.; Rodriguez, J. I.; Wang, B.; and . 2020. Defect of Euclidean distance degree. *Adv. Appl. Math.*, 121: 102101.

Rangapuram, S. S.; Werner, L. D.; Benidis, K.; Mercado, P.; Gasthaus, J.; and Januschowski, T. 2021. End-to-End Learning of Coherent Probabilistic Forecasts for Hierarchical Time Series. In *ICML*, volume 139, 8832–8843. PMLR.

Salinas, D.; Bohlke-Schneider, M.; Callot, L.; Medico, R.; and Gasthaus, J. 2019. High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes. In *NeurIPS*, 6824–6834.

Sen, R.; Yu, H.; and Dhillon, I. S. 2019. Think Globally, Act Locally: A Deep Neural Network Approach to High-Dimensional Time Series Forecasting. In *NeurIPS*, 4838–4847.

Shi, K.; Wang, Y.; Lu, H.; Zhu, Y.; and Niu, Z. 2021. EKGTF: A knowledge-enhanced model for optimizing social network-based meteorological briefings. *Inf. Process. Manag.*, 58(4): 102564.

Shih, S.; Sun, F.; and Lee, H. 2019. Temporal pattern attention for multivariate time series forecasting. *Mach. Learn.*, 108(8-9): 1421–1441.

Wan, S.; and Niu, Z. 2020. A Hybrid E-Learning Recommendation Approach Based on Learners' Influence Propagation. *IEEE Trans. Knowl. Data Eng.*, 32(5): 827–840.

Wang, S.; Hu, L.; Cao, L.; Huang, X.; Lian, D.; and Liu, W. 2018a. Attention-Based Transactional Context Embedding for Next-Item Recommendation. In *AAAI*, 2532–2539. AAAI Press.

Wang, X.; He, X.; Feng, F.; Nie, L.; and Chua, T. 2018b. TEM: Tree-enhanced Embedding Model for Explainable Recommendation. In *WWW*, 1543–1552. ACM.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; Chang, X.; and Zhang, C. 2020. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *KDD*, 753–763. ACM.

Xu, D.; Cheng, W.; Zong, B.; Song, D.; Ni, J.; Yu, W.; Liu, Y.; Chen, H.; and Zhang, X. 2020. Tensorized LSTM with

Adaptive Shared Memory for Learning Trends in Multivariate Time Series. In *AAAI*, 1395–1402. AAAI Press.

Yang, Z.; Yan, W. W.; Huang, X.; and Mei, L. 2020. Adaptive Temporal-Frequency Network for Time-Series Forecasting. *IEEE Trans. Knowl. Data Eng.*, PP(99): 1–1.

Yu, B.; Yin, H.; and Zhu, Z. 2018. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *IJCAI*, 3634–3640.

Zhang, Q.; Cao, L.; Shi, C.; and Niu, Z. 2021. Neural Time-Aware Sequential Recommendation by Jointly Modeling Preference Dynamics and Explicit Feature Couplings. *IEEE Transactions on Neural Networks and Learning Systems*, 1–13.

Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; and Zhang, W. 2021. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. In *AAAI*, 11106–11115. AAAI Press.

Zhu, Y.; Lin, Q.; Lu, H.; Shi, K.; Liu, D.; Chambua, J.; Wan, S.; and Niu, Z. 2021. Recommending Learning Objects through Attentive Heterogeneous Graph Convolution and Operation-Aware Neural Network. *IEEE Transactions on Knowledge and Data Engineering*.

Zhuang, D. E. H.; Li, G. C. L.; and Wong, A. K. C. 2014. Discovery of Temporal Associations in Multivariate Time Series. *IEEE Trans. Knowl. Data Eng.*, 26(12): 2969–2982.

Zhuang, Z.; Yeh, C. M.; Wang, L.; Zhang, W.; and Wang, J. 2020. Multi-stream RNN for Merchant Transaction Prediction. *CoRR*, abs/2008.01670.