Detecting Beneficial Feature Interactions for Recommender Systems

Yixin Su, [1] **Rui Zhang,** [2] **Sarah Erfani,** [1] **Zhenghua Xu** [3]
[1] University of Melbourne
[2] Tsinghua University
[3] Hebei University of Technology
yixins1@student.unimelb.edu.au, rui.zhang@ieee.org, sarah.erfani@unimelb.edu.au, zhenghua.xu@hebut.edu.cn

Abstract

Feature interactions are essential for achieving high accuracy in recommender systems. Many studies take into account the interaction between every pair of features. However, this is suboptimal because some feature interactions may not be that relevant to the recommendation result, and taking them into account may introduce noise and decrease recommendation accuracy. To make the best out of feature interactions, we propose a graph neural network approach to effectively model them, together with a novel technique to automatically detect those feature interactions that are beneficial in terms of recommendation accuracy. The automatic feature interaction detection is achieved via edge prediction with an L_0 activation regularization. Our proposed model is proved to be effective through the information bottleneck principle and statistical interaction theory. Experimental results show that our model (i) outperforms existing baselines in terms of accuracy, and (ii) automatically identifies beneficial feature interactions.

Introduction

Recommender systems play a central role in addressing information overload issues in many Web applications, such as e-commerce, social media platforms, and lifestyle apps. The core of recommender systems is to predict how likely a user will interact with an item (e.g., purchase, click). An important technique is to discover the effects of features (e,g., contexts, user/item attributes) on the target prediction outcomes for fine-grained analysis (Shi, Larson, and Hanjalic 2014). Some features are correlated to each other, and the joint effects of these correlated features (i.e., *feature interactions*) are crucial for recommender systems to get high accuracy (Blondel et al. 2016; He and Chua 2017). For example, it is reasonable to recommend a user to use *Uber* on a *rainy* day at off-work hours (e.g., during *5-6pm*). In this situation, considering the feature interaction $<$ *5-6pm*, *rainy* $>$ is more effective than considering the two features separately. Therefore, in recent years, many research efforts have been put in modeling the feature interactions (He and Chua 2017; Lian et al. 2018; Song et al. 2019). These models take into account the interaction between every pair of features. However, in practice, not all feature interactions are relevant to the recommendation result (Langley et al. 1994; Siegmund et al. 2012). Modeling the feature interactions that provide little useful information may introduce noise and cause overfitting, and hence decrease the prediction accuracy (Zhang et al. 2017; Louizos, Welling, and Kingma 2018). For example, a user may use *Gmail* on a workday no matter what weather it is. However, if the interaction of workday and specific weather is taking into account in the model and due to the bias in the training set (e.g., the weather of the days when the Gmail usage data are collected happens to be cloudy), the interaction $<$ *Monday*, *Cloudy* $>$ might be picked up by the model and make less accurate recommendations on Gmail usage. Some work considers each feature interaction's importance through the attention mechanism (Xiao et al. 2017; Song et al. 2019). However, these methods still take all feature interactions into account. Moreover, they capture each *individual* feature interaction's contribution to the recommendation prediction, failing to capture the *holistic* contribution of a set of feature interactions together.

In this paper, we focus on identifying the set of feature interactions that together produce the best recommendation performance. To formulate this idea, we propose the novel problem of *detecting beneficial feature interactions*, which is defined as identifying the set of feature interactions that contribute most to the recommendation accuracy (see Definition 1 for details). Then, we propose a novel graph neural network (GNN)-based recommendation model, L_0-SIGN, that detects the most beneficial feature interactions and utilizes only the beneficial feature interactions for recommendation, where each data sample is treated as a graph, features as nodes and feature interactions as edges. Specifically, our model consists of two components. One component is an L_0 edge prediction model, which detects the most beneficial feature interactions by predicting the existence of edges between nodes. To ensure the success of the detection, an L_0 activation regularization is proposed to encourage unbeneficial edges (i.e. feature interactions) to have the value of 0, which means that edge does not exist. Another component is a graph classification model, called **S**tatistical **I**nteraction **G**raph neural **N**etwork (SIGN). SIGN takes nodes (i.e., features) and detected edges (i.e., beneficial feature interactions) as the input graph, and outputs recommendation pre-

*Rui Zhang and Zhenghua Xu are corresponding authors.

dictions by effectively modeling and aggregating the node pairs that are linked by an edge.

Theoretical analyses are further conducted to verify the effectiveness of our model. First, the most beneficial feature interactions are guaranteed to be detected in L_0-SIGN. This is proved by showing that the empirical risk minimization procedure of L_0-SIGN is a variational approximation of the Information Bottleneck (IB) principle, which is a golden criterion to find the most relevant information correlating to target outcomes from inputs (Tishby, Pereira, and Bialek 2000). Specifically, only the most beneficial feature interactions will be retained in L_0-SIGN. It is because, in the training stage, our model simultaneously minimizes the number of detected feature interactions by the L_0 activation regularization, and maximizes the recommendation accuracy with the detected feature interactions. Second, we further show that the modeling of the detected feature interactions in SIGN is very effective. By accurately leveraging the relational reasoning ability of GNN, *iff* a feature interaction is detected to be beneficial in the L_0 edge prediction component, it will be modeled in SIGN as a statistical interaction (an interaction is called statistical interaction if the joint effects of variables are modeled correctly).

We summarize our contributions as follows:

- This is the first work to formulate the concept of beneficial feature interactions, and propose the problem of detecting beneficial feature interactions for recommender systems.
- We propose a model, named L_0-SIGN, to detect the beneficial feature interactions via a graph neural network approach and L_0 regularization.
- We theoretically prove the effectiveness of L_0-SIGN through the information bottleneck principle and statistical interaction theory.
- We have conducted extensive experimental studies. The results show that (i) L_0-SIGN outperforms existing baselines in terms of accuracy; (ii) L_0-SIGN effectively identifies beneficial feature interactions, which result in the superior prediction accuracy of our model.

Related Work

Feature Interaction based Recommender Systems Recommender systems are one of the most critical research domains in machine learning (Lu et al. 2015; Wang et al. 2019). Factorization machine (FM) (Rendle 2010) is one of the most popular algorithms in considering feature interactions. However, FM and its deep learning-based extensions (Xiao et al. 2017; He and Chua 2017; Guo et al. 2017) consider all possible feature interactions, while our model detects and models only the most beneficial feature interactions. Recent work considers the importance of feature interactions by giving each feature interaction an attention value (Xiao et al. 2017; Song et al. 2019), or select important interactions by using gates (Liu et al. 2020) or by searching in a tree-structured space (Luo et al. 2019). In these methods, the importance is not determined by the holistic contribution of the feature interactions, thus limit the performance. Our model detects beneficial feature interactions that together produce the best recommendation performance.

Graph Neural Networks (GNNs) GNNs can facilitate learning entities and their holistic relations (Battaglia et al. 2018). Existing work leverages GNNs to perform relational reasoning in various domains. For example, Duvenaud et al. (2015) and Gilmer et al. (2017) use GNNs to predict molecules' property by learning their features from molecular graphs. Chang et al. (2016) use GNNs to learn object relations in dynamic physical systems. Besides, some relational reasoning models in computer vision such as (Santoro et al. 2017; Wang et al. 2018) have been shown to be variations of GNNs (Battaglia et al. 2018). Fi-GNN (Li et al. 2019) use GNNs for feature interaction modeling in CTR prediction. However, it still models all feature interactions. Our model theoretically connects the beneficial feature interactions in recommender systems to the edge set in graphs and leverages the relational reasoning ability of GNNs to model beneficial feature interactions' holistic contribution to the recommendation predictions.

L_0 Regularization L_0 regularization sparsifies models by penalizing non-zero parameters. Due to the problem of non-differentiable, it does not attract attention previously in deep learning domains until Louizos, Welling, and Kingma (2018) solve this problem by proposing a hard concrete distribution in L_0 regularization. Then, L_0 regularization has been commonly utilized to compress neural networks (Tsang et al. 2018; Shi, Glocker, and Castro 2019; Yang et al. 2017). We explore to utilize L_0 regularization to limit the number of detected edges in feature graphs for beneficial feature interaction detection.

Problem Formulation and Definitions

Consider a dataset with input-output pairs: $D = \{(X_n, y_n)\}_{1 \leq n \leq N}$, where $y_n \in \mathbb{R}/\mathbb{Z}$, $X_n = \{c_k : x_k\}_{k \in J_n}$ is a set of categorical features (c) with their values (x), $J_n \subseteq J$ and J is an index set of all features in D. For example, in app recommendation, X_n consists of a user ID, an app ID and context features (e.g., *Cloudy*, *Monday*) with values to be 1 (i.e., recorded in this data sample), and y_n is a binary value to indicate whether the user will use this app. Our goal is to design a model $F(X_n)$ that detects the most beneficial *pairwise*[1] feature interactions and utilizes only the detected feature interactions to predict the true output y_n.

Beneficial Pairwise Feature Interactions Inspired by the definition of relevant feature by usefulness (Langley et al. 1994; Blum and Langley 1997), we formally define the beneficial feature interactions in Definition 1.

Definition 1. ***(Beneficial Pairwise Feature Interactions)*** *Given a data sample $X = \{x_i\}_{1 \leq i \leq k}$ of k features whose corresponding full pairwise feature interaction set is $A = \{(x_i, x_j)\}_{1 \leq i,j \leq k}$, a set of pairwise feature interactions $I_1 \subseteq A$ is more beneficial than another set of pairwise feature interactions $I_2 \subseteq A$ to a model with X as input if the accuracy of the predictions that the model produces using I_1 is higher than the accuracy achieved using I_2.*

[1]We focus on pairwise feature interactions in this paper, and we leave high-order feature interactions in future work.

The above definition formalizes our detection goal: find and retain only a part of feature interactions that together produce the highest prediction accuracy by our model.

Statistical Interaction Statistical interaction, or non-additive interaction, ensures a joint influence of several variables on an output variable is not additive (Tsang et al. 2018). Sorokina et al. (2008) formally define the pairwise statistical interaction:

Definition 2. ***(Pairwise Statistical Interaction)*** *Function $F(X)$, where $X = \{x_i\}_{1\leq i\leq k}$ has k variables, shows* ***no*** *pairwise statistical interaction between variables x_i and x_j if $F(X)$ can be expressed as the sum of two functions $f_{\setminus i}$ and $f_{\setminus j}$, where $f_{\setminus i}$ does not depend on x_i and $f_{\setminus j}$ does not depend on x_j:*

$$\begin{aligned} F(X) =& f_{\setminus i}(x_1, ..., x_{i-1}, x_{i+1}, ..., x_k) \\ &+ f_{\setminus j}(x_1, ..., x_{j-1}, x_{j+1}, ..., x_k). \end{aligned} \tag{1}$$

More generally, if using $\boldsymbol{v}_i \in \mathbb{R}^d$ to describe the i-th variable with d factors (Rendle 2010), e.g., variable embedding, each variable can be described in a vector form $\boldsymbol{u}_i = x_i \boldsymbol{v}_i$. Then, we define the pairwise statistical interaction in variable factor form by changing the Equation 1 into:

$$\begin{aligned} F(X) =& f_{\setminus i}(\boldsymbol{u}_1, ..., \boldsymbol{u}_{i-1}, \boldsymbol{u}_{i+1}, ..., \boldsymbol{u}_k) \\ &+ f_{\setminus j}(\boldsymbol{u}_1, ..., \boldsymbol{u}_{j-1}, \boldsymbol{u}_{j+1}, ..., \boldsymbol{u}_k). \end{aligned} \tag{2}$$

The definition indicates that the interaction information of variables (features) x_i and x_j will not be captured by $F(X)$ if it can be expressed as the above equations. Therefore, to correctly capture the interaction information, our model should not be expressed as the above equations. In this paper, we theoretically prove that the interaction modeling in our model strictly follow the definition of pairwise statistical interaction, which ensures our model to correctly capture interaction information.

Our Proposed Model

In this section, we formally describe L_0-SIGN's overview structure and the two components in detail. Then, we provide theoretical analyses of our model.

L_0-SIGN

Model Overview Each input of L_0-SIGN is represented as a graph (without edge information), where its features are nodes and their interactions are edges. More specifically, a data sample n is a graph $G_n(X_n, E_n)$, and $E_n = \{(e_n)_{ij}\}_{i,j\in X_n}$ is a set of edge/interaction values [2], where $(e_n)_{ij} \in \{1, 0\}$, 1 indicates that there is an edge (beneficial feature interaction) between nodes i and j, and 0 otherwise. Since no edge information are required, $E_n = \emptyset$.

While predicting, the L_0 edge prediction component, $F_{ep}(X_n; \boldsymbol{\omega})$, analyzes the existence of edges on each pair of nodes, where $\boldsymbol{\omega}$ are parameters of F_{ep}, and outputs the predicted edge set $E_n^{'}$. Then, the graph classification component, SIGN, performs predictions based on $G(X_n, E_n^{'})$.

[2]In this paper, nodes and features are used interchangeably, and the same as edges and feature interactions.

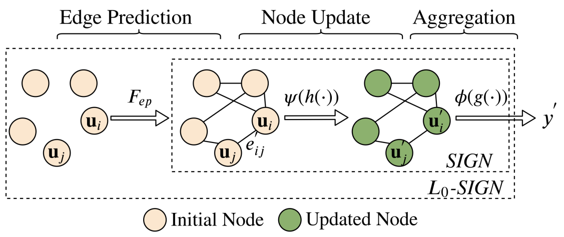

Figure 1: An overview of L_0-SIGN.

Specifically, SIGN firstly conducts interaction modeling on each pair of initial node representation that are linked by an edge. Then, each node representation is updated by aggregating all of the corresponding modeling results. Finally, all updated node representations are aggregated to get the final prediction. The general form of SIGN prediction function is $y_n^{'} = f_S(G_n(X_n, E_n^{'}); \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is SIGN's parameters and the predicted outcome $y_n^{'}$ is the graph classification result. Therefore, the L_0-SIGN prediction function f_{LS} is:

$$f_{LS}(G_n(X_n, \emptyset); \boldsymbol{\theta}, \boldsymbol{\omega}) = f_S(G_n(X_n, F_{ep}(X_n; \boldsymbol{\omega})); \boldsymbol{\theta}). \tag{3}$$

Figure 1 shows the structure of L_0-SIGN[3]. Next, we will show the two components in detail. When describing the two components, we focus on one input-output pair, so we omit the index "n" for simplicity.

L_0 Edge Prediction Model A (neural) matrix factorization (MF) based model is used for edge prediction. MF is effective in modeling relations between node pairs by factorizing the adjacency matrix of a graph into node dense embeddings (Menon and Elkan 2011). In L_0-SIGN, since we do not have the ground truth adjacency matrix, the gradients for optimizing this component come from the errors between the outputs of SIGN and the target outcomes.

Specifically, the edge value, $e_{ij}^{'} \in E^{'}$, is predicted by an edge prediction function $f_{ep}(\boldsymbol{v}_i^e, \boldsymbol{v}_j^e) : \mathbb{R}^{2\times b} \rightarrow \mathbb{Z}_2$, which takes a pair of node embeddings for edge prediction with dimension b as input, and output a binary value, indicating whether the two nodes are connected by an edge. $\boldsymbol{v}_i^e = \boldsymbol{o}_i \boldsymbol{W}^e$ is the embedding of node i for edge prediction, where $\boldsymbol{W}^e \in \mathbb{R}^{|J|\times b}$ are parameters and $\boldsymbol{o}_i$ is the one-hot embedding of node i. To ensure that the detection results are identical to the same pair of nodes, f_{ep} should be invariant to the order of its input, i.e., $f_{ep}(\boldsymbol{v}_i^e, \boldsymbol{v}_j^e) = f_{ep}(\boldsymbol{v}_j^e, \boldsymbol{v}_i^e)$. For example, in our experiments, we use an multilayer neural network (MLP) with the input as the element-wise product result of $\boldsymbol{v}_i^e$ and $\boldsymbol{v}_j^e$ (details are in the section of experiments). Note that $e_{ii}^{'} = 1$ can be regarded as the feature i being beneficial.

While training, an L_0 activation regularization is performed on $E^{'}$ to minimize the number of detected edges, which will be described later in the section about the empirical risk minimization function of L_0-SIGN.

[3]Section *D* of Appendix lists the pseudocodes of our model and the training procedures.

SIGN In SIGN, each node i is first represented as an initial node embedding $\boldsymbol{v}_i$ of d dimensions for interaction modeling, i.e., each node has node embeddings $\boldsymbol{v}_i^e$ and $\boldsymbol{v}_i$ for edge prediction and interaction modeling, respectively, to ensure the best performance of respective tasks. Then, the interaction modeling is performed on each node pair (i, j) that $e_{ij}^{'} = 1$, by a non-additive function $h(\boldsymbol{u}_i, \boldsymbol{u}_j) : \mathbb{R}^{2\times d} \rightarrow \mathbb{R}^d$ (e.g., an MLP), where $\boldsymbol{u}_i = x_i \boldsymbol{v}_i$. The output of $h(\boldsymbol{u}_i, \boldsymbol{u}_j)$ is denoted as $\boldsymbol{z}_{ij}$. Similar to f_{ep}, h should also be invariant to the order of its input. The above procedure can be reformulated as $\boldsymbol{s}_{ij} = e_{ij}^{'}\boldsymbol{z}_{ij}$, where $\boldsymbol{s}_{ij} \in \mathbb{R}^d$ is called the statistical interaction analysis result of (i, j).

Next, each node is updated by aggregating all of the analysis results between the node and its neighbors using a linear aggregation function ψ: $\boldsymbol{v}_i^{'} = \psi(\varsigma_i)$, where $\boldsymbol{v}_i^{'} \in \mathbb{R}^d$ is the updated embedding of node i, ς_i is a set of statistical interaction analysis results between node i and its neighbors. Note that ψ should be invariant to input permutations, and be able to take inputs with variant number of elements (e.g., element-wise summation/mean).

Finally, each updated node embedding will be transformed into a scalar value by a linear function $g : \mathbb{R}^d \rightarrow \mathbb{R}$, and all scalar values are linearly aggregated as the output of SIGN. That is: $y^{'} = \phi(\nu)$, where $\nu = \{g(\boldsymbol{u}_i^{'}) \mid i \in X\}$, $\boldsymbol{u}_i^{'} = x_i \boldsymbol{v}_i^{'}$ and $\phi : \mathbb{R}^{|\nu|\times 1} \rightarrow \mathbb{R}$ is an aggregation function having similar properties to ψ. Therefore, the prediction function of SIGN is:

$$f_S(G; \boldsymbol{\theta}) = \phi(\{g(\psi(\{e_{ij}^{'} h(\boldsymbol{u}_i, \boldsymbol{u}_j)\}_{j\in X}))\}_{i\in X}). \tag{4}$$

In summary, we formulate the L_0-SIGN prediction function of Equation 3 with the two components in detail [4]:

$$f_{LS}(G; \boldsymbol{\omega}, \boldsymbol{\theta}) = \phi(\{g(\psi(\{f_{ep}(\boldsymbol{v}_i^e, \boldsymbol{v}_j^e) h(\boldsymbol{u}_i, \boldsymbol{u}_j)\}_{j\in X}))\}_{i\in X}). \tag{5}$$

Empirical Risk Minimization Function

The empirical risk minimization function of L_0-SIGN minimizes a loss function, a reparameterized L_0 activation regularization[5] on $E_n^{'}$, and an L_2 activation regularization on $\boldsymbol{z}_n$. Instead of regularizing parameters, activation regularization regularizes the output of models (Merity, McCann, and Socher 2017). We leverage activation regularization to link our model with the IB principle to ensures the success of the interaction detection, which will be discussed in the theoretical analyses. Formally, the function is:

$$\begin{aligned}\mathcal{R}(\boldsymbol{\theta}, \boldsymbol{\omega}) &= \frac{1}{N}\sum_{n=1}^{N}(\mathcal{L}(F_{LS}(G_n; \boldsymbol{\omega}, \boldsymbol{\theta}), y_n) \\ &\quad + \lambda_1 \sum_{i,j\in X_n} (\pi_n)_{ij} + \lambda_2 \left\|\boldsymbol{z}_n\right\|_2), \\ \boldsymbol{\theta}^*, \boldsymbol{\omega}^* &= \underset{\boldsymbol{\theta},\boldsymbol{\omega}}{\arg\min}\, \mathcal{R}(\boldsymbol{\theta}, \boldsymbol{\omega}),\end{aligned} \tag{6}$$

where $(\pi_n)_{ij}$ is the probability of $(e_n^{'})_{ij}$ being 1 (i.e., $(e_n^{'})_{ij} = Bern((\pi_n)_{ij})$), $G_n = G_n(X_n, \emptyset)$, λ_1 and λ_2 are weight factors for the regularizations, $\mathcal{L}(\cdot)$ corresponds to a loss function and $\boldsymbol{\theta}^*$, $\boldsymbol{\omega}^*$ are final parameters.

[4]The time complexity analysis is in Section E of Appendix.

[5]Section F of Appendix gives detailed description about L_0 regularization and its reparameterization trick.

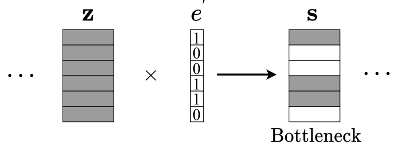

Figure 2: The interaction analysis results $\boldsymbol{s}$ from L_0-SIGN are like the relevant part (bottleneck) in the IB principle.

A practical difficulty of performing L_0 regularization is that it is non-differentiable. Inspired by (Louizos, Welling, and Kingma 2018), we smooth the L_0 activation regularization by approximating the Bernoulli distribution with a hard concrete distribution so that $e_{ij}^{'}$ is differentiable. Section G of Appendix gives details about the approximation.

Theoretical Analyses

We conduct theoretical analyses to verify our model's effectiveness, including how L_0-SIGN satisfies the IB principle to guarantee the success of beneficial interaction detection, the relation of the statistical interaction analysis results with the spike-and-slab distribution (the golden standard in sparsity), and how SIGN and L_0-SIGN strictly follow the statistical interaction theory for effective interaction modeling.

Satisfaction of Information Bottleneck (IB) Principle IB principle (Tishby, Pereira, and Bialek 2000) aims to extract the most relevant information that input random variables $\boldsymbol{X}$ contains about output variables $\boldsymbol{Y}$ by considering a trade-off between the accuracy and complexity of the process. The relevant part of $\boldsymbol{X}$ over $\boldsymbol{Y}$ denotes $\boldsymbol{S}$. The IB principle can be mathematically represented as:

$$\min \quad (I(\boldsymbol{X}; \boldsymbol{S}) - \beta I(\boldsymbol{S}; \boldsymbol{Y})), \tag{7}$$

where $I(\cdot)$ denotes mutual information between two variables and $\beta > 0$ is a weight factor.

The empirical risk minimization function of L_0-SIGN in Equation 6 can be approximately derived from Equation 7 (Section A of Appendix gives a detailed derivation):

$$\min \mathcal{R}(\boldsymbol{\theta}, \boldsymbol{\omega}) \approx \min(I(\boldsymbol{X}; \boldsymbol{S}) - \beta I(\boldsymbol{S}; \boldsymbol{Y})). \tag{8}$$

Intuitively, the L_0 regularization in Equation 6 minimizes a Kullback–Leibler divergence between every $e_{ij}^{'}$ and a Bernoulli distribution $Bern(0)$, and the L_2 regularization minimizes a Kullback–Leibler divergence between every $\boldsymbol{z}_{ij}$ and a multivariate standard distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. As illustrated in Figure 2, the statistical interaction analysis results, $\boldsymbol{s}$, can be approximated as the relevant part $\boldsymbol{S}$ in Equation 7.

Therefore, through training L_0-SIGN, $\boldsymbol{s}$ is the most compact (beneficial) representation of the interactions for recommendation prediciton. This provides a theoretical guarantee that the most beneficial feature interactions will be detected.

Relation to Spike-and-slab Distribution The spike-and-slab distribution (Mitchell and Beauchamp 1988) is the golden standard in sparsity. It is defined as a mixture of a delta spike at zero and a continuous distribution over the real line (e.g., a standard normal):

$$p(a) = Bern(\pi), \qquad p(\theta \mid a = 0) = \delta(\theta),$$
$$p(\theta \mid a = 1) = \mathcal{N}(\theta \mid 0, 1). \tag{9}$$

We can regard the spike-and-slab distribution as the product of a continuous distribution and a Bernoulli distribution. In L_0-SIGN, the predicted edge value vector $\boldsymbol{e}^{'}$ (the vector form of $E^{'}$) is a multivariate Bernoulli distribution and can be regarded as $p(a)$. The interaction modeling result $\boldsymbol{z}$ is a multivariate normal distribution and can be regarded as $p(\theta)$. Therefore, L_0-SIGN's statistical interaction analysis results, $\boldsymbol{s}$, is a multivariate spike-and-slab distribution that performs edge sparsification by discarding unbeneficial feature interactions. The retained edges in the spike-and-slab distribution are sufficient for L_0-SIGN to provide accurate predictions.

Statistical Interaction in SIGN The feature interaction modeling in SIGN strictly follows the definition of statistical interaction, which is formally described in Theorem 1 (Section *B* of Appendix gives the proof):

Theorem 1. ***(Statistical Interaction in SIGN)*** *Consider a graph $G(X, E)$, where X is the node set and $E = \{e_{ij}\}_{i,j \in X}$ is the edge set that $e_{ij} \in \{0, 1\}, e_{ij} = e_{ji}$. Let $G(X, E)$ be the input of SIGN function $f_S(G)$ in Equation 4, then the function flags pairwise statistical interaction between node i and node j if and only if they are linked by an edge in $G(X, E)$, i.e., $e_{ij} = 1$.*

Theorem 1 guarantees that SIGN will capture the interaction information from node pairs *iff* they are linked by an edge. This ensures SIGN to accurately leverage the detected beneficial feature interactions for both inferring the target outcome and meanwhile providing useful feedback to the L_0 edge prediction component for better detection.

Statistical Interaction in L_0-SIGN L_0-SIGN provides the same feature interaction modeling ability as SIGN, since we can simply extend Theorem 1 to Corollary 1.1 (Section *C* of Appendix gives the proof):

Corollary 1.1. ***(Statistical Interaction in L_0-SIGN)*** *Consider a graph G that the edge set is unknown. Let G be the input of L_0-SIGN function $F_{LS}(G)$ in Equation 5, the function flags pairwise statistical interaction between node i and node j if and only if they are predicted to be linked by an edge in G by L_0-SIGN, i.e., $e_{ij}^{'} = 1$.*

Experiments

We focuses on answering three questions: (i) how L_0-SIGN performs compared to baselines and whether SIGN helps detect more beneficial interactions for better performance? (ii) How is the detection ability of L_0-SIGN? (iii) Can the statistical interaction analysis results provide potential explanations for the recommendations predictions?

Experimental Protocol

Datasets We study two real-world datasets for recommender systems to evaluate our model:

Dataset	#Features	#Graphs	#Nodes/Graph
Frappe	5,382	288,609	10
MovieLens	90,445	2,006,859	3
Twitter	1,323	144,033	4.03
DBLP	41,324	19,456	10.48

Table 1: Dataset statistics. All datasets are denoted in graph form. Twitter and DBLP datasets are used for question (ii).

Frappe (Baltrunas et al. 2015) is a context-aware recommendation dataset that records app usage logs from different users with eight types of contexts (e,g, weather). Each log is a graph (without edges), and nodes are user ID, app ID, or the contexts.
MovieLens-tag (He and Chua 2017) focuses on the movie tag recommendation (e.g., "must-see"). Each data instance is a graph, with nodes as user ID, movie ID, and a tag that the user gives to the movie.

To evaluate the question (ii), we further study two datasets for graph classification, which will be discussed later. The statistics of the datasets are summarized in Table 1.

Baselines We compare our model with recommender system baselines that model all feature interactions:
FM (Koren 2008): It is one of the most popular recommendation algorithms that models every feature interactions. **AFM** (Xiao et al. 2017): Addition to FM, it calculates an attention value for each feature interaction. **NFM** (He and Chua 2017): It replaces the dot product procedure of FM by an MLP. **DeepFM** (Guo et al. 2017): It uses MLP and FM for interaction analysis, respectively. **xDeepFM** (Lian et al. 2018): It is an extension of DeepFM that models feature interactions in both explicit and implicit way. **AutoInt** (Song et al. 2019): It explicitly models all feature interactions using a multi-head self-attentive neural network. We use the same MLP settings in all baselines (if use) as our interaction modeling function h in SIGN for fair comparison.

Experimental Set-up In the experiments, we use element-wise mean as both linear aggregation functions $\psi(\cdot)$ and $\phi(\cdot)$. The linear function $g(\cdot)$ is a weighted sum function (i.e., $g(\boldsymbol{u_i^{'}}) = \boldsymbol{w}_g^T \boldsymbol{u_i^{'}}$, where $\boldsymbol{w}_g \in \mathbb{R}^{d \times 1}$ are the weight parameters). For the interaction modeling function $h(\cdot)$, we use a MLP with one hidden layer after element-wise product: $h(\boldsymbol{u}_i, \boldsymbol{u}_j) = \boldsymbol{W}_2^h \sigma(\boldsymbol{W}_1^h(\boldsymbol{u}_i \odot \boldsymbol{u}_j) + \boldsymbol{b}_1^h) + \boldsymbol{b}_2^h$, where $\boldsymbol{W}_1^h, \boldsymbol{W}_2^h, \boldsymbol{b}_1^h, \boldsymbol{b}_2^h$ are parameters of MLP and $\sigma(\cdot)$ is a Relu activation function. We implement the edge prediction model based on the neural collaborative filtering framework (He et al. 2017), which has a similar form to $h(\cdot)$: $f_{ep}(\boldsymbol{v}_i^e, \boldsymbol{v}_j^e) = \boldsymbol{W}_2^e \sigma(\boldsymbol{W}_1^e(\boldsymbol{v}_i^e \odot \boldsymbol{v}_j^e) + \boldsymbol{b}_1^e) + \boldsymbol{b}_2^e$. We set node embedding sizes for both interaction modeling and edge prediction to 8 (i.e., $b, d = 8$) and the sizes of hidden layer for both h and f_{ep} to 32. We choose the weighting factors λ_1 and λ_2 from $[1 \times 10^{-5}, 1 \times 10^{-1}]$ that produce the best performance in each dataset[6].

Each dataset is randomly split into training, validation,

[6]Our implementation of the L_0-SIGN model is available at https://github.com/suyixin12123/SIGN-pytorch.

	FRAPPE		MOVIELENS	
	AUC	ACC	AUC	ACC
FM	0.9263	0.8729	0.9190	0.8694
AFM	0.9361	0.8882	0.9205	0.8711
NFM	0.9413	0.8928	0.9342	0.8903
DEEPFM	0.9422	0.8931	0.9339	0.8895
xDEEPFM	0.9435	0.8950	0.9347	0.8906
AUTOINT	0.9432	0.8947	0.9351	0.8912
SIGN	0.9448	0.8974	0.9354	0.8921
L_0-SIGN	**0.9580**	**0.9174**	**0.9407**	**0.8970**

Table 2: Summary of results in comparison with baselines.

and test datasets with a proportion of 70%, 15%, and 15%. We choose the model parameters that produce the best results in validation set when the number of predicted edges being steady. We use accuracy (ACC) and the area under a curve with Riemann sums (AUC) as evaluation metrics.

Model Performance

Table 2 shows the results of comparing our model with recommendation baselines, with the best results for each dataset in bold. The results of SIGN are using all feature interactions as input, i.e., the input is a complete graph.

Through the table, we observe that: (i) L_0-SIGN outperforms all baselines. It shows L_0-SIGN's ability in providing accurate recommendations. Meanwhile, SIGN solely gains comparable results to competitive baselines, which shows the effectiveness of SIGN in modeling feature interactions for recommendation. (ii) L_0-SIGN gains significant improvement from SIGN. It shows that more accurate predictions can be delivered by retaining only beneficial feature interactions and effectively modeling them. (iii) FM and AFM (purely based on dot product to model interactions) gain lower accuracy than other models, which shows the necessity of using sophisticated methods (e.g., MLP) to model feature interactions for better predictions. (iv) The models that explicitly model feature interactions (xDeepFM, AutoInt, and L_0-SIGN) outperform those that implicitly model feature interactions (NFM, DeepFM). It shows that explicit feature interaction analysis is promising in delivering accurate predictions.

Comparing SIGN with Other GNNs in Our Model

To evaluate whether our L_0 edge prediction technique can be used on other GNNs and whether SIGN is more suitable than other GNNs in our model, we replace SIGN with existing GNNs: GCN (Kipf and Welling 2017), Chebyshev filter based GCN (Cheby) (Defferrard, Bresson, and Vandergheynst 2016) and GIN (Xu et al. 2019). We run on two datasets for graph classification since they contain heuristic edges (used to compare with the predicted edges):

Twitter (Pan, Wu, and Zhu 2015) is extracted from twitter sentiment classification. Each tweet is a graph with nodes being word tokens and edges being the co-occurrence between two tokens in each tweet.

DBLP (Pan et al. 2013) consists of papers with labels indicating either they are from DBDM or CVPR field. Each paper is a graph with nodes being paper ID or keywords and edges being the citation relationship or keyword relations.

	TWITTER		DBLP	
	AUC	ACC	AUC	ACC
GCN	0.7049	0.6537	0.9719	0.9289
L_0-GCN	0.7053	0.6543	0.9731	0.9301
CHEBY	0.7076	0.6522	0.9717	0.9291
L_0-CHEBY	0.7079	0.6519	0.9719	0.9297
GIN	0.7149	0.6559	0.9764	0.9319
L_0-GIN	0.7159	0.6572	0.9787	0.9328
SIGN	0.7201	0.6615	0.9761	0.9316
L_0-SIGN	**0.7231**	**0.6670**	**0.9836**	**0.9427**

Table 3: The results in comparison with existing GNNs. The model names without "L_0-" use heuristic edges, and those with "L_0-" automatically detect edges via our L_0 edge prediction technique.

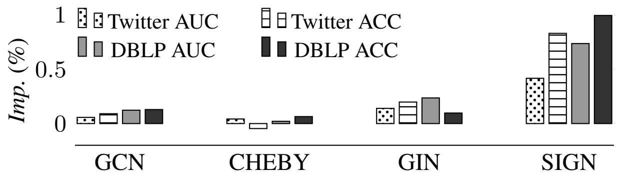

Figure 3: The comparison of improvement via the L_0 edge prediction technique to using heuristic edges in Table 3.

Table 3 shows the accuracies of each GNN that runs both on using the given (heuristic) edges (without "L_0" in name) and on predicting edges with our L_0 edge prediction model (with "L_0" in name). We can see that for all GNNs, L_0-GNNs gain competitive results comparing to corresponding GNNs. It shows that our model framework lifts the requirement of domain knowledge in defining edges in order to use GNNs in some situations. Also, L_0-SIGN outperforms other GNNs and L_0-GNNs, which shows L_0-SIGN's ability in detecting beneficial feature interactions and leveraging them to perform accurate predictions. Figure 3 shows each GNN's improvement from the results from "GNN" to "L_0-GNN" in Table 3. It shows that among all the different GNN based models, SIGN gains the largest improvement from the L_0 edge prediction technique v.s. SIGN without L_0 edge prediction. It shows that SIGN can help the L_0 edge prediction model to better detect beneficial feature interactions for more accurate predictions.

Evaluation of Interaction Detection

We then evaluate the effectiveness of beneficial feature interaction detection in L_0-SIGN.

Prediction Accuracy vs. Numbers of Edges Figure 4 shows the changes in prediction accuracy and the number of edges included while training. The accuracy first increases while the number of included edges decreases dramatically. Then, the number of edges becomes steady, and the accuracy reaches a peak at similar epochs. It shows that our model can recognize unbeneficial feature interactions and remove them for better prediction accuracy.

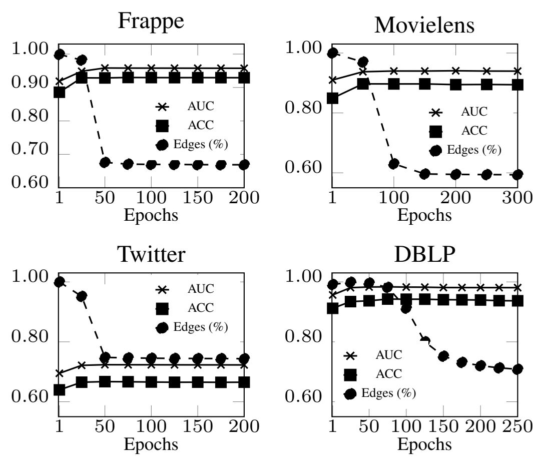

Figure 4: The changes in prediction accuracy and number of edges (in percentage) while training.

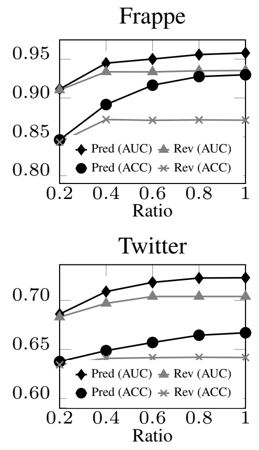
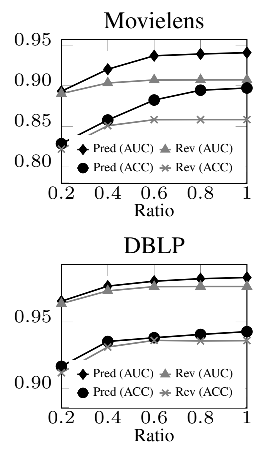

Figure 5: Evaluating different number of edges. “Pred” is the predicted edges and “Rev” is the reversed edges.

Predicted Edges vs. Reversed Edges We evaluate how *predicted edges* and *reversed edges* (the excluded edges) influence the performance. We generate 5 edge sets with a different number of edges by randomly selecting from 20% predicted edges (ratio 0.2) to 100% predicted edges (ratio 1.0). We generate another 5 edge sets similarly from reversed edges. Then, we run SIGN on each edge set for 5 times, and show the averaged results in Figure 5. It shows that using predicted edges always gets higher accuracy than using reversed edges. The accuracy stops increasing when using reversed edges since the ratio 0.6, while it continually improves when using more predicted edges. According to Definition 1, the detected edges are proved beneficial since considering more of them can provide further performance gain and the performance is the best when using all predicted edges, while considering the reversed edges cannot. Note that the increment of reversed edges from 0.2 to 0.6 may come from covering more nodes since features solely can provide some useful information for prediction.

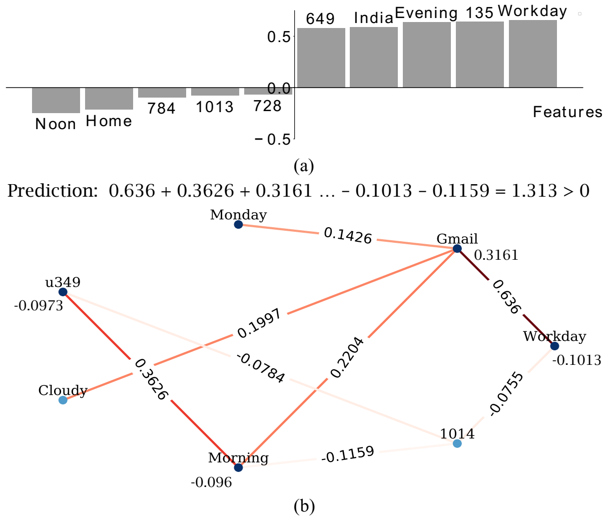

Figure 6: (a) The highest interaction values with *Gmail*. The numbers are city indexes (e.g., 1014). (b) The prediction that the user *u349* will use *Gmail* as the prediction value is 1.313 > 0. Darker edges are higher interaction values.

Case Study

Another advantage of interaction detection is to automatically discover potential explanations about recommendation predictions. We conduct case studies on the Frappe dataset to show how L_0-SIGN provides the explanations.

We first show the beneficial interactions that have the highest interaction values with *Gmail* in Figure 6a. We can see that *Workday* has a high positive value, while *Home* has a high negative value. It may show that Gmail is very likely to be used in workdays since people need to use Gmail while working, whereas Gmail is not likely to be used at home since people may not want to dealing with email while resting. We then show how L_0-SIGN provides potential explanations for the prediction on each data sample. Figure 6b visualizes a prediction result that a user (*u349*) may use *Gmail*. The graph shows useful information for explanations. Despite the beneficial interactions such as $< Gmail, Workday >$ that have discussed above, we can also see that *Cloudy* and *Morning* have no beneficial interaction, meaning that whether it is a cloudy morning does not contribute to decide the user’s will of using Gmail.

Conclusion and Future Work

We are the first to propose and formulate the problem of detecting beneficial feature interactions for recommender systems. We propose L_0-SIGN to detect the beneficial feature interactions via a graph neural network approach and L_0 regularization. Theoretical analyses and extensive experiments show the ability of L_0-SIGN in detecting and modeling beneficial feature interactions for accurate recommendations. In future work, we will model high-order feature interactions in GNNs with theoretical foundations to effectively capture high-order feature interaction information in graph form.

Acknowledgments

This work is supported by the China Scholarship Council.

References

Baltrunas, L.; Church, K.; Karatzoglou, A.; and Oliver, N. 2015. Frappe: Understanding the usage and perception of mobile app recommendations in-the-wild. *arXiv preprint arXiv:1505.03014* .

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V.; Malinowski, M.; Tacchetti, A.; Raposo, D.; Santoro, A.; and Faulkner, R. 2018. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261* .

Blondel, M.; Ishihata, M.; Fujino, A.; and Ueda, N. 2016. Polynomial Networks and Factorization Machines: New Insights and Efficient Training Algorithms. In *ICML*, 850–858.

Blum, A. L.; and Langley, P. 1997. Selection of Relevant Features and Examples in Machine Learning. *AI* 97(1-2): 245–271.

Chang, M. B.; Ullman, T.; Torralba, A.; and Tenenbaum, J. B. 2016. A compositional object-based approach to learning physical dynamics. In *ICLR*, 1–14.

Defferrard, M.; Bresson, X.; and Vandergheynst, P. 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. In *NeurIPS*, 3844–3852.

Duvenaud, D. K.; Maclaurin, D.; Iparraguirre, J.; Bombarell, R.; Hirzel, T.; Aspuru-Guzik, A.; and Adams, R. P. 2015. Convolutional networks on graphs for learning molecular fingerprints. In *NeurIPS*, 2224–2232.

Gilmer, J.; Schoenholz, S. S.; Riley, P. F.; Vinyals, O.; and Dahl, G. E. 2017. Neural message passing for quantum chemistry. In *ICML*, 1263–1272.

Guo, H.; Tang, R.; Ye, Y.; Li, Z.; and He, X. 2017. Deepfm: a factorization-machine based neural network for ctr prediction. In *IJCAI*, 1725–1731.

He, X.; and Chua, T.-S. 2017. Neural factorization machines for sparse predictive analytics. In *SIGIR*, 355–364.

He, X.; Liao, L.; Zhang, H.; Nie, L.; Hu, X.; and Chua, T.-S. 2017. Neural Collaborative Filtering. In *WWW*, 173–182.

Kipf, T. N.; and Welling, M. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*, 1–14.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *SIGKDD*, 426–434.

Langley, P.; et al. 1994. Selection of Relevant Features in Machine Learning. In *AAAI*, volume 184, 245–271.

Li, Z.; Cui, Z.; Wu, S.; Zhang, X.; and Wang, L. 2019. Fignn: Modeling Feature Interactions via Graph Neural Networks for CTR Prediction. In *CIKM*, 539–548.

Lian, J.; Zhou, X.; Zhang, F.; Chen, Z.; Xie, X.; and Sun, G. 2018. xdeepfm: Combining Explicit and Implicit Feature Interactions for Recommender Systems. In *SIGKDD*, 1754–1763.

Liu, B.; Zhu, C.; Li, G.; Zhang, W.; Lai, J.; Tang, R.; He, X.; Li, Z.; and Yu, Y. 2020. AutoFIS: Automatic Feature Interaction Selection in Factorization Models for Click-Through Rate Prediction. In *SIGKDD*, 2636–2645. ACM.

Louizos, C.; Welling, M.; and Kingma, D. P. 2018. Learning Sparse Neural Networks through L_0 Regularization. In *ICLR*, 1–11.

Lu, J.; Wu, D.; Mao, M.; Wang, W.; and Zhang, G. 2015. Recommender System Application Developments: a Survey. *DSS* 74: 12–32.

Luo, Y.; Wang, M.; Zhou, H.; Yao, Q.; Tu, W.-W.; Chen, Y.; Dai, W.; and Yang, Q. 2019. Autocross: Automatic Feature Crossing for Tabular Data in Real-world Applications. In *SIGKDD*, 1936–1945.

Menon, A. K.; and Elkan, C. 2011. Link Prediction via Matrix Factorization. In *ECML PKDD*, 437–452.

Merity, S.; McCann, B.; and Socher, R. 2017. Revisiting Activation Regularization for Language RNNs. In *ICML Workshop*.

Mitchell, T. J.; and Beauchamp, J. J. 1988. Bayesian variable selection in linear regression. *ASA* 83(404): 1023–1032.

Pan, S.; Wu, J.; and Zhu, X. 2015. CogBoost: Boosting for Fast Cost-sensitive Graph Classification. In *TKDE*, 2933–2946.

Pan, S.; Zhu, X.; Zhang, C.; and Philip, S. Y. 2013. Graph Stream Classification Using Labeled and Unlabeled Graphs. In *ICDE*, 398–409.

Rendle, S. 2010. Factorization Machines. In *ICDM*, 995–1000.

Santoro, A.; Raposo, D.; Barrett, D. G.; Malinowski, M.; Pascanu, R.; Battaglia, P.; and Lillicrap, T. 2017. A simple neural network module for relational reasoning. In *NeurIPS*, 4967–4976.

Shi, C.; Glocker, B.; and Castro, D. C. 2019. PVAE: Learning Disentangled Representations with Intrinsic Dimension via Approximated L0 Regularization. In *PMLR*, 1–6.

Shi, Y.; Larson, M.; and Hanjalic, A. 2014. Collaborative Filtering Beyond the User-item Matrix: A Survey of the State of the Art and Future Challenges. *CSUR* 47(1): 1–45.

Siegmund, N.; Kolesnikov, S. S.; Kästner, C.; Apel, S.; Batory, D.; Rosenmüller, M.; and Saake, G. 2012. Predicting Performance via Automated Feature-interaction Detection. In *ICSE*, 167–177.

Song, W.; Shi, C.; Xiao, Z.; Duan, Z.; Xu, Y.; Zhang, M.; and Tang, J. 2019. Autoint: Automatic Feature Interaction Learning via Self-attentive Neural Networks. In *CIKM*, 1161–1170.

Sorokina, D.; Caruana, R.; Riedewald, M.; and Fink, D. 2008. Detecting Statistical Interactions with Additive Groves of Trees. In *ICML*, 1000–1007.

Tishby, N.; Pereira, F. C.; and Bialek, W. 2000. The Information Bottleneck Method. *arXiv preprint physics/0004057* .

Tsang, M.; Liu, H.; Purushotham, S.; Murali, P.; and Liu, Y. 2018. Neural Interaction Transparency (NIT): Disentangling Learned Interactions for Improved Interpretability. In *NeurIPS*, 5804–5813.

Wang, X.; Girshick, R.; Gupta, A.; and He, K. 2018. Non-local neural networks. In *CVPR*, 7794–7803.

Wang, X.; Zhang, R.; Sun, Y.; and Qi, J. 2019. Doubly Robust Joint Learning for Recommendation on Data Missing Not at Random. In *ICML*, 6638–6647. PMLR.

Xiao, J.; Ye, H.; He, X.; Zhang, H.; Wu, F.; and Chua, T.-S. 2017. Attentional factorization machines: Learning the weight of feature interactions via attention networks. In *IJCAI*, 3119–3125.

Xu, K.; Hu, W.; Leskovec, J.; and Jegelka, S. 2019. How Powerful are Graph Neural Networks? *ICLR* 1–13.

Yang, C.; Bai, L.; Zhang, C.; Yuan, Q.; and Han, J. 2017. Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *SIGKDD*, 1245–1254.

Zhang, C.; Bengio, S.; Hardt, M.; Recht, B.; and Vinyals, O. 2017. Understanding Deep Learning Requires Rethinking Generalization. In *ICLR*, 1–11.