Learning Personalized Representations using Graph Convolutional Network

Hongyu Shen*
Amazon Alexa
Seattle, WA, USA
hongyus@amazon.com

Jinoh Oh
Amazon Alexa
Seattle, WA, USA
ojino@amazon.com

Shuai Zhao
Amazon Alexa
Seattle, WA, USA
shuzhao@amazon.com

Guoyin Wang
Amazon Alexa
Seattle, WA, USA
guoyiwan@amazon.com

Tara Taghavi
Amazon Alexa
Seattle, WA, USA
taghavit@amazon.com

Sungjin Lee
Amazon Alexa
Seattle, WA, USA
sungjinl@amazon.com

ABSTRACT

Generating representations that precisely reflect customers' behavior is an important task for providing personalized skill routing experience in Alexa. Currently, Dynamic Routing (DR) team, which is responsible for routing Alexa traffic to providers/skills, relies on two features to be served as personal signals: absolute traffic count and normalized traffic count of every skill usage per customer. Neither of them considers the network-structure for interactions between customers and skills, which contain richer information for customer preferences. In this work, we first build a heterogeneous edge-attributed graph based customers' past interactions with the invoked skills, in which the user requests (utterances) are modeled as edges. Then we propose a graph convolutional network(GCN)-based model, namely Personalized Dynamic Routing Feature Encoder (PDRFE), that generates personalized customer representations learned from the built graph. Compared with existing models, PDRFE is able to further capture contextual information in the graph convolutional function. The performance of our proposed model is evaluated by a downstream task, defect prediction, that predicts the defect label from the learned embeddings of customers and their triggered skills. We observe up to 41% improvements on the cross-entropy metric for our proposed models compared to the baselines.

CCS CONCEPTS

• **Computing methodologies** → *Learning in probabilistic graphical models.*

KEYWORDS

graph neural networks, recommendation system, deep learning, personalization

*Work is done during the internship at Amazon Alexa AI.

MLG'22, August 15, 2022, Washington DC

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Hongyu Shen, Jinoh Oh, Shuai Zhao, Guoyin Wang, Tara Taghavi, and Sungjin Lee. 2022. Learning Personalized Representations using Graph Convolutional Network. In *Proceedings of International Workshop on Mining and Learning with Graphs (MLG'22).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The advancement of Alexa AI technologies enables customers to have various options for interacting with Alexa devices. For example, a person can ask Echo to play a song, to turn on a TV, or to open air conditioners while being away from home. Alexa provides such convenience to help people automate daily tasks with ease. So far, Alexa AI has created more than 200k different skills that range from video display, automatic subscriptions to home services and etc. Skills are similar to apps to process each customer query request. Such services cover almost every possible task in customer's daily life. Providing such high-end services with Alexa has also attracted nearly 800 million registered users generating about 500 million user-Alexa interactions daily.

One of the core concepts behind Alexa services is the representation learning models created by Alexa Dynamic Routing (Alexa DR) team. Alexa DR develops ranking models and rules to generate representations of the skills and use these representations to rank each skill for every customer's request. Subsequently, customers' requests can be routed to the skills (e.g. the internal function in Alexa that provides video display or weather broadcasting) that can response to those requests. Recently, Alexa DR starts to extend its ranking models to consider customer-skill interactions by using simple features such as normalized impression counts of successful customer interactions with skills as input features to its ranking models. However, these simple features are not strong enough to fully encode the complex nature of customer-skill interactions. For example, a simple impression count hardly captures the similarity between the skills, and thus looses its predictive power in that regard. Another pitfall is that simple features are computed and used without any consideration on the context. For example, the normalized traffic count for a pair of customer and skill is invariant to the utterance text.

In fact, previous studies have widely demonstrated the superior performance of applying graph modeling to describe interactions between two or more entities [5, 10, 11, 13–15]. This is similar to

modeling customer-sill interactions in the form of a graph. there are also studies pointing out that a graph encoded with contextual information can further improve model performance from various perspectives [2, 4, 6, 8]. It provides a natural mechanism to consider context information during modeling through edge-based convolution operations , which validates the attempt in combining the utterance text in modeling customer-skill relations.

However, there is one limitation when using the existing GCN models in our particular setup: they does not consider the edge attribute in characterizing the directional between the customer and the skill nodes. For example, with existing GCN-based models, the edge between customer and skill is predicted only by the customer and skill embedding or by a shared edge attribute that addresses equal weights on its connecting nodes. This means a customer will have the same edge probability to a certain skill regardless of the utterance text, which does not capture the true interaction for the probability of matching the customers to a certain skill should be different given different customers and the same utterance in general. Furthermore, the probabilities for the match of skills to a certain customer given different utterances should be different. The two existing issues in our particular setup suggests the need of a new GCN algorithm.

In this paper, we propose a model, named Personalized Dynamic Routing Feature Encoder (PDRFE), that aims to generate personalized embeddings for the customer nodes given the customer and his/her triggering utterances for different skills. PDRFE includes a *personalizer* mechanism, to provide customer embeddings adjusted with respect to the utterance text. Besides, we also consider two other modules in PDRFE to further improve the model performance by considering the edge features. The first one is the edge-based attention module. PDRFE applies attention module to distinguish the important connection and update the customer embedding and skill embedding accordingly. This is because some skills are designed to be more interactive than others, and create more frequent but less important request such as "next", "repeat the question". Thus, importance of edge needs to be distinguished and weighted differently to precisely capture the customer's true preference to skills. The second module is the edge convolution that considers all the edge features in updating the corresponding node embedding. In our case in Alexa, the edge features (e.g. utterances) indicate the customers behaviours to their invoked skills. And it also helps identifying the difference between skills. As a result, it is natural to consider the edge information in updating the node embeddings for both the customers and skills. Overall, we compare our proposed model with 4 baselines through a downstream evaluation task (i.e., defect prediction) that predicts the defect label from the personalized embedding of customers and the embedding of their triggering skills. Additionally, we also provide ablation study for the modules we proposed in PDRFE to help identify the contribution of sub-modules to the downstream task.

The rest of paper is organized as follows: We first discuss problem setup and the proposed PDRFE model in the next section, followed by the experiment section that includes experiment setup, the comparison results between PDRFE and the 4 baseline models through a downstream task evaluation, as well as ablation studies for PDRFE. Finally, we conclude our study and explicitly state the customer impact of this study for Amazon.

2 PROBLEM DEFINITION

Here we formally define our problem in the context of graph representation learning:

Given a bipartite customer-skill interaction graph $G = (V, E)$ where V is the set of nodes and E is the set of edges and $V = \{U, S\}$ where U refers to the set of customer nodes and S refers to the set of skill nodes and $e_{u,s}$ denotes that a service request from user u is fulfilled by skill s ($e_{u,s} \in E$) (Note that, here we simplify the notation, but there can be multiple interactions between a customer u and a skill s). For simplicity, we use $\mathbf{e}_{u,s}$ to represent the feature vector to the edge $e_{u,s}$. The objective is to design a model f that maps the initial input embeddings $(\mathbf{h}_u^0, \mathbf{e}_{u,s}, \mathbf{h}_s^0)$ of a graph to a set of final embedding vectors for both the user and skill nodes, denoted as $(\mathbf{h}_u^L, \mathbf{h}_s^L) = f(\mathbf{h}_u^0, \mathbf{e}_{u,s}, \mathbf{h}_s^0)$, such that when incorporating these embeddings to the downstream task, the corresponding objective function is optimized. Here the superscript L for both $\mathbf{h}_u$ and $\mathbf{h}_s$ indicates the output embeddings of both customers and skills from the GCN model f. We choose the objective for representation learning as link prediction. In short, we have:

- **Input:** G
- **Output:** $\mathbf{h}_u^L, \mathbf{h}_s^L$ for all $u, s \in V$
- **Link prediction:**
  $\sum_{u \in U} \sum_{s \in \mathcal{N}^+(u), \hat{s} \in \mathcal{N}^-(u)} \max(M - < \mathbf{h}_u^L, \mathbf{h}_s^L > + < \mathbf{h}_u^L, \mathbf{h}_{\hat{s}}^L >, 0)$
- **Downstream task (defect prediction):**
  $\min_\sigma \frac{1}{|E|} \sum_{u \in U, s \ in S} CE(\sigma(\mathbf{h}_u^L, \mathbf{h}_s^L), y)$,

where $\sigma(\cdot, \cdot)$ is the defect classifier, and $< \cdot, \cdot >$ is the inner product. $\mathcal{N}^+(u)$ is positive neighbors — the set of skill node that are connected to the customer u node, and $\mathcal{N}^-(u)$ is negative neighbors - the set of randomly sampled noise skill nodes that are not connected to the customer node u in the graph G. M is a hyperparameter for the link prediction objective, y is defect label, and $CE(\cdot, \cdot)$ is the cross-entropy loss function.

3 PROPOSED METHOD

Here we describe the structures of the proposed PDRFE model. Specifically, it involves 3 major components: 1) An encoder for edge attributes 2) An edge-based convolution operator in GCN that updates the embedding $(\mathbf{h}_u, \mathbf{h}_s)$ 3) A personalizer designed for generating personalized customer embedding that reflects a customer's request. Overall, Figure 1 illustrates the pipeline for the PDRFE model.

3.1 Encoder for Edge Attributes

In this work, the primary edge attribute is utterance text. We use a pre-trained Bi-LSTM text encoder to convert this utterance to feature vector, and there are three reasons. First, our focus is about the interaction and not the semantic meaning of the utterance. Thus, if it is carelessly trained from the scratch, there is a risk that the learned text encoder does not fully reflect semantic meaning of the utterance. Second, using a pre-trained model reduces complexity in training and enables quicker iterations. Third, using the Bi-LSTM over the other structures is to capture the cross-hypothesis correlation, which is an intermediate for generating utterance embeddings [7]. Please note that this pre-trained model is widely used

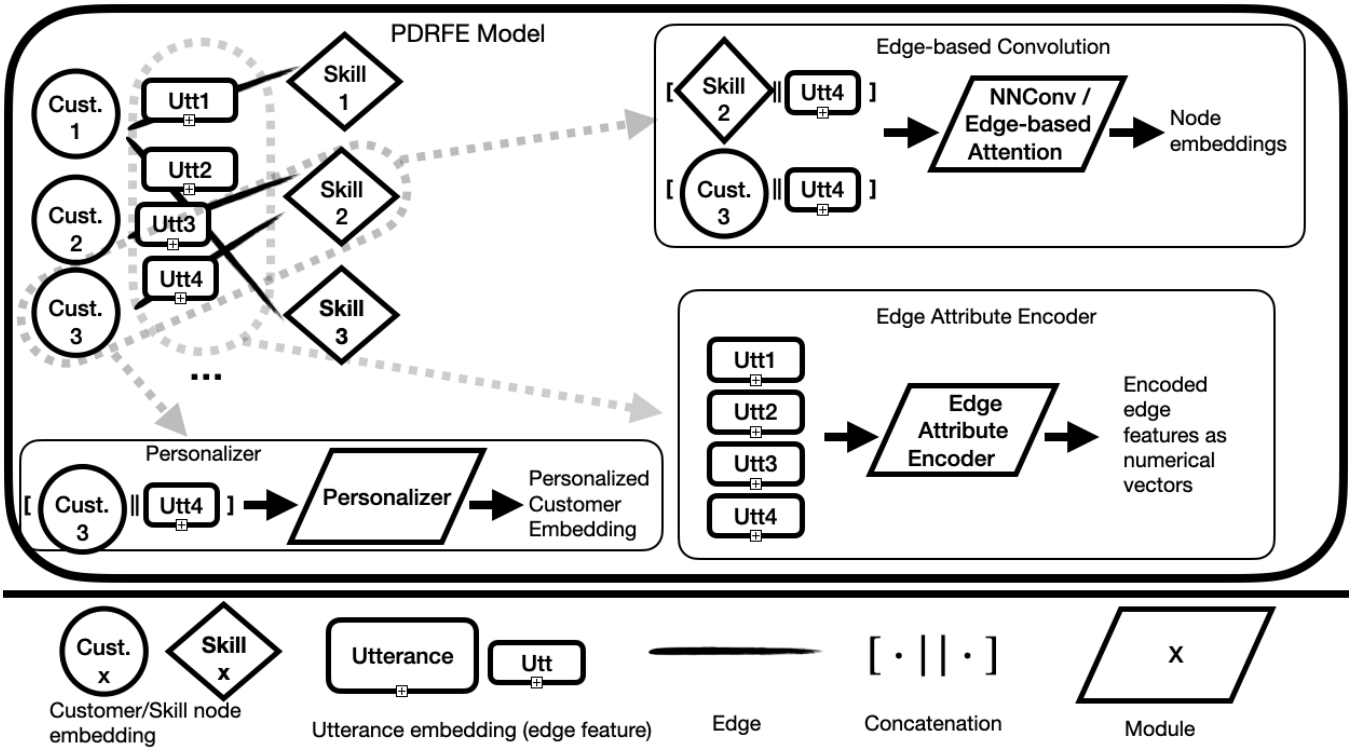


Figure 1: Diagram of PDRFE Pipeline. There are three main components in the PDRFE pipeline. 1. The edge attribute encoder encodes all utterances to numerical vectors, which are the edge features in our graph. 2. The personalizer personalizes the customer node embedding by feeding the concatenation of customer node embedding and the utterance embedding to a personalizer module. 3. The Edge-based convoulution aggregates information concerning both the skill node embedding, personalized embedding and the utterance embedding.

Table 1: Fake Examples for Customer-Skill Interaction Log

Customer ID (CID)	Skill ID (SID)	Utterance (interaction1)	Utterance (interaction2)	...
10xxxxx	Help	What happened to my music	Can you help me	...
21xxxxxx	Help	What can I do today	Can you do a tutorial	...
21xxxxxx	Communication	No	Connect to my phone	...
89xxxxx	Help	Can you do a tutorial	I have feedback	...
10xxxxx	Communication	Drop in all devices	Connect to my phone	...
...	...	...	...	...

in Alexa, hence the performance of this model has already been fully tested in production.

3.2 Edge-based Convolution Operator

Edge-based convolution operators allow the propagation of the edge features to all of its connected neighboring nodes. It is known that considering edge features during graph convolution usually enriches the information for the nodes because such edge information characterizes the natural interaction between the connected nodes [2–4]. For example, an edge can record the bond types in a molecular graph, the rating scores in a user-movie social media graph, or the contextual information in our customer-skill graph in Alexa, etc. To improve the performance of our model, we explore two edge-based convolution operators.

NNConv: NNConv is proposed in the Message Passing Neural Network designed to infer quantum properties of molecules based on the structure of the molecules and their bond types (e.g. edge features) [3]. It demonstrates the superior performance to the corresponding baselines in terms of the inferences on quantum

properties by considering the edge features in GCN message passing stage. Analogous to their setup, where customer and skill nodes refer to the vertices of molecules and utterances refer to the the bond types, we choose to apply this module in the proposed PDRFE model. For completeness, the message passing function is shown in Eq. 1:

$$\mathbf{h}_i^{l+1} = \mathbf{h}_i^l + \text{mean}_{j \in \mathcal{N}(i)} \left(\left\{ (W_e \cdot \mathbf{e}_{i,j} + b_e) \mathbf{h}_j^l \right\} \right), \tag{1}$$

where mean$(\cdot)$ refers to the mean aggregation function in GCN. h_i^l refers to the embedding of i-th node at layer l. W_e and b_e are the linear transformation parameters for the input edge features $\mathbf{e}_{i,j}$. $\mathcal{N}(i)$ refers to the neighboring node set to the node i.

Edge-based attention:. In GCNs, it is well known that attention allows for the efficient assignments of different weights to the connected edges of a node, which endows the interpretability on the importance of those edges to that node [11, 13, 14]. In our case, since utterances characterize customer-skill interactions, and attention is used to determine the weights of such interactions, we can also introduce the utterance factor into the attention module to control the generation of attention weights. This way, the attention weights would reflect the importance of such utterance as compared to all other utterances connected to the same node. Please refer to Eq. 2-4 below for details:

$$\mathbf{h}_i^{l+1} = \sum_{j \in \mathcal{N}(i)} \alpha_{i,j} W^l \mathbf{h}_j^l, \tag{2}$$

$$\alpha_{i,j}^l = \text{softmax}(k_{i,j}^l), \tag{3}$$

$$k_{i,j}^l = \text{LeakyReLU} \left(\vec{a}^T [W_a [\mathbf{h}_i^l \| \mathbf{e}_{i,j}] \| W_a [\mathbf{h}_j^l \| \mathbf{e}_{i,j}]] \right), \tag{4}$$

where W_a and W^l are the linear encoding matrix for attention module and for message passing at layer l, respectively. $\alpha_{i,j}$ and $\vec{a}$ refer to the attention weights and linear encoding vector for attention module, respectively. $[\cdot \| \cdot]$ refers to the concatenation operator.

3.3 Personalizer

The typical output of GCN model is node embeddings. However, these node embeddings are too generic to be used for context-sensitive applications as ours. For example, the link prediction objective, shown in Section 2, predicts the edge between a customer and a skill using simple inner product without having any context considerations. To resolve this problem, we design a new mechanism to provide the context-tailored customer embedding based on the learned customer embedding and utterance. We denote this mechanism as a *Personalizer*. Formally defined as follows,

$$\mathbf{h}_s^p = f_p([\mathbf{h}_s^l \| utt]) \tag{5}$$

where $\mathbf{h}_s^p$ is context-considered customer embedding, f_p is a *personalizer* function, and utt is the utterance text that we would like to consider.

In this paper, we modeled f_p to be a 2-layer neural network. This personalized embedding $\mathbf{h}_s^p$ replaced the conventional context-free customer embedding $\mathbf{h}_u^L$ in link prediction objective (Section 2), as well as replacing the customer embedding in the downstream task. This *Personalizer* also works with parallel edges where the customer node is connected to the skill node via multiple different utterances. Note that, this mechanism is trained during the representation learning time and not the downstream task training time.

4 EXPERIMENT SETUP & RESULTS

In this section, we describe how we setup our experiment and the dataset for training/evaluation for both representation learning and downstream evaluation (Section 4.1), define the baseline models to compare our work with (Section 4.2), perform hyper-parameter tuning (Section 4.3). We then present the results of our best performing model, and provide ablation study on the modules proposed in the previous section to demonstrate the effects of each of the modules to the downstream task.

4.1 Dataset Setup

In this section, we provide the details of the dataset preparation for both representation learning (a graph dataset) and downstream evaluation (a regular input-output dataset). Both datasets are generated from the raw customer-skill interaction logs that are randomly collected to remove biases during data collection. For privacy, we only provide a fake example for illustration purposes (see Table 1). Additionally, we include metadata come with the interaction logs in Table 2. These metadata will be used either in generating the initial node embeddings or for edge feature assignments. Note that the "Defect" in Table 2 is an 0-1 binary label used for y, which is predicted by CPDR team, in the downstream evaluation.

4.2 Baselines

In order to verify the performance of our proposed model, we compare it with 4 baseline models. Here we list all these models in Table 3 with reasons of selection.

4.2.1 Representation Learning. To prepare the graph for representation learning, firstly we subsample our dataset following these three steps to obtain a subsampled interaction log: 1) Randomly subsample 10% customers by their customer ID (CID). 2) Collect all the interactions corresponding to the subsampled CIDs in 14-day logs. Then we turn this log into a graph by treating each interaction log as an edge in the graph connecting the corresponding customer and skill nodes. Overall, the processed graph contains 107,480,710 number of customer-skill interactions (e.g. edges) for 12,031,327 customers (e.g. customer node) and 8,110 skills (e.g. skill node).

To train and evaluate the representation model, we randomly split this graph to two subgraphs based on uniform edge sampling. We sample 80% edges for training graph and the other 20% for testing graph.

To conduct the experiments, we use an Amazon AWS cluster with V100 NVIDIA GPUs with the following libraries: PyTorch [9], DGL [12] and ""torch-two-sample" [1].

4.2.2 Downstream Evaluation. Once the representation model is trained, we would generate embeddings for all the nodes, including personalized embeddings. We then convert the graph back into the customer-skill interaction log table with the only change udated to the personalized embeddings for given customer, skill and the corresponding utterance (e.g. $\mathbf{e}_{i,j}$). We randomly split the dataset into

Table 2: Metadata in the Customer-Skill Interaction Logs

Name of Metadata	Description	Cardinality	Example	Usage
Skill_category	Category of skills	22	Social / Education & reference	For initializing skill node embeddings
Skill_type	Skill type	7	CUSTOM	
Skill_subcategory	Subcategory of skills	69	Education & Reference / Social Networking	
Reporting_category	General category	28	Education / Social	
wbr_cor	Weekly business review country	19	US / ROW	For initializing customer node embeddings
is_prime	Prime membership?	2	Y / N	
is_amu	Amazon music subscriber?	2	Y / N	
is_smart_home_cust	Smart home customer ?	2	Y / N	
Utterance	Utterances (users' requests)	N/A	Play tiktok	Edge features
Defect	Whether a request is fulfilled successfully	2	1 / 0	

Table 3: Baseline Models

Model Name	Description	Reason of the Choice
PinSage [16]	PinSage is a large scale GCN model designed by Pinterest for recommending pins (images) to people. Considered only one node type	Representation learning model that has been applied to real-world scenarios
Revised PinSage	Extend PinSage to 2 types of nodes (both customer and skill)	Improve the original PinSage for comparison in our case
Rational Graph Convolutional Network (RGCN) [10]	Simple GCN designed for link prediction	Baseline for link prediction
Raw One-hot Encoding	No Training, just use one-hot encoder on the user/skill metadata and use these for the downstream task	Baseline for non-graph-based approach

Table 4: Legend for Reference Models.

Reference Model Configuration	Description
RGCN	The backbone RGCN model
EC	The backbone RGCN + NNConv
EAtt	The backbone RGCN + edge-based attention
Per	The backbone RGCN + *Personalizer*

train/validation/test sets in the standard way with split ratios being 6:2:2. The data are then evaluated with the proposed downstream objective (see Section 2).

4.3 Hyperparameter Setup

The following hyperparameter setup (shown in Table 6) is applied for the representation learning and the downstream evaluation. For the downstream evaluation, we use two classifiers: 1) logistic regression module being the linear classifier 2) A nonlinear classifier which consists of a 2-layer neural network with ReLU activation function. The hidden dimension for this network is 32. We choose to use a small number for the hidden dimension to avoid overfitting of this nonlinear classifier, which can improve the quality of learned representation.

4.4 Model Comparisons

Here we present the comparison between the PDRFE model and the 4 baseline models. Results are evaluated according to the testing cross entropy for the downstream defect prediction task (refer to

Table 5: Comparison between Our Best Models with the Baselines. Evaluation is based on the testing cross-entropy in the downstream evaluation. Bold numbers refer to the best results.

Models	Cross-Entropy (lower the better)	
	Logistic Regression	2-layer Neural Network
PinSage (Baseline)	0.543	0.441
Revised PinSage (Baseline)	0.519	0.421
Raw One-Hot Encoding (Baseline)	0.462	0.424
RGCN (Baseline)	0.372	0.371
PDRFE (w/ "NNConv")	**0.317**	**0.303**
PDRFE (w/ edge-based attention)	**0.324**	**0.302**

Table 6: Hyperparameter Steup

Hyperparameter	Model	
	Representation Learning	Defect Prediction
Batch Size	512/1024/ 2048/4096 (×5 for negative sampling)	256
Learning Rate	10^{-4}	10^{-4}
Hidden Dimension	128	32
Total # of Epochs	≤ 10	When validation loss is stable

Table 5). It is clear that the PDRFE models (with either the "NNConv" or the edge-based attention) obtain lower cross-entropy. The relative improvements in cross-entropy ranges between 15% to 41% for logistic regression model, and 19% to 31% for the 2-layer neural network model. In addition, it is obvious that the incorporation of the utterance embeddings into the model contribute to huge improvements for the downstream task. On one hand, the edge features improve the precision for node representations. On the other hand, the *personalizer* resolves the two aforementioned ambiguities. Both components contribute to the out-performance of PDRFE over the baseline models.

4.5 Ablation Study

To study the individual effects of each module in PDRFE, we perform this ablation study on the 4 modules in PDRFE. Results are shown in Figure 2. It is shown that in all cases in this ablation study, the models that have the proposed components in PDRFE outperform the reference models where such components are removed. Specifically, the maximum performance boost for the 3 components are: 1) *Personalizer*: 12% 2) "NNConv": 19% 3) Edge-based attention: 13%. Each component can contribute to improvements in the defect prediction. However, when consider them jointly in PDRFE model, the performance is even better. This indicates the necessity of incorporating the edge features into the message passing as well as the *personalizer* in the GCN model.

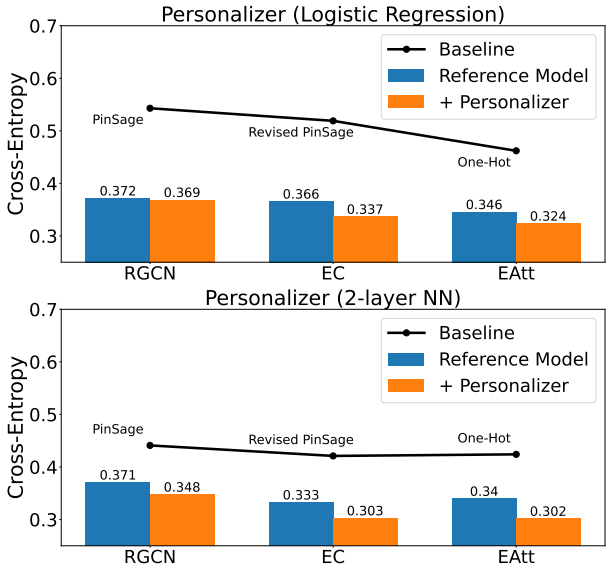
5 CONCLUSION

In this work, we build a heterogeneous graph with attributed edges based on customers' past interactions with the invoked skills. Besides, we propose PDRFE, a GCN-based representation learning model, designed for generating personalized customer-skill embeddings from the built graph. The proposed model is evalulated according to a binary classification downstream task, known as defect prediction, which reflects the customer's preferences to the skills. We compare our models to the proposed baselines and observed up to 41% improvement in the cross-entropy in the downstream evaluation. Additionally, an ablation study is included to understand the performance contribution of the modules in PDRFE where the resulting improvements range from 12% to 19%.

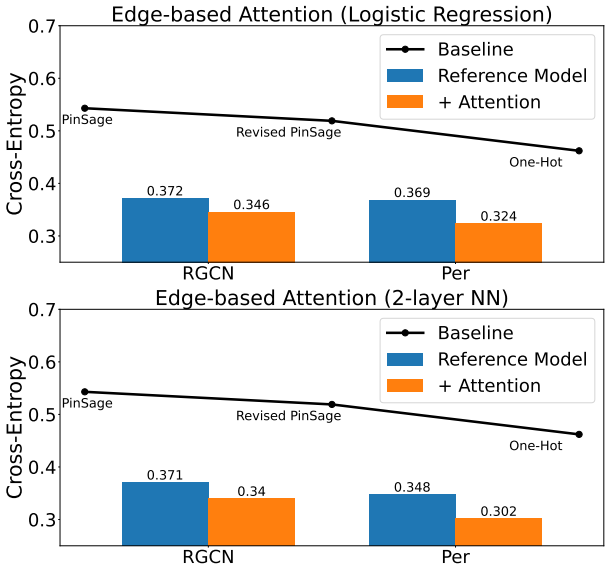
Our PDRFE model impacts customer experience and improves downstream customer satisfaction by reducing the defect rate during fulfillment of a user's request by Alexa's skills. We demonstrate that PDRFE model generates more precise customer and skill representations which encode customer-skill relationship, resulting in lower cross-entropy in the defect prediction. As a result, such characterization of customer-skill connectivity by PDRFE can be possibly applied to the existing Alexa DR tools in order to improve the accuracy of the current routing services. For example, it can benefit Alexa DR's ranking model (HypRank [7]) which considers joining the information from the user side such as utterance, interpretation, etc. and forms hypothesis to re-rank the existing skill candidates to find the most likely skill to answer a user's request. It reduces the defect rate while answering users' requests. However, this model does not consider the customer and skill interactions as a graph, which means the current representations in HypRank may not reflect the natural interactions between customers and skills. In the future work, we plan to inject the representations from the proposed PDRFE model to provide more contextual information for the customers and skills to reflect their natural connectivity into HypRank ranking decisions.

REFERENCES

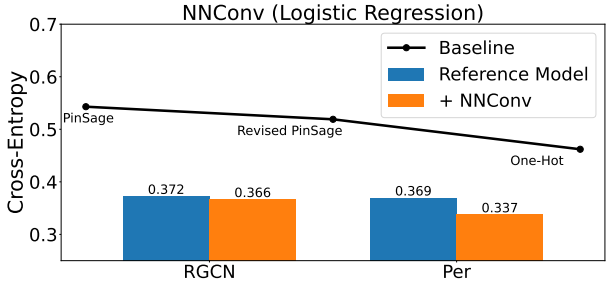
[1] [n.d.]. Torch-Two-Sample Library. https://torch-two-sample.readthedocs.io/en/latest/.
[2] Vijay Prakash Dwivedi and Xavier Bresson. 2020. A generalization of transformer networks to graphs. arXiv preprint arXiv:2012.09699 (2020).
[3] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In International conference on machine learning. PMLR, 1263–1272.
[4] Liyu Gong and Qiang Cheng. 2019. Exploiting edge features for graph neural networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 9211–9219.

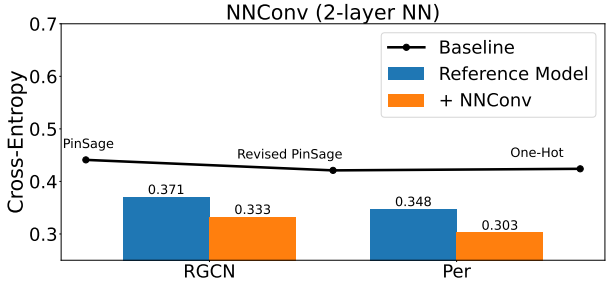


(a) *Personalizer*



(b) Edge-based attention





(c) "NNConv"

Figure 2: Ablation study on modules in PDRFE. The 3 bar-plots include the comparisons between the models with and without the modules discussed in the method section. Specifically, there are 3 modules included in this comparison: 1. The *personalizer*; 2. The NNConv; 3. The edge-based attention. The reference models (in blue) are the models without the component, which are compared to the models with the component (in orange). The numbers reported in the figures are cross-entropies. The sub-captions indicate which module in PDRFE we compare with, and the tick labels indicate the name of the reference models with details in the legend Table 4.

[5] William L. Hamilton, Rex Ying, and Jure Leskovec. 2017. Representation Learning on Graphs: Methods and Applications. IEEE Data Eng. Bull. 40, 3 (2017), 52–74. http://sites.computer.org/debull/A17sept/p52.pdf

[6] Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In Proceedings of The Web Conference 2020. 2704–2710.

[7] Young-Bum Kim, Dongchan Kim, Joo-Kyung Kim, and Ruhi Sarikaya. 2018. A scalable neural shortlisting-reranking approach for large-scale domain classification in natural language understanding. arXiv preprint arXiv:1804.08064 (2018).

[8] Ao Li, Zhou Qin, Runshi Liu, Yiqun Yang, and Dong Li. 2019. Spam review detection with graph convolutional networks. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management. 2703–2711.

[9] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga,

Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[10] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In European semantic web conference. Springer, 593–607.

[11] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. arXiv preprint arXiv:1710.10903 (2017).

[12] Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, et al. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. arXiv preprint arXiv:1909.01315 (2019).

[13] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 950–958.

[14] Xiao Wang, Houye Ji, Chuan Shi, Bai Wang, Yanfang Ye, Peng Cui, and Philip S Yu. 2019. Heterogeneous graph attention network. In The World Wide Web Conference. 2022–2032.

[15] Zhouxia Wang, Tianshui Chen, Jimmy Ren, Weihao Yu, Hui Cheng, and Liang Lin. 2018. Deep reasoning with knowledge graph for social relationship understanding. arXiv preprint arXiv:1807.00504 (2018).

[16] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 974–983.