CAT: Interpretable Concept-based Taylor Additive Models

Viet Duong
William & Mary
Williamsburg, VA, United States
vqduong@wm.edu

Qiong Wu
AT&T CDO
Bedminster, NJ, United States
qw6547@att.com

Zhengyi Zhou
AT&T CDO
Bedminster, NJ, United States
zz547k@att.com

Hongjue Zhao*
University of Illinois at Urbana-Champaign
Champaign, IL, United States
hongjue2@illinois.edu

Chenxiang Luo
William & Mary
Williamsburg, VA, United States
cluo02@wm.edu

Eric Zavesky
AT&T CDO
Austin, TX, United States
ez2685@att.com

Huaxiu Yao
The University of North Carolina at Chapel Hill
Chapel Hill, NC, United States
huaxiu@cs.unc.edu

Huajie Shao*
William & Mary
Williamsburg, VA, United States
hshao@wm.edu

Abstract

As an emerging interpretable technique, Generalized Additive Models (GAMs) adopt neural networks to individually learn non-linear functions for each feature, which are then combined through a linear model for final predictions. Although GAMs can explain deep neural networks (DNNs) at the feature level, they require large numbers of model parameters and are prone to overfitting, making them hard to train and scale. Additionally, in real-world datasets with many features, the interpretability of feature-based explanations diminishes for humans. To tackle these issues, recent research has shifted towards concept-based interpretable methods. These approaches try to integrate concept learning as an intermediate step before making predictions, explaining the predictions in terms of human-understandable concepts. However, these methods require domain experts to extensively label concepts with relevant names and their ground-truth values. In response, we propose CAT, a novel interpretable Concept-bAsed Taylor additive model to simplify this process. CAT does not require domain experts to annotate concepts and their ground-truth values. Instead, it only requires users to simply categorize input features into broad groups, which can be easily accomplished through a quick metadata review. Specifically, CAT first embeds each group of input features into one-dimensional high-level concept representation, and then feeds the concept representations into a new white-box Taylor Neural Network (TaylorNet). The TaylorNet aims to learn the non-linear relationship between the inputs and outputs using polynomials. Evaluation results across multiple benchmarks demonstrate that CAT can outperform or compete with the baselines while reducing the need of extensive model parameters. Importantly, it can effectively explain model predictions through high-level concepts. Source code is available at `github.com/vduong143/CAT-KDD-2024`.

CCS Concepts

• **Computing methodologies** → *Learning latent representations*; **Machine learning approaches**.

Keywords

Interpretable Machine Learning; Concept-based Learning; Neural Additive Models

ACM Reference Format:
Viet Duong, Qiong Wu, Zhengyi Zhou, Hongjue Zhao*, Chenxiang Luo, Eric Zavesky, Huaxiu Yao, and Huajie Shao*. 2024. CAT: Interpretable Concept-based Taylor Additive Models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3637528.3672020

1 Introduction

While deep neural networks (DNNs) have demonstrated remarkable success in various areas, the lack of interpretability impedes their deployment in high-stakes applications, such as autonomous vehicles, finance, and healthcare [3]. Thus, enhancing DNN interpretability has emerged as a pivotal area of research in recent years.

Earlier studies primarily focused on perturbation-based post-hoc approaches [7, 21, 35], but these methods are either computationally expensive or hard to faithfully represent the model's behavior [15, 40, 44]. To address these issues, recent works have shifted focus to Generalized Additive Models (GAMs) [1, 8, 14, 37]. GAMs aims to learn non-linear transformation of input features separately into smoothed structures known as shape functions, and then use a linear combination of these functions to make predictions. Despite their potential, GAMs require extensive model parameters and suffer from scalability issue, since they use separate DNNs or an ensemble of numerous decision trees [8, 33] to learn

*Corresponding author.

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

ACM ISBN 979-8-4007-0490-1/24/08
https://doi.org/10.1145/3637528.3672020

the shape function of each feature. Furthermore, while GAMs offer insights into the significance and behavior of individual features via shape function visualizations, these explanations may not always be readily interpretable for humans.

In contrast, human explanations often rely on concept-based reasoning, which semantically groups low-level features into broader concepts, and then explains decisions using these high-level concepts. For example, in medical diagnostics like diabetes, physicians usually explain their conclusions by referring to high-level factors, such as family history, medical history, dietary patterns, and blood tests. This approach has spurred research into integrating concept-based interpretability into DNNs. Specifically, concept-based interpretable methods introduce an intermediate step to learn human-understandable concepts from input features, which then inform predictions made by white-box predictors like linear models or decision trees. Yet, these methods require domain experts to label extensive concepts and their ground-truth values, e.g., categorizing blood test results on a scale from 0 to 71 using the APACHE II scoring system [2].

To overcome these limitations, we propose a novel interpretable concept-based Taylor additive model, called CAT, that can explain predictions using high-level concepts without relying heavily on domain experts to label concepts and their ground-truth values. As shown in Fig. 1, the proposed CAT consists of two main components: (i) concept encoders and (ii) a white-box Taylor Neural Network (TaylorNet). Specifically, the concept encoders aim to learn high-level concept representations from low-level features, where each encoder produces a one-dimensional representation from a cluster of features. TaylorNet aims to approximate non-linear functions using polynomials without activation functions. It directly learns the relationship between the input and output with polynomials, largely improving the interpretability. A significant challenge is to reduce the computational complexity of TaylorNet with high-order polynomials. To overcome this, we adopt Tucker decomposition [26, 47] to decompose the higher-order coefficients in Taylor expansion into a set of low-rank tensors.

We assess the proposed CAT across multiple benchmark datasets for tabular and visual reasoning tasks. The evaluation results demonstrate the good performance of our method on these datasets. It can achieve higher or comparable accuracy to the best baseline with a reduction in the number of required model parameters. Furthermore, we demonstrated the efficacy of our concept encoder by integrating it with other interpretable baselines. Importantly, CAT offers improved explanation capabilities by articulating model predictions through human-understandable concepts.

In summary, the main contributions of this work include: (1) We introduce CAT, a novel interpretable framework capable of explaining DNNs predictions through high-level concepts; (2) We develop a white-box TaylorNet to directly learn the relationships between the input and output with polynomials, largely enhancing interpretability; (3) Extensive experimental results demonstrate that our method outperforms or competes with the baselines on six benchmark datasets, achieving this with fewer or a similar number of model parameters; and (4) We present a case study illustrating how CAT allow users to comprehend model predictions by categorizing input features into concepts using basic data understanding and describing predictions in terms of Taylor polynomial.

2 Related Works

Classical Interpretable Methods. Early works on explaining black-box machine learning models mainly adopted perturbation-based post-hoc approaches [34, 39] to estimate the feature importance. One typical example of these methods is LIME, which explains individual predictions of a neural network by approximating it with interpretable models, such as linear models or decision trees fitted over simulated data points by randomly perturbing the given inputs [39]. However, recent research [15, 44] showed that post-hoc approaches could be unfaithful to the predictions of the original model, and are computationally expensive [43].

Generalized Additive models (GAMs). Some recent works have focused on a new line of interpretable machine learning methods, called Generalized Additive Models (GAMs). The basic idea of GAMs is to learn non-linear shape function of each input feature using a separate DNN and then use a linear combination of these shape functions to predict results. Since each input feature learned by DNN is independent, it can help us explain the predictions based on the corresponding shape function. For instance, some representative models, such as Neural Additive Models (NAM) [1] and its tree-based variant NODE-GAM [8], learn feature-wise shape functions using a separate DNN or oblivious decision tree ensembles for each individual feature. However, these methods requires a large number of parameters and are easy to suffer from overfitting. To overcome this challenge, some researchers introduced Neural Basis Models (NBM) [37], which use a single DNN to learn shared bases and then input the bases into one linear model for each feature to learn its shape function. By doing this, NBM can improve its scalability to many input features. More recently, Scalable Polynomial Additive Models (SPAM) was proposed to incorporate high-order feature interactions for prediction and explanation [14]. Another recent study adopted soft decision trees with hierarchical constraints to learn sparse pairwise interactions, improving the scalability and interpretability of tree-based GAMs [19]. However, these approaches try to explain the predictions from low-level features instead of high-level concepts that humans can easily understand. In particular, when the number of input features is large, it is still hard to fully interpret the prediction results.

Concept-based Interpretability. Our proposed method is closely related to concept-based learning, where models learn intermediate mappings from raw inputs to human-specified high-level concepts, then use only these annotated concepts to make final predictions. Recently, concept-based models have emerged as an important interpretable machine learning framework for many applications, such as medical diagnosis [12, 13, 16, 25], visual question answering [5, 50], and image recognition [10, 25, 31]. For example, Koh et al. [25] proposed Concept Bottleneck Model (CBM) to predict concepts annotated by medical experts from X-ray image data. However, this approach may result in low accuracy when concept labels do not contain all the necessary information for downstream tasks [25]. To deal with this problem, Mahinpei et al. [36] proposed to use an additional set of unsupervised concepts to improve the accuracy at the cost of the interpretability. Then, some researchers tried to trade-off the interpretability and accuracy in a follow-up work [18, 51]. However, existing work are heavily dependent on the human annotated concepts with values, which is infeasible in

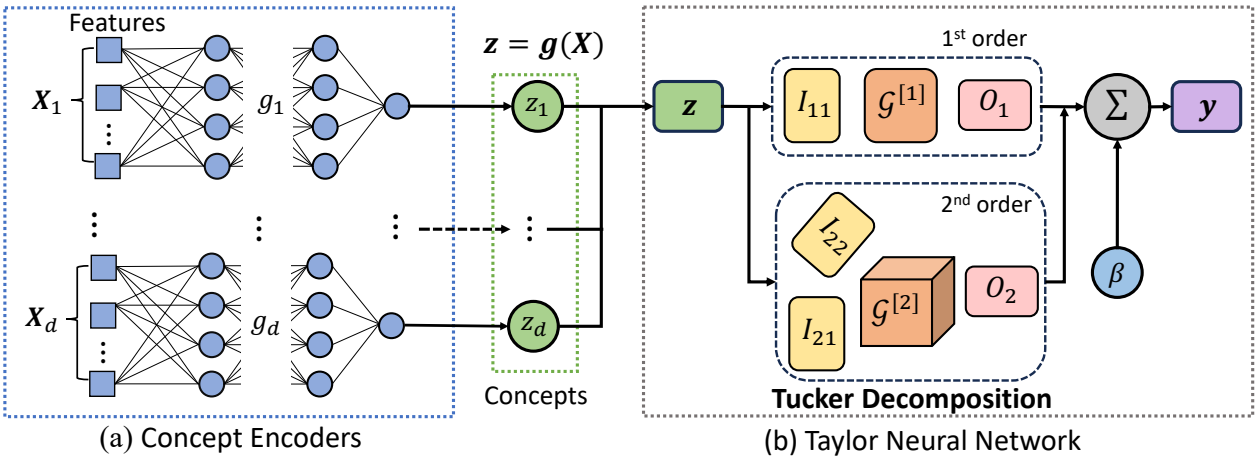


Figure 1: The Overall framework of CAT. It consists of two main components: concept encoders and Taylor Neural Networks (TaylorNet). Each concept encoder embeds a group of low-level features into a one-dimensional high-level concept representation. The TaylorNet is a white-box model that uses the high-level concept representations to make predictions.

many real-world settings. Different from prior works, we propose to group input variables into high-level concepts based on categories without imposing specific values on them, thereby reducing the cost of human annotations and making it more practical and flexible to real-world applications.

3 Preliminaries

In this section, we first describe the research problem, and then review the basic knowledge of Taylor series expansion and Tucker decomposition.

3.1 Problem Definition

Given a multivariate input data $\boldsymbol{X}$, our goal is to learn a prediction model defined by function $\boldsymbol{h} : \boldsymbol{X} \rightarrow \boldsymbol{y}$ that maps $\boldsymbol{X}$ to the vector $\boldsymbol{y}$ of target labels representative of a regression or classification problem. In this work, we consider the problem of creating an interpretable model that can explain its predictions based on abstractions of the input features known as concepts. To this end, we assume that the data come with some descriptive information or metadata about the input features, so that we can manually and/or empirically divide the input space $\boldsymbol{X}$ into d groups of features $\{\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_d\}$ representing high-level concepts. In order to condition the prediction of target variable $\boldsymbol{y}$ on high-level concepts, we decompose the original function $\boldsymbol{h}$ into 2 functions: concept encoders $\boldsymbol{g}$ and target predictor $\boldsymbol{f}$, such that $\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$. In particular, concept encoders $\boldsymbol{g} = \{g_m : X_m \rightarrow z_m | m = 1, \cdots, d\}$ consist of m encoders, each g_m individually maps one feature group $\boldsymbol{X}_m$ to a 1-D *scalar* representation z_m. Then, the concept representations are combined into an intermediate concept vector $\boldsymbol{z} = \{z_1, \ldots, z_m\}$. Finally, target predictor $\boldsymbol{f} : \boldsymbol{z} \rightarrow \boldsymbol{y}$ uses concept vector $\boldsymbol{z}$ as input to predict the target $\boldsymbol{y}$. If the learned concepts $\boldsymbol{z}$ are semantically meaningful and the predictor $\boldsymbol{f}$ is interpretable, then humans can interpret the model's decision process by attributing its predictions to the relevant concepts. Table 1 summarizes the main notations that will be used throughout the paper.

Table 1: Summary of notations.

Notation	Definition
$\boldsymbol{X}$	multivariate input matrix
d	number of concept groups
$\boldsymbol{z}$	concept embedding vector
$\boldsymbol{f} : \boldsymbol{z} \rightarrow \boldsymbol{y}$	mapping function from concepts to target variable(s)
$\boldsymbol{g} = \{g_m : X_m \rightarrow z_m \mid m = 1, \cdots, d\}$	set of concept encoders, each operates on one group of features $\boldsymbol{X}_m$, $m = 1, \ldots, d$.
$\boldsymbol{h} = \boldsymbol{f} \circ \boldsymbol{g}$	function representing the entire framework, which is $\boldsymbol{f}$ composed with $\boldsymbol{g}$ in our method.
$\otimes, \times_n, \bar{\times}_n$	Kronecker product, mode-n matrix product, mode-n vector product
k, N	term order of Taylor polynomial, the total order of Taylor polynomial
$\Delta \boldsymbol{z}$	Input of TaylorNet
$\boldsymbol{\mathcal{G}}^{[k]}$	Learnable core tensor of TaylorNet

3.2 Taylor Polynomials

In mathematics, Taylor's theorem [46] states that given a vector-valued multivariate function $\boldsymbol{f} : \mathbb{R}^d \rightarrow \mathbb{R}^o$, its approximation using a Taylor polynomial of order N at a point $\boldsymbol{z} = \boldsymbol{z}_0$ is given by:

$$f(z) \approx \sum_{k=0}^{N} \frac{1}{k!} \left[\sum_{j=1}^{d} \left(\Delta z_j \frac{\partial}{\partial z_j} \right) \right]^k f \Big|_{z_0}, \tag{1}$$

where $\boldsymbol{z} \in \mathbb{R}^d$, $\boldsymbol{x}_0 \in \mathbb{R}^d$, $\Delta z_j = z_j - z_{j,0}$, and $j = 1, \cdots, d$. This Taylor polynomial can also be expressed as the followig tensor form [11]:

$$f(z) \approx f(z_0) + \sum_{k=1}^{N} \left(\boldsymbol{\mathcal{W}}^{[k]} \prod_{j=2}^{k+1} \bar{\times}_j \Delta z^\top \right), \tag{2}$$

where $\bar{\times}_n$ denotes the *mode-n matrix product* [26], $\Delta \boldsymbol{z} = \boldsymbol{z} - \boldsymbol{z}_0$, and parameter tensors $\boldsymbol{\mathcal{W}}^{[k]} \in \mathbb{R}^{o \prod_{n=1}^{k} \times d}$ for $k = 1, \cdots, N$ are

the k-th order *scaled derivatives* of $\boldsymbol{f}$ at point $\boldsymbol{z} = \boldsymbol{z}_0$. According to the Stone–Weierstrass theorem [45], polynomials defined in Eq. 2 can approximate any continuous function defined in a closed interval as closely as desired. In general, Taylor polynomials of higher order produce more precise approximations. However, as the order increases, the computational complexity grows exponentially, leading to scalability issues in high-dimensional input data.

3.3 Tucker Decomposition

Tucker decomposition is a tensor decomposition technique [26], aiming to decompose a tensor into a set of factor matrices and a small core tensor [47]. Fundamentally, Tucker decomposition can be considered as a generalization of principal component analysis to higher-order analysis. In particular, given an N-way tensor $\boldsymbol{\mathcal{W}}$, the Tucker decomposition of $\boldsymbol{\mathcal{W}}$ is given by:

$$\boldsymbol{\mathcal{W}} = \boldsymbol{\mathcal{G}} \times_1 \boldsymbol{U}^{(1)} \times_2 \boldsymbol{U}^{(2)} \times_3 \cdots \times_N \boldsymbol{U}^{(N)}, \tag{3}$$

where $\boldsymbol{\mathcal{G}}$ is the N-way core tensor, and $\boldsymbol{U}^{(k)}$ for $k = 1, \cdots, N$ are the factor matrices along corresponding mode k. Some researchers [26] alternatively express Eq. 3 in matricized form using Kronecker products as:

$$\boldsymbol{W}_{(k)} = \boldsymbol{U}^{(k)} \boldsymbol{G}_{(k)} \left(\boldsymbol{U}^{(N)} \otimes \cdots \otimes \boldsymbol{U}^{(k+1)} \otimes \boldsymbol{U}^{(k-1)} \otimes \cdots \otimes \boldsymbol{U}^{(1)} \right)^{\mathrm{T}}, \tag{4}$$

where $\otimes$ denotes Kronecker product, and $\boldsymbol{W}_{(k)}$ and $\boldsymbol{G}_{(k)}$ are the mode-k matricization of the tensors $\boldsymbol{\mathcal{W}}$ and $\boldsymbol{\mathcal{G}}$ respectively.

4 Methodology

In this section, we first introduce the motivation behind learning high-level concepts from input features as the foundation of an interpretable machine learning framework. Given the learned concepts, we propose an interpretable TaylorNet, an expressive approximation algorithm, to capture concept semantics and interactions for building accurate prediction models, as illustrated in Fig. 1. To reduce the computational costs of TaylorNet, we adopt Tucker decomposition to decompose the higher-order coefficients in Taylor expansion into a set of low-rank tensors.

4.1 Concept Encoders

Generalized Additive Models (GAMs) [17] is an emerging paradigm that can interpret machine learning models at a feature level. The intuition behind these models is similar to that of linear regression, which learns the approximation $\boldsymbol{h}(\boldsymbol{X})$ of dependent variable $\boldsymbol{y}$ by parameterizing a linear combination $\beta_0 + \sum_{i=1}^{n} \beta_i X_i$ of input features $\boldsymbol{X} = X_1, \cdots, X_n$. In typical GAMs, each $\beta_i X_i$ is replaced by shape function $s_i(X_i)$ that transforms X_i into a smooth representation, such that the sum of $s_i(X_i)$'s is the generalized smooth estimate of $\boldsymbol{y}$. Furthermore, GAMs can be extended to model pairwise and higher-order feature interactions [33] for improved accuracy. In this case, $\boldsymbol{h}(\boldsymbol{X})$ is given by:

$$\boldsymbol{h}(\boldsymbol{X}) = \beta_0 + \sum_{i=1}^{n} s_i(X_i) + \sum_{j \neq i} s_{ij}(X_i, X_j) + \cdots + s_{1 \cdots n}(\boldsymbol{X}), \tag{5}$$

where $s_{ij}(X_i, X_j)$ in the third term is the second-order or pairwise interaction between X_i and X_j, and the subsequent terms denotes higher-order transformation of up to n-way feature interaction among all X_i's. Evidently, as the order of feature interaction increases, high-order GAMs approach the level of expressiveness closer to that of fully-connected DNNs, leading to better performance. However, beyond third-order interactions, these models will be hardly interpretable [33], especially when the number of features n is large. In addition, we observe that among all the input features and their high-order interactions, hardly all, or only a few of them are meaningful to the prediction accuracy and human interpretation. Therefore, in order to limit the degree of interaction between input variables to those that are relevant and intelligible to humans, we propose to learn high-level concepts from each group of semantically related features.

Given a multivariate input space $\boldsymbol{X}$, we aim to partition $\boldsymbol{X}$ into d groups of features $\{\boldsymbol{X}_1, \boldsymbol{X}_2, \cdots, \boldsymbol{X}_d\}$ that represent the corresponding high-level concepts. For ease of explanation, we consider a general application of concept-based method on tabular data. Specifically, we assume that the studied datasets are accompanied by some form of metadata detailing the meaning of corresponding input features. Given the metadata, we can group closely related features into high-level concepts manually and/or in conjunction with possible analyses on empirical similarity and correlation of feature metadata or values [27, 49]. Since these concept groups convey high-level representations for their low-level features, we can disregard the less meaningful terms in Eq. 5 such that $\boldsymbol{h}(\boldsymbol{X})$ can be approximated as the sum of concepts representations and their high-order interactions:

$$\boldsymbol{h}(\boldsymbol{X}) \approx \beta_0 + \sum_{k=1}^{d} s_k(\boldsymbol{X}_k) + \sum_{l \neq k} s_{kl}(\boldsymbol{X}_k, \boldsymbol{X}_l) + \cdots + s_{1 \cdots d}(\boldsymbol{X}_1, \cdots, \boldsymbol{X}_d), \tag{6}$$

where the first term is the sum of high-level concept representations for each group of closely related features, and each following term represents the interaction among groups of concepts. If the number of concepts d is small, then the order of interaction within the model is much lower than in Eq. 5, leading to a more interpretable model. Additionally, interactions among concepts can serve as the proxy for the interactions among enclosed features, allowing humans to interpret the interactions among a large number of features at an abstract level.

To learn the latent vector representation $\boldsymbol{z}$ of the high-level concepts from tabular data, we utilize an ensemble of d DNN concept encoders $\boldsymbol{g} = \{g_m : \boldsymbol{X}_m \rightarrow \boldsymbol{z}_m | m = 1, \cdots, d\}$, each operating on a group of features as illustrated in Fig. 1(a) to obtain intermediate concepts $\boldsymbol{z} = \boldsymbol{g}(\boldsymbol{X})$. For higher-dimensional input data such as 2D image, this is achieved by using specifically designed concept discovery algorithms such as disentangled representation learning [24, 41, 42], which can extract the disentangled latent factors $\boldsymbol{z}$ representing high-level visual concepts from images. Furthermore, we can substitute $\boldsymbol{f}(\boldsymbol{z}) = \boldsymbol{f}(\boldsymbol{g}(\boldsymbol{X})) = \boldsymbol{h}(\boldsymbol{X})$ into Eq. 6. Then by defining each kth-order concept interaction as the product of k corresponding concept embeddings and a weight parameter, $\boldsymbol{f}(\boldsymbol{z})$ can be expressed in polynomial form of order $N \leq d$ as follows:

$$\boldsymbol{f}(\boldsymbol{z}) \approx \beta_0 + \sum_{k=1}^{N} \left(\boldsymbol{\mathcal{W}}^{[k]} \prod_{j=2}^{k+1} \bar{\times}_j \boldsymbol{z} \right), \tag{7}$$

where parameter tensors $\boldsymbol{\mathcal{W}}^{[k]} \in \mathbb{R}^{o \prod_{m=1}^{k} \times d}$ for $k = 1, \cdots, N$ characterize the k-th order concept interactions.

However, one major challenge is that the Taylor polynomial in Eq. 7 will be computationally expensive as its order is large. To deal with this problem, we propose to develop a novel TaylorNet based on Tucker decomposition, which allows each mode to have more possible interactions between the latent factors, so that it is more expressive than CP tensor decomposition in existing works [11, 14].

4.2 Learning Predictive Models with TaylorNet

We aim to develop a Taylor Neural Network (TaylorNet) as given in Eq. 2 to approximate function $\boldsymbol{f}$ with latent concept vector $\boldsymbol{z}$ as input. Since $\boldsymbol{f}$ is unknown at training time, we consider $\boldsymbol{f}(\boldsymbol{z}_0)$ and $\boldsymbol{\mathcal{W}}^{[k]}$'s as *learnable parameters*. As tensors $\boldsymbol{\mathcal{W}}^{[k]}$ denote the k-th order scaled derivatives of function $\boldsymbol{h}$, the number of parameters required to learn $\boldsymbol{\mathcal{W}}^{[k]}$ grows exponentially with respect to polynomial order k (i.e., $\mathcal{O}(d^k)$). To overcome this issue, we adopt Tucker decomposition on $\boldsymbol{\mathcal{W}}^{[k]}$ following Eq. 3 as follows:

$$\begin{aligned}\boldsymbol{\mathcal{W}}^{[k]} &= \boldsymbol{\mathcal{G}}^{[k]} \times_1 \boldsymbol{O}_k \times_2 \boldsymbol{I}_{k1} \cdots \times_{k+1} \boldsymbol{I}_{kk} \\ &= \boldsymbol{\mathcal{G}}^{[k]} \times_1 \boldsymbol{O}_k \prod_{j=1}^{k} \times_{j+1} \boldsymbol{I}_{kj},\end{aligned} \tag{8}$$

where $\boldsymbol{\mathcal{G}}^{[k]} \in \mathbb{R}^{r_{out,k} \prod_{j=1}^{k} \times r_{in,k,j}}$ is the core tensor; $\boldsymbol{I}_{kj} \in \mathbb{R}^{d \times r_{in,k,j}}$ and $\boldsymbol{O}_k \in \mathbb{R}^{o \times r_{out,k}}$ for $j = 1, \ldots, k$ are input and output factor matrices respectively. For each k-th-order term of the Taylor polynomial, $r_{in,k,j}$ and $r_{out,k}$ are denoted as the ranks of Tucker decomposition corresponding to the j-th input and output dimension.

We further substitute Eq. 8 into Eq. 2, such that the k-th term of Taylor polynomial can be written as:

$$\boldsymbol{\mathcal{W}}^{[k]} \prod_{j=2}^{k+1} \times_j \Delta \boldsymbol{z}^\top = \boldsymbol{\mathcal{G}}^{[k]} \times_1 \boldsymbol{O}_k \left(\prod_{i=1}^{k} \times_{i+1} \boldsymbol{I}_{ki} \right) \left(\prod_{j=1}^{k} \times_{j+1} \Delta \boldsymbol{z}^\top \right) \tag{9}$$

Subsequently, we apply the commutative and associative properties of mode-n product [26] on Eq. 9 yielding:

$$\boldsymbol{\mathcal{W}}^{[k]} \prod_{j=2}^{k+1} \times_j \Delta \boldsymbol{z}^\top = \boldsymbol{\mathcal{G}}^{[k]} \times_1 \boldsymbol{O}_k \left[\prod_{j=1}^{k} \times_{j+1} \left(\Delta \boldsymbol{z}^\top \boldsymbol{I}_{kj} \right) \right] \in \mathbb{R}^{o}. \tag{10}$$

Since current deep learning frameworks such as Pytorch and Tensorflow do not support batch-wise mode-n multiplication, we additionally follow the mode-n unfolding rule as previously demonstrated by Eq. 3 and Eq. 4 to rewrite Eq. 10 using Kronecker products for ease of implementation. We obtain the following:

$$\boldsymbol{\mathcal{W}}^{[k]} \prod_{j=2}^{k+1} \times_j \Delta \boldsymbol{z}^\top = \boldsymbol{O}_k \boldsymbol{G}_k \left[\left(\boldsymbol{I}_{kk}^\top \Delta \boldsymbol{z} \right) \otimes \cdots \otimes \left(\boldsymbol{I}_{k1}^\top \Delta \boldsymbol{z} \right) \right], \tag{11}$$

where $\boldsymbol{G}_k = \boldsymbol{G}_{(1)}^{[k]}$ denotes the mode-1 matricization of tensor $\boldsymbol{\mathcal{G}}^{[k]}$.

Afterwards, the remaining step is to substitute the above Eq. 11 into the original Taylor polynomial in Eq. 2, which is given by:

$$\boldsymbol{f}(\boldsymbol{z}) = \boldsymbol{\beta} + \sum_{k=1}^{N} \boldsymbol{O}_k \boldsymbol{G}_k \left[\left(\boldsymbol{I}_{kk}^\top \Delta \boldsymbol{z} \right) \otimes \cdots \otimes \left(\boldsymbol{I}_{k1}^\top \Delta \boldsymbol{z} \right) \right], \tag{12}$$

where $\boldsymbol{\beta} = \boldsymbol{f}(\boldsymbol{z}_0)$, $\boldsymbol{O}_k$, $\boldsymbol{G}_k$, and $\boldsymbol{I}_{kj}^\top$ $(k = 1, \ldots, N;\ j = 1, \ldots, k)$ are learnable parameters. Fig. 1(b) visualizes the proposed TaylorNet with Tucker decomposition. In theory, it is possible to stack multiple layers of TaylorNet to construct high-order polynomials with high expressivity. However, to reduce the number of model parameters and allow straightforward interpretation of the model's prediction, we only use single-layer Taylor network with small orders of the Taylor polynomial (e.g., 2 or 3).

Computational Complexity of TaylorNet. Given d and o as the input and output dimension respectively, the original computational complexity of the k-order term of Taylor polynomial is $\mathcal{O}(od^k)$. Using Tucker decomposition results in a substantial reduction in the number of parameters to $\mathcal{O}\left(r_{out,k} \prod_{j=1}^{k} + o r_{out,k} + d \sum_{j=1}^{k} r_{in,j,k}\right)$. In particular, when the rank of the core tensor in Tucker decomposition is much smaller than d and o, the training time of TaylorNet will be improved by orders of magnitude.

5 Experiments

In this section, we conduct extensive experiments to evaluate the performance of the proposed CAT across multiple tabular and image benchmark datasets. Additionally, we do a case study to demonstrate that our method can effectively explain the prediction results using high-level concepts. Finally, we also demonstrate the effectiveness of the concept encoders applied to other baselines.

5.1 Datasets

We conduct experiments on six real-world benchmarks, including four tabular datasets (1-4) and two image datasets (5-6), as below:

(1) **Airbnb Listings and Reviews** (Airbnb [23]): This dataset encompasses over 250,000 Airbnb listings across major global cities. We formulate a regression task on this dataset, predicting the listing price based on host details, locations, and property information. With 126 features, we manually categorize them into 6 concepts.
(2) **WiDS Diabetes Detection** (Diabetes [22]): This binary classification dataset indicates whether a patient is diagnosed with diabetes within the initial 24 hours of admission. Features include demographic information, medical history, and various lab metrics, totaling 176 features grouped into 6 categories by data providers [22].
(3) **COMPAS Recidivism** (COMPAS [28]): This binary classification dataset aims to predict the risk of repeated offense among convicted criminals. It includes 6 input features: age, sex, race, priors count, charge degree, and custody length. Based on their semantics, we divide these features into 2 groups: demographic (first 3) and criminal history (last 3).
(4) **Daily and Sports UCI-HAR** (UCI-HAR [38]): This multiclass dataset from the UCI ML Repository involves predicting 6 daily activities performed by volunteers over a period of time while wearing a smartphone with multiple sensors on the waist. The signal sequences come from 3 sensor types: body accelerometer, gravity accelerometer, and gyroscope. Bulbul et al. [38] further derived jerk signals from accelerometers, then calculated triaxial signal magnitudes and fast Fourier transform frequencies on them to obtain a total of 18 different signal types. To obtain tabular features from time series data, they extracted 33 features including descriptive statistics, energy, and autocorrelation coefficients from each signal type to gather 561 features in total for the training data. We consider these 18 signal types as high-level concepts in our experiment.

Table 2: Summary of Experimental Datasets

Name	Airbnb	COMPAS	Diabetes	UCI-HAR	MNIST	CelebA
Instances	275,598	6,172	130,157	10,299	70,000	30,000
Features	126	6	176	561	-	-
Concepts	6	2	6	18	6	9
Classes	-	2	2	6	10	2
Feature Type	Mixed	Mixed	Mixed	Numeric	Image	Image

(5) **MNIST** [29]: This dataset consists of 70,000 images of handwritten digits (0-9) for multi-class classification. Following previous works [24, 41], we utilize disentangled representation learning to extract 6 high-level latent factors like style and shape from MNIST images as high-level-concepts for performing classification.
(6) **CelebA** [30]: This dataset contains 30,000 high-quality RBG images of center-aligned facial photographs of celebrities. We formulate a binary classification task on this dataset based on the gender annotations (male/female) provided by Lee et al. [30]. Additionally, we extract 9 high-level concepts such as skin tone, hair azimuth, hair length, etc. from facial images using disentangled representation learning [41].

The summary of these datasets is presented in Table 2. Furthermore, we adopt a fixed random sample ratio of 80-10-10 for training, validation, and testing for Airbnb, COMPAS, and Diabetes. Since the train-test splits are provided in UCI-HAR, MNIST, and CelebA datasets, we reserve 10% of each training dataset for validation.

5.2 Baselines

We compare the performance of CAT with the following baselines for both classification and regression problems. Note that the following two methods, MLP and XGBoost, are uninterpretable while the remaining are interpretable.

- **Multi-layer Perceptron (MLP)**: MLP serves as a standard uninterpretable black-box neural network, setting the upper bound on prediction performance.
- **Gradient Boosted Trees (XGBoost)**: XGBoost [9] is a robust machine learning algorithm based on an ensemble of decision trees. Given the typically large number of trees, the model becomes uninterpretable. We utilize the `xgboost` library in our experiments.
- **Explainable Boosting Machines (EBM)**: EBM models are Generalized Additive Models (GAMs) that leverage millions of shallow bagged trees to learn a shape function for each feature individually [6, 33]. We implement EBM using the `interpretml` library.
- **Neural Additive Models (NAM)**: NAMs [1] extend GAMs with neural network components to learn one Multi-Layer Perceptron (MLP) per feature. We implement the original NAM following prior work [14] to expedite training time.
- **Neural Basis Models (NBM)**: As an extension of NAMs, NBMs learn a set of basis functions shared across all features instead of individual shape functions for each feature [37].
- **Scalable Polynomial Additive Models (SPAM)**: Representing the current state-of-the-art GAMs, SPAMs incorporate high-order interactions between shape functions learned by prior NAMs using polynomial neural networks [14].
- **Grand-Slamin Additive Modeling with Structural Constraints (Grand-Slamin)**: This method utilizes tree-based GAMs with sparsity and structural constraints to selectively learn second-order interactions between shape functions derived from soft decision trees [20] for enhanced interpretability and scalability [19].

Note that we do not compare with concepts-based interpretability methods [10, 25, 51], since these approaches require domain experts to label a lot of concepts and then specify the corresponding ground-truth values. In addition, the above interpretable baselines, such as EBM, NAM, NBM, SPAM, and Grand-Slamin are difficult to interpret if number of features are large, since they treat each feature individually without organizing them into concepts; they are also computationally expensive because they use large trees or DNNs on each feature.

5.3 Experimental Settings

CAT Architecture: We employ the following structure for the concept encoders: a Multi-Layer Perceptron (MLP) with 3 hidden layers having 64, 64, and 32 hidden units, along with LeakyReLU activation [48], following recent approaches in concept-based methods [25, 51]. For the TaylorNet, we opt for rank $r = 8$ for Taylor order 2 and $r = 16$ for Taylor order 3. The initial value of Taylor series expansion is 0. In Section 5.7, we will investigate the impact of these hyperparameters on prediction performance.
Implementation Details. MLP, NAM, NBM, SPAM, Grand-Slamin, and CAT models are implemented in Pytorch and trained using the Adam optimizer with decoupled weight decay regularization (AdamW [32]) on A6000 GPU machines with 48GB memory. We train the model with 100 epochs on all six datasets with early stopping. For CAT on all datasets, we tune the starting learning rate in the interval $[0.0001, 0.1]$, concept encoder dropout and TaylorNet dropout coefficients in the discrete set $\{0, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5\}$. The best-performing hyperparameters are determined using the validation set through grid search. A similar tuning procedure is applied to MLP, NAM, NBM, SPAM, and Grand-Slamin. Finally, for EBMs and XGBoost, we use CPU machines and follow the training guidelines provided by corresponding code libraries.
Evaluation Details. We report the appropriate performance metrics for different prediction tasks: (i) Root Mean-Squared Error (RMSE) for regression task; and (ii) Accuracy and Macro-F1 for classification tasks. Moreover, we report the average performance over 3 runs with different random seeds.

5.4 Main Results

In this subsection, we assess the performance of CAT using six benchmarks, considering a variety of tasks from regression to multi-class classification, and different types of data such as tabular data and images. We compare our method with the baselines above. Table 3 shows the comparative results of different black-box and interpretable models averaged over three random seeds. Overall, both of the proposed CAT with order 2 and 3 comfortably outperform most of their interpretable counterparts and are comparable to some black-box DNNs on all six benchmarks. Since CAT and SPAM both utilize polynomial functions to make predictions, their performance is very close to other each. However, SPAM requires more

Table 3: Performance comparison between CAT and prior ML methods on benchmark datasets averaged over three random seeds. Note that the best result is highlighted in bold black and the second best is highlighted in green. We can observe that CATs generally outperform NAM, NBM, SPAMs, and Grand-Slamin, and outperform EBM in five out of six benchmarks.

Models	AirBnb	COMPAS		Diabetes		UCI-HAR		MNIST		CelebA	
	RMSE↓	Acc↑	Macro-F1↑	Acc↑	Macro-F1↑	Acc↑	Macro-F1↑	Acc↑	Macro-F1↑	Acc↑	Macro-F1↑
Black-box Baselines											
XGBoost	0.5131	0.6713	0.6664	0.8258	0.7129	0.9796	0.9381	0.9903	0.9510	0.7734	0.7528
MLP	0.5437	0.6599	0.6468	0.8257	0.7232	0.9840	0.9514	0.9902	0.9510	0.7768	0.7601
Interpretable Models											
EBM	0.6344	0.6710	0.6643	0.8269	0.7031	**0.9867**	**0.9593**	0.9808	0.9030	0.7413	0.7061
NAM	0.6681	0.6699	0.6623	0.8242	0.7199	0.9785	0.9346	0.9723	0.8635	0.7441	0.7226
NBM	0.6637	0.6742	0.6708	0.8257	0.7167	0.9792	0.9367	0.9770	0.8878	0.7455	0.7231
SPAM (Order 2)	0.5664	0.6659	0.6569	0.8230	0.7242	0.9809	0.9414	0.9860	0.9318	0.7468	0.7129
SPAM (Order 3)	0.5560	0.6688	0.6608	0.8272	0.7188	0.9801	0.9388	0.9883	0.9426	**0.7642**	0.7385
Grand-Slamin	0.5811	0.6704	0.6630	0.8266	0.7260	0.9800	0.9392	0.9864	0.9317	0.7537	0.7241
CAT (Order 2)	**0.5486**	**0.6772**	**0.6710**	**0.8286**	**0.7269**	0.9814	0.9431	**0.9892**	**0.9469**	0.7609	**0.7436**
CAT (Order 3)	**0.5461**	**0.6793**	**0.6726**	**0.8295**	**0.7270**	**0.9829**	**0.9480**	**0.9902**	**0.9517**	**0.7728**	**0.7579**

Table 4: Benchmarks on the number of parameters and training throughput (training examples per second) between interpretable ML methods on six datasets. Here the best result is highlighted in bold black and the second best is highlighted in green. We omit the training throughput of EBMs since they are trained on CPU machines. Overall, CATs have lower number of parameters and higher training throughput than NAM, NBM, and SPAM. Despite having the most compact models for Airbnb and Diabetes, the number of parameters in EBM drastically grow when applied on larger datasets.

Models	Number of Parameters↓						Training Throughout (samples/sec)↑					
	Airbnb	COMPAS	Diabetes	UCI-HAR	MNIST	CelebA	Airbnb	COMPAS	Diabetes	UCI-HAR	MNIST	CelebA
EBM	**28,110**	**9,827**	351,165	12,661,020	86,800	112,640	_	_	_	_	_	_
NAM	822,780	40,326	1,142,750	2,398,500	40,326	67,210	9,553	4,929	6,793	1,667	46,684	19,339
NBM	76,897	64,664	82,071	**124,077**	64,720	65,076	24,792	5,059	17,308	3,917	51,247	21,244
SPAM (Order 2)	1,719,219	82,254	2,338,102	7,657,734	83,862	136,822	4,927	4,641	3,428	949	33,344	17,324
SPAM (Order 3)	2,604,292	124,982	3,617,729	11,601,687	128,998	207,634	3,259	4,544	2,271	646	28,608	15,090
Grand-Slamin	494,173	**4,852**	952,176	2,911,253	16,984	13,044	2,091	4,725	1,316	909	48,157	20,051
CAT (Order 2)	**48,742**	14,354	**51,310**	**161,138**	**880**	**4,896**	**90,333**	**5,415**	**59,958**	**7,598**	**67,097**	**25,773**
CAT (Order 3)	52,990	18,514	**51,646**	227,634	**5,200**	**11,760**	**62,672**	**5,273**	**57,519**	**7,336**	**62,981**	**24,864**

model parameters in Tab. 4, since it leverages low-level features to make predictions while our CAT uses high-level concepts.

Below, we further examine the prediction performance of CAT models on each benchmark dataset. Firstly, our proposed method outperforms all interpretable baselines for the regression task on Airbnb dataset. The reason why CAT models works well for regression is that they aproximate real-valued target variables using Taylor polynomials, which can capture high-order interactions between different inputs. Particularly, CAT models, and SPAM which also use polynomials, have significantly lower prediction error (RMSE) than the other interpretable baselines that only utilize linear models. Also, CAT can compete closely with fully-connected MLPs, whose non-interpretable architecture implicitly models every order of feature interaction. Next, regarding binary classification of diabetes and recidivism, one notable observation is that using second-order concept interactions is sufficient for CAT to outperform all baselines including black-box models. One possible explanation is that our method can aggregates important patterns into its high-level concept representations, while other models can be negatively impacted by less useful low-level features. Additionally, on the multi-class classification dataset UCI-HAR, both CAT models outperform all interpretable baselines except for EBM. Although EBM performs better than CAT models and all other baselines on UCI-HAR, it is the least scalable interpretable method for this multi-class classification problem which we will further explore in the following experiment. Lastly, the proposed CAT is superior to interpretable baseline methods on both image datasets MNIST and CelebA, demonstrating its efficacy for visual reasoning.

Furthermore, we benchmark the number of parameters and training throughput for all interpretable models on six datasets, which is summarized in Tab. 4. Here, we report the training throughput (measured in samples iterated per second) for methods that were trained on GPU machines over a fixed number of training iterations. Compared to other DNN-based interpretable methods, our CAT models generally have the smallest number of parameters and the highest training throughput. The main reason behind it is our adoption of high-level concept encoders to compress the input features, and the lightweight TaylorNet with Tucker Decomposition. Since NAMs use one DNN to encode each feature separately, they require a large number of parameters and take significantly more time to train. SPAMs also suffer from this problem as they adopt NAM to learn feature representations. Although Grand-Slamin utilizes sparse pairwise interactions to reduce the number of parameters compared to SPAMs, its training speed is not improved on large

datasets such as Airbnb and Diabetes due to the inefficiency of soft decision trees [20]. In addition, despite requiring more parameters than NBM for the UCI-HAR dataset, CAT models can be trained much more efficiently because our DNN concept encoders can be computed in parallel [14], while NBM uses one large DNN to learn many basis representations from the input features [37]. Even if EBMs are the most compact models for Airbnb and COMPAS datasets, one problem with EBM is the drastic increase in the number of parameters when they are applied on data with higher output dimension, whereas other methods including ours only seem to be affected by the input dimension. Specifically, going from Airbnb to Diabetes dataset, as the number of input and output dimensions increase from 126 to 176 and 1 to 2 respectively (Tab. 2), the number of parameters in EBM grows more than 12 times. Also, despite outperforming CAT and all other baselines on UCI-HAR, EBM requires the largest model of over 12 million parameters. Therefore, it is extremely difficult to scale EBM to multi-class classification datasets with hundreds of features in real-world settings [14].

5.5 A Case Study of Interpretability

In this subsection, we present a case study to interpret the listing price prediction on the Airbnb dataset. Firstly, we provide a summary of the grouping of closely related features on Airbnb in Tab. 5. Upon observing this table, it becomes apparent that even in the absence of concept annotation, if the features in the dataset are appropriately named and described in the metadata, general users can easily group them into human-understandable high-level concepts. It's important to note that our concept encoder differs from previous methods in concept-based learning [10, 25, 51], where concepts are required to have not only meaningful names, but also accurate ground-truth values, for example 0/1 for *bad/good Location*. In contrast, our concept encoder does not rely on precise ground-truth values. These concepts can then be effectively utilized by a white-box model for making predictions. During interpretation, the predictions can be traced back to these high-level concepts, enabling users to explain the model's decision at an abstract level.

Table 5: Summary of high-level concepts extracted using feature names and provided metadata on the Airbnb dataset. This table demonstrates that the manual grouping of features into concepts is straightforward on a general application dataset with meaningful feature names and metadata.

High-level Concept	Num. of Features	Low-level Features
Amenities (z_1)	100	essentials, wifi, kitchen, tv, heating, washer, etc.
Host (z_2)	7	host tenure, host location, response time, response rate, etc.
Listing (z_3)	3	minimum nights, maximum nights, instant bookable
Reviews (z_4)	7	overall, accuracy , clealiness, communication, value
Property (z_5)	4	property type, room type, guests accomodated, bedrooms
Location (z_6)	5	neighborhood, district, city, longitude, latitude

The ease of interpretation for CAT models results from the combination of the compact high-level concept inputs and TaylorNet's inherrent interpretability. After learning the high-level concept representations by the concept encoders, we feed them into the white-box TaylorNet to model the non-linear functions with polynomials. For the Airbnb dataset, the discovered Taylor polynomial of order 2 that estimates the listing price is given by:

$$\begin{aligned} f(\boldsymbol{z}) =& 0.02z_1^2 - 0.82z_1z_2 - 0.8z_1z_3 + 1.39z_1z_4 + 1.46z_1z_5 + 2.15z_1z_6 + 0.69z_1 \\ & - 0.1z_2^2 + 0.48z_2z_3 - 1.01z_2z_4 - 0.33z_2z_5 - 1.08z_2z_6 - 0.46z_2 \\ & + 0.43z_3^2 - 1.0z_3z_4 - 0.75z_3z_5 - 1.2z_3z_6 - 0.44z_3 \\ & + 0.09z_4^2 + 0.96z_4z_5 + 1.07z_4z_6 + 0.45z_4 \\ & + 0.08z_5^2 + 2.28z_5z_6 + 0.96z_5 \\ & - 0.46z_6^2 + 1.73z_6 - 0.03, \end{aligned} \tag{13}$$

where z_m for $m = 1, \ldots, 6$ is defined in Tab. 5. The Taylor polynomial enables us to examine the contributions of concepts and their higher-order interactions using standardized regression coefficients [4]. These coefficients, akin to those in linear models, are computed by multiplying each polynomial coefficient by the standard deviation of its corresponding input feature and dividing by the standard deviation of the regression targets. Accordingly, we demonstrate the contributions of six high-level concepts and their second-order interactions to the listing price on the Airbnb dataset in Fig. 2. From this figure, we observe that *Location* and *Property* description have the most significant influence on the price. Additionally, the second-order interactions between *Property*×*Location* and *Amenities* × *Location* are significant factors. This highlights that CAT's decision process mirrors human reasoning at a high level, as human individuals often attribute rental price to location, property quality, and amenities.

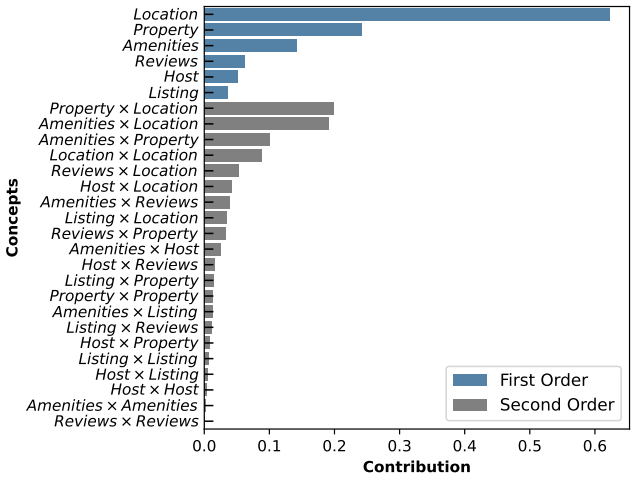


Figure 2: Concept contributions using second-order CAT model for predicting listing price in the Airbnb dataset. Contributions are given by the standardized regression coefficients of the Taylor polynomial. We observe that the *Location* and *Property* descriptions influence the listing price the most.

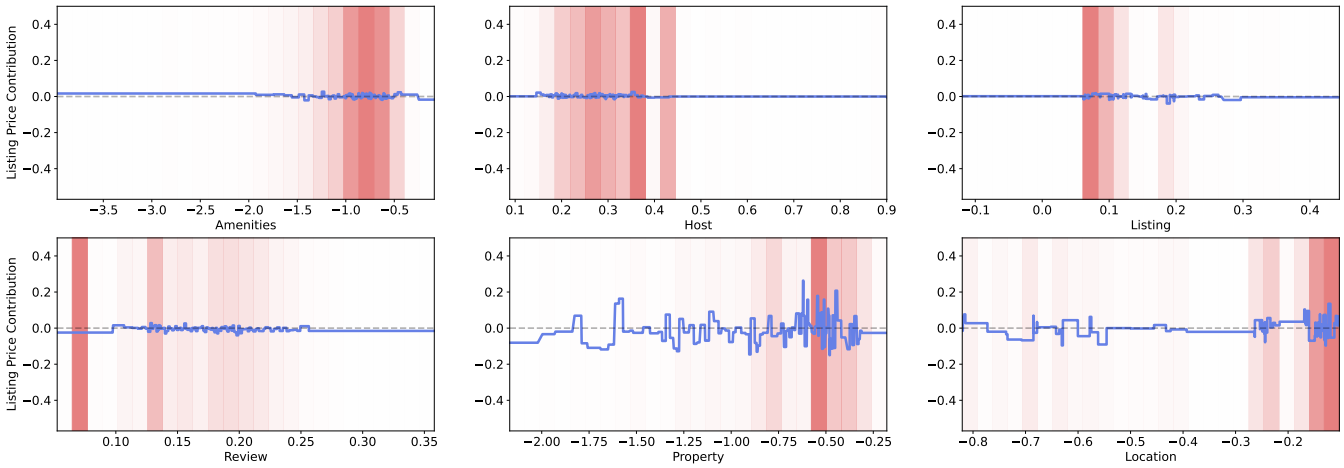


Figure 3: Shape functions for the first-order concepts learned by the second-order CAT model on the Airbnb dataset. The x-axis represents the values of the concepts, while the y-axis indicates the contributions of each value to the listing price. The blue line represents the shape function for a concept. Pink bars represent the normalized data density for 25 bins of concept values.

Moreover, since CAT constructs explanations from a small number of concepts, the explanations are notably shorter than those from methods with feature-based interpretability, particularly those with high-order feature interactions like SPAM and EBM. Specifically, our second-order CAT explains the Airbnb listing price with 27 concepts and interactions, succinctly visualized in Fig. 2, whereas second-order SPAM would necessitate 8127 terms.

Last but not least, for each concept learned by the second-order CAT model on the Airbnb dataset, we visualize its shape function and the corresponding normalized data density on the same graph, as illustrated in Fig. 3. In particular, the shape functions, indicated by the semi-transparent blue lines, elucidate how values of certain concepts, such as *Location* and *Property*, influence the listing price. For example, from Fig. 3, values of the *Location* concept within the range $[-0.58, -0.59]$ have a positive impact on the listing price. Consequently, users can discern the original features related to these concepts, enabling more detailed explanations.

In addition to this case study, we provide visualizations to interpret the gender prediction results on the image dataset CelebA using the second-order CAT model in Appendix A.1.

5.6 Effectiveness of Concept Encoders

To further demonstrate the effectiveness of our concept encoders, we integrate this component into two representative interpretable models: NAM and SPAM. Specifically, we extend NAM to NAM+ by incorporating the concept encoders to replace separate neural networks and subsequently feeding the concept representations into a linear model. Additionally, we develop SPAM+ as an extension of SPAM, wherein we substitute NAM with the concept encoders and then utilize the learned concept representations in a polynomial neural network. As illustrated in Table 6, both NAM+ and SPAM+ demonstrate superior performance when compared to NAM and SPAM, thereby emphasizing the effectiveness of our concept encoders. We also explore the impact of the concept encoders on the prediction performance and computational cost of the proposed CAT model in Appendix A.2.

Table 6: Effectiveness of the concept encoders. Here boldface denotes where NAM+ and SPAM+ are better than their counterparts with feature-wise DNNs. The results averaged over three random seeds illustrate that incorporating our concept encoders improves the performance of the original models.

Models	AirBnb	Diabetes	UCI-HAR
	RMSE↓	Acc↑	Acc↑
NAM	0.6681	0.8242	0.9785
NAM+	**0.6069**	**0.8250**	**0.9795**
SPAM (Order 2)	0.5664	0.8230	0.9809
SPAM+ (Order 2)	**0.5515**	**0.8264**	**0.9812**
SPAM (Order 3)	0.5560	0.8272	0.9801
SPAM+ (Order 3)	**0.5504**	**0.8274**	**0.9824**

5.7 Hyper-parameters Tuning

We also investigate some important hyperparameters in TaylorNet, such as polynomial orders and ranks, on prediction performance, as detailed in Appendix A.3.

6 Conclusion

In this paper, we introduced CAT, a novel interpretable machine learning model capable of explaining and understanding predictions through human-understandable concepts. Unlike prior work, our method did not heavily rely on a large number of labeled concepts by domain experts. Specifically, the proposed CAT consisted of concept encoders that learned high-level concept representations from each group of input features, and a white-box TaylorNet that could approximate the non-linear mapping function between the input and output with polynomials. Additionally, Tucker decomposition was employed to reduce the computational complexity of TaylorNet. Extensive experimental results demonstrated that CAT not only achieved competitive or outstanding performance but also significantly reduced the number of model parameters. Importantly, it was able to explain model predictions using high-level concepts that humans can easily understand.

References

[1] Rishabh Agarwal, Levi Melnick, Nicholas Frosst, Xuezhou Zhang, Ben Lengerich, Rich Caruana, and Geoffrey E Hinton. 2021. Neural additive models: Interpretable machine learning with neural nets. *Advances in neural information processing systems* 34 (2021), 4699–4711.

[2] I APACHE. 1985. APACHE II: A severity of disease classification system. (1985).

[3] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. 2020. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information fusion* 58 (2020), 82–115.

[4] Johan Bring. 1994. How to standardize regression coefficients. *The American Statistician* 48, 3 (1994), 209–213.

[5] Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. 2019. Semantic bottleneck for computer vision tasks. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part II 14*. Springer, 695–712.

[6] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. 1721–1730.

[7] Diogo V Carvalho, Eduardo M Pereira, and Jaime S Cardoso. 2019. Machine learning interpretability: A survey on methods and metrics. *Electronics* 8, 8 (2019), 832.

[8] Chun-Hao Chang, Rich Caruana, and Anna Goldenberg. 2021. Node-gam: Neural generalized additive model for interpretable deep learning. *arXiv preprint arXiv:2106.01613* (2021).

[9] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. 785–794.

[10] Zhi Chen, Yijie Bei, and Cynthia Rudin. 2020. Concept whitening for interpretable image recognition. *Nature Machine Intelligence* 2, 12 (2020), 772–782.

[11] Grigorios G Chrysos, Stylianos Moschoglou, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, and Stefanos Zafeiriou. 2020. P-nets: Deep polynomial neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7325–7335.

[12] James R Clough, Ilkay Oksuz, Esther Puyol-Antón, Bram Ruijsink, Andrew P King, and Julia A Schnabel. 2019. Global and local interpretability for cardiac MRI classification. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 656–664.

[13] Jeffrey De Fauw, Joseph R Ledsam, Bernardino Romera-Paredes, Stanislav Nikolov, Nenad Tomasev, Sam Blackwell, Harry Askham, Xavier Glorot, Brendan O'Donoghue, Daniel Visentin, et al. 2018. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine* 24, 9 (2018), 1342–1350.

[14] Abhimanyu Dubey, Filip Radenovic, and Dhruv Mahajan. 2022. Scalable interpretability via polynomials. *Advances in neural information processing systems* 35 (2022), 36748–36761.

[15] Amirata Ghorbani, Abubakar Abid, and James Zou. 2019. Interpretation of neural networks is fragile. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 3681–3688.

[16] Mara Graziani, Vincent Andrearczyk, and Henning Müller. 2018. Regression concept vectors for bidirectional explanations in histopathology. In *Understanding and Interpreting Machine Learning in Medical Image Computing Applications: First International Workshops, MLCN 2018, DLF 2018, and iMIMIC 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16-20, 2018, Proceedings 1*. Springer, 124–132.

[17] Trevor J Hastie. 2017. Generalized additive models. In *Statistical models in S*. Routledge, 249–307.

[18] Marton Havasi, Sonali Parbhoo, and Finale Doshi-Velez. 2022. Addressing leakage in concept bottleneck models. *Advances in Neural Information Processing Systems* 35 (2022), 23386–23397.

[19] Shibal Ibrahim, Gabriel Afriat, Kayhan Behdin, and Rahul Mazumder. 2023. GRAND-SLAMIN'Interpretable Additive Modeling with Structural Constraints. *Advances in Neural Information Processing Systems* 36 (2023).

[20] Ozan Irsoy, Olcay Taner Yıldız, and Ethem Alpaydın. 2012. Soft decision trees. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*. IEEE, 1819–1822.

[21] Maksims Ivanovs, Roberts Kadikis, and Kaspars Ozols. 2021. Perturbation-based methods for explaining deep neural networks: A survey. *Pattern Recognition Letters* 150 (2021), 228–234.

[22] Kaggle. 2021. WiDS Datathon 2021: Diabetes Detection. https://www.kaggle.com/code/iamleonie/wids-datathon-2021-diabetes-detection. Accessed: 2023-12-20.

[23] Kaggle. 2022. Airbnb Listing & Reviews. https://www.kaggle.com/datasets/mysarahmadbhat/airbnb-listings-reviews. Accessed: 2023-12-20.

[24] Hyunjik Kim and Andriy Mnih. 2018. Disentangling by factorising. In *International conference on machine learning*. PMLR, 2649–2658.

[25] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. 2020. Concept bottleneck models. In *International conference on machine learning*. PMLR, 5338–5348.

[26] Tamara G Kolda and Brett W Bader. 2009. Tensor decompositions and applications. *SIAM review* 51, 3 (2009), 455–500.

[27] Cihan Kuzudisli, Burcu Bakir-Gungor, Nurten Bulut, Bahjat Qaqish, and Malik Yousef. 2023. Review of feature selection approaches based on grouping of features. *PeerJ* 11 (2023), e15666.

[28] Jeff Larson, Surya Mattu, Lauren Kirchner, and Julia Angwin. 2016. ProPublica Compas Analysis—Data and Analysis for 'Machine Bias.'. *https://github.com/propublica/compas-analysis* (2016).

[29] Yann LeCun. 1998. The MNIST database of handwritten digits. *http://yann. lecun. com/exdb/mnist/* (1998).

[30] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. 2020. Maskgan: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5549–5558.

[31] Max Losch, Mario Fritz, and Bernt Schiele. 2019. Interpretability beyond classification output: Semantic bottleneck networks. *arXiv preprint arXiv:1907.10882* (2019).

[32] Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101* (2017).

[33] Yin Lou, Rich Caruana, and Johannes Gehrke. 2012. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 150–158.

[34] Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems* 30 (2017).

[35] Andreas Madsen, Siva Reddy, and Sarath Chandar. 2022. Post-hoc interpretability for neural nlp: A survey. *Comput. Surveys* 55, 8 (2022), 1–42.

[36] Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. 2021. Promises and pitfalls of black-box concept learning models. *arXiv preprint arXiv:2106.13314* (2021).

[37] Filip Radenovic, Abhimanyu Dubey, and Dhruv Mahajan. 2022. Neural basis models for interpretability. *Advances in Neural Information Processing Systems* 35 (2022), 8414–8426.

[38] Jorge Reyes-Ortiz, Davide Anguita, Alessandro Ghio, Luca Oneto, and Xavier Parra. 2012. Human Activity Recognition Using Smartphones. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C54S4K.

[39] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Model-agnostic interpretability of machine learning. *arXiv preprint arXiv:1606.05386* (2016).

[40] Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature machine intelligence* 1, 5 (2019), 206–215.

[41] Huajie Shao, Yifei Yang, Haohong Lin, Longzhong Lin, Yizhuo Chen, Qinmin Yang, and Han Zhao. 2022. Rethinking Controllable Variational Autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 19250–19259.

[42] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. 2020. Controlvae: Controllable variational autoencoder. In *International Conference on Machine Learning*. PMLR, 8655–8664.

[43] Dylan Slack, Anna Hilgard, Sameer Singh, and Himabindu Lakkaraju. 2021. Reliable post hoc explanations: Modeling uncertainty in explainability. *Advances in neural information processing systems* 34 (2021), 9391–9404.

[44] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. 2020. Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*. 180–186.

[45] Marshall H Stone. 1948. The generalized Weierstrass approximation theorem. *Mathematics Magazine* 21, 5 (1948), 237–254.

[46] George Brinton Thomas, Maurice D Weir, Joel Hass, Frank R Giordano, and Recep Korkmaz. 2010. *Thomas' calculus*. Vol. 12. Pearson Boston.

[47] Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31, 3 (1966), 279–311.

[48] Jin Xu, Zishan Li, Bowen Du, Miaomiao Zhang, and Jing Liu. 2020. Reluplex made more practical: Leaky ReLU. In *2020 IEEE Symposium on Computers and communications (ISCC)*. IEEE, 1–7.

[49] Liuyi Yao, Yaliang Li, Sheng Li, Jinduo Liu, Mengdi Huai, Aidong Zhang, and Jing Gao. 2022. Concept-level model interpretation from the causal aspect. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[50] Kexin Yi, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum. 2018. Neural-symbolic vqa: Disentangling reasoning from vision and language understanding. *Advances in neural information processing systems* 31 (2018).

[51] Mateo Espinosa Zarlenga, Pietro Barbiero, Gabriele Ciravegna, Giuseppe Marra, Francesco Giannini, Michelangelo Diligenti, Zohreh Shams, Frederic Precioso, Stefano Melacci, Adrian Weller, et al. 2022. Concept embedding models. *arXiv preprint arXiv:2209.09056* (2022).

A Appendix

A.1 Additional Interpretability Results on the CelebA Dataset

Besides the case study presented in Subsection 5.5, we provide additional visualizations to interpret the gender prediction results from the second-order CAT model on the image dataset CelebA. First, we showcase the gender prediction contributions of 9 high-level concepts acquired from disentangled representation learning, and the 16 most salient second-order interactions among these concepts in Fig. 4. From this figure, it is evident that certain concepts such as *SkinTone*, *HairAzimuth*, and *HairLength* exhibit the most influence on the model's gender predictions, aligning closely with human reasoning at a high level. Furthermore, we include Fig. 5 to delineate the shape functions of 9 first-order concepts, providing clarity on how variations in specific concept values, such as *SkinTone* and *HairLength*, impact predictions of female gender. For instance, individuals with darker *SkinTone* are more likely to be classified as male, whereas those with longer *HairLength* tend to be classified as female. Consequently, by employing the outlined concept-based explanation method, users can interpret the model's decision-making process with relative ease.

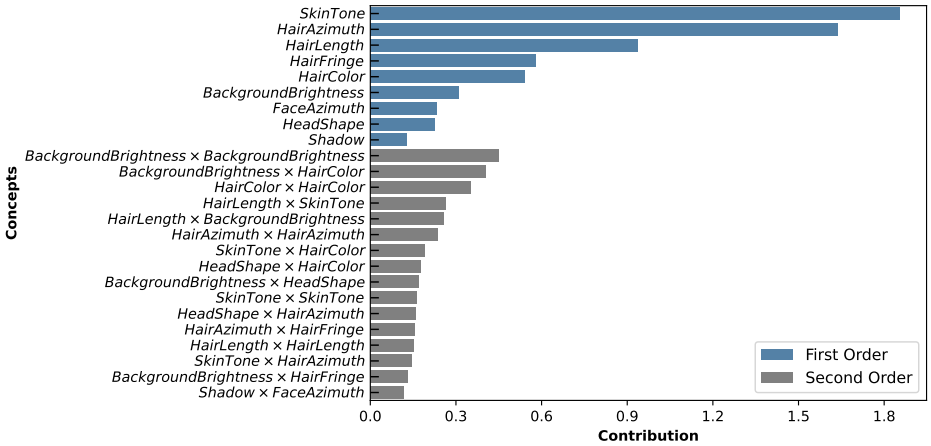


Figure 4: Concept contributions using second-order Taylor for predicting gender in the CelebA dataset. Contributions are given by the standardized regression coefficients of the Taylor polynomial. We observe that the *SkinTone*, *HairAzimuth*, and *HairLength* concepts influence the gender prediction the most.

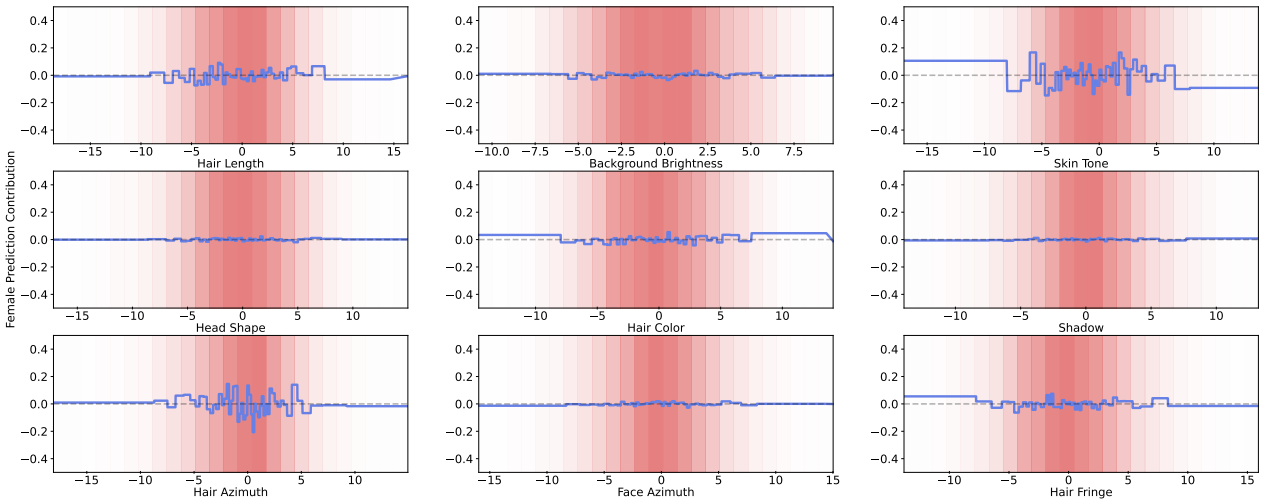


Figure 5: Shape functions for the first-order concepts learned by the second-order CAT model on the CelebA dataset. The x-axis represents the values of the concepts, while the y-axis indicates the contributions of each value to the prediction of a female person. The blue line represents the shape function for a concept. Pink bars represent the normalized data density for 25 bins of concept values.

A.2 Ablation of Concept Encoders in the Proposed CAT

To further assess the efficacy of the concept encoders (CE), we conduct an ablation study by systematically excluding them from the proposed CAT models (i.e., the input features are fed directly into TaylorNet), and measure the resulted prediction performance and computation

Table 7: Effectiveness of the concept encoders in the proposed CAT using three benchmark datasets. Here boldface denotes where the CAT models are better than their counterparts without concept encoders (CE). The results averaged over three random seeds demonstrate that the accuracy of CAT drops while the number of parameters increases without concept encoding.

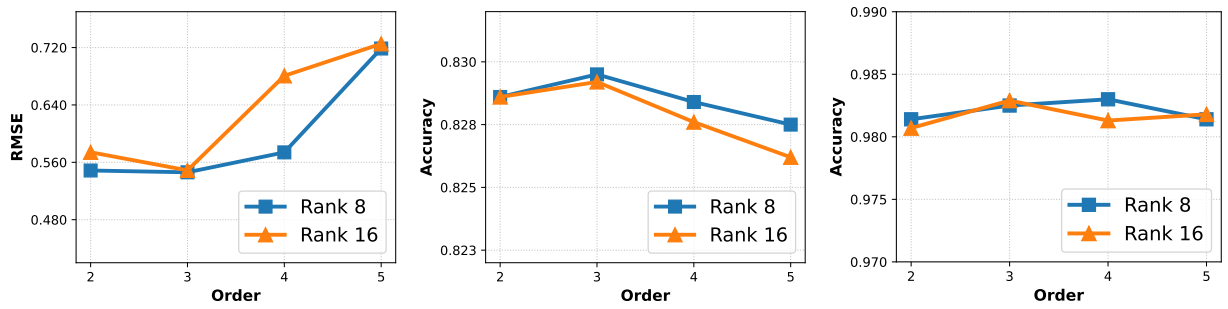
Models	Airbnb		Diabetes		UCI-HAR	
	RMSE↓	Num. of Params↓	Acc↑	Num. of Params↓	Acc↑	Num. of Params ↓
CAT (Order 2)	**0.5486**	**48,742**	**0.8286**	**51,310**	**0.9814**	**161,138**
CAT w/o CE (Order 2)	0.5981	131,136	0.8241	138,288	0.9723	194,256
CAT (Order 3)	**0.5461**	**52,990**	**0.8295**	**51,646**	**0.9829**	**227,634**
CAT w/o CE (Order 3)	0.5706	860,670	0.8114	869,580	0.9728	1,190,656

cost using three datasets: Airbnb, Diabetes, and UCI-HAR. The results are presented in Table 7. From this table, we can draw a conclusion that without the concept encoders, the accuracy of CAT drops while the number of parameters increases drastically. This underscores the necessity of incorporating them into our model.

A.3 Hyper-parameter Tuning

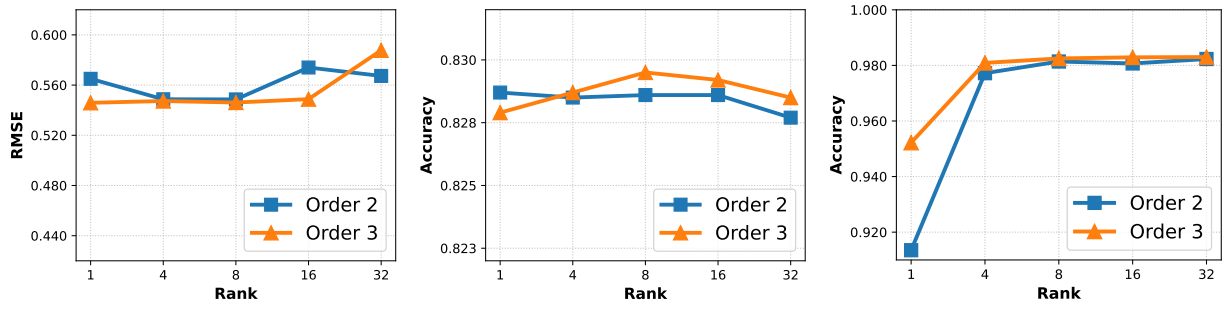
Effect of polynomial order. We explore the impact of order N in TaylorNet on model performance. Although it is natural to expect the quality of approximation to improve as the order of polynomials increases, the input and output dimensions are important factors for choosing an appropriate combination of polynomial orders and Tucker decomposition ranks to obtain satisfactory prediction performance. We can observe from Fig. 6 that using higher-order Taylor polynomials (e.g. ≥ 3) is beneficial on a dataset with higher input and output dimensions like UCI-HAR (Fig. 6c), where the input space is approximately 10 times larger than the other two datasets.

Effect of decomposition rank. We also study the effect of the rank of Tucker decomposition on prediction performance. Additionally, choosing a higher rank r for Tucker decomposition on smaller weight tensors on Airbnb and Diabetes datasets leads to diminishing returns. As illustrated in Fig. 7, we can see the model performance will be gradually improved as the rank r increases from 1 to 8, and then it will remain almost unchanged or degraded due to overfitting as the rank rises from 8 to 32.



(a) Airbnb. (b) Diabetes. (c) UCI-HAR.

Figure 6: Effect of the order of TaylorNet on predictions using three benchmark datasets. Regarding the Airbnb dataset, lower RMSE scores indicate better performance. For Diabetes and UCI-HAR, we compare the prediction accuracy of the target categories. We can see that as the order of polynomials increases, the prediction performance on small datasets will drop due to overfitting.



(a) Airbnb. (b) Diabetes. (c) UCI-HAR.

Figure 7: Effect of the rank of Tucker decomposition on three benchmark datasets. Regarding the Airbnb dataset, lower RMSE scores indicate better performance. For Diabetes and UCI-HAR, we compare the prediction accuracy of the target categories. We can see that increasing the rank improves performance on UCI-HAR with large input dimension, but it will negatively impact performance on small datasets due to overfitting.