CONTRASTIVE KNOWLEDGE GRAPH ATTENTION NETWORK FOR REQUEST-BASED RECIPE RECOMMENDATION

Xiyao Ma, Zheng Gao* , Qian Hu, Mohamed Abdelhady*

Amazon Alexa AI

ABSTRACT

To improve daily customer experience, kitchen assistant becomes one of the enabled service in intelligent voice assistants, presenting personalized and relevant recipes to satisfy customer requests. Current solutions for recipe recommendation suffers from two limitations: First, user-recipe interactions are modeled in a uniform manner, which neglects the diversity of user preferences on recipe adoptions, inherently hurting the model performance. Second, users may interact with recipe randomly, resulting in inevitable data noise issue. In this work, we alleviate the foregoing issues by proposing contrastive knowledge graph attention network for recipe recommendation, where a knowledge graph attention-based recommender helps learn fine-grained user and recipe embeddings by modeling diversified user preferences from user behaviors. Moreover, a contrastive learning module that integrates unsupervised and supervised contrastive learning is proposed to improve model robustness. The experimental results on two real-world datasets show that the proposed approach outperforms the state-of-the-art baseline methods.

Index Terms— natural language understanding, recommendation system, knowledge graph, contrastive learning

1. INTRODUCTION

Intelligent personal assistants (IPAs) such as Alexa, Siri, and Google Assistant can fulfill users' requests and make people's daily lives convenient by answering questions ranging from weather to stock price. To further enrich user experience, an IPA can not only answer questions regarding to food, ingredients, nutrition, and cooking tips, but also recommend personalized recipes according to customer requests. For example, when a customer asks for recipes by "show me recipes for chicken", the IPA is able to recommend recipes such as such as "Chicken Parmesan","Buffalo Chicken Wings", and so on.

Extensive research efforts have been made to recommend top-k items for a target user. Content-based approaches [1, 2] tend to produce items matching the user's interests according to the various features of items. Collaborative filtering (CF) [3] learns user preference on historical user-item interactions with the assumption that behaviorally similar users are likely to have similar preference on items, ranging from non-graph-based methods like MF and NCF [4, 5], to graph-based models like NGCF and LightGCN [6, 7]. Content-based CF methods are also proposed to combine the best of the two worlds [8, 9]. To alleviate label sparsity issue, knowledge graphs-enhanced approaches such as KGAT [10] and KGCN [11] are proposd with different information propogation methods on knowledge graphs.

However, the methods are still suffering from two limitations: (1) Without considering the diversified user preferences over different recipes could lead to coarse-grained representation and suboptimal performance, as customers can interact with recipes in various behaviors that reflect different levels of user preferences. For instance, "cooking along" behavior reflects stronger user preference but are with less amount than "browse" behavior in the dataset. (2) As noisy behaviors (e.g., random clicks) commonly exist, the learnt user and recipe representations are less robust to noisy interactions.

To alleviate the foregoing issues, we propose a contrastive knowledge graph attention network (C-KGAT). In details, we first present a knowledge graph attention-based recommender to learn fine-grained user and recipe embeddings by profiling user diversified preferences from user sequential behaviors. To improve model robustness, we propose a contrastive learning module with two auxiliary tasks to improve model robustness. We conduct experiments on two internal recipe datasets. Our model not only outperforms state-of-the-art baselines in terms of accuracy, but also show superiority regarding to model robustness.

2. PROBLEM FORMULATION

Given a target user and her utterance request, we aim to recommend top-K relevant and personalized recipes to the user by learning a model to predict the probability of a target recipe that the user would adopt. Generally, this recipe recommendation scenario involves a set of users $\mathcal{U}$, a set of recipes $\mathcal{I}$ that each comes along with textual and categorical features (e.g, recipe name, rating, and cooking time), and observed implicit feedbacks $\mathcal{Y}^+ = \{y_{ui} = 1 | u \in \mathcal{U}, i \in \mathcal{I}\}$, where entry y_{ui} indicates that user u interacted with recipe i before with a sequential behaviors $b_1, b_2, ..., b_k$. Each behavior is associated with a behavior type from a pre-defined and business-oriented behavior types list, including 'browse", "check ingredients", "add to cart", and "start cooking along". Furthermore, we construct a recipe knowledge graph (KG) with recipe enti-

ties $\mathcal{I}$ and different types of attribute entities $\mathcal{E}$, including cuisine, ingredients, keywords, and dietary information. Different types of relations connect recipes and corresponding types of attribute entities. We further insert user-recipe interactions into the KG, resulting in a **Collaborative Knowledge Graph (CKG)** $\mathcal{G} = \{(h, r, t) \mid h, t \in \mathcal{V}, r \in \mathcal{R}\}$ where node set $\mathcal{V} = \mathcal{E} \cup \mathcal{I} \cup \mathcal{U}$ and the relation set $\mathcal{R}$ includes observed interactions $\mathcal{Y}^+$ and relations in the recipe KG [10].

3. PROPOSED METHODS

In this work, we propose contrastive knowledge attention for recipe recommendation. As shown in Figure 1, it mainly consists of three components: (1) Knowledge graph embedding leverage the structure of CKG to learn entity embedding. (2) A KGAT-based recommender is proposed to learn collaborative user and recipe embeddings by modeling the diversified user preferences. (3) A contrastive learning module contrast user and recipe embeddings from different graph views to improve model robustness.

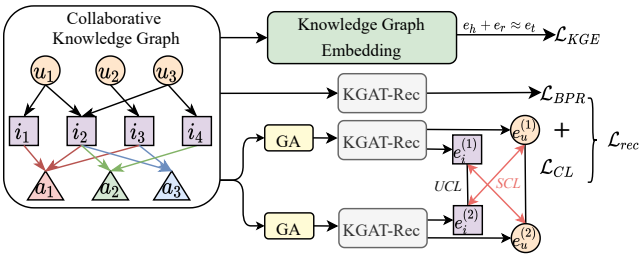


Fig. 1. Framework of our proposed C-KGAT, where GA and KGAT-Rec indicates graph augmentation and knowledge graph attention-based recommender, respectively.

3.1. Knowledge Graph Embedding

Each user, recipe, and attribute entity is associated with an ID embedding, annotated by e_u, e_i, and e_a, respectively, which are used to initialize their entity embeddings in the collaborative knowledge graph. We adopt TransE [12] that projects head, relation, and tail into the same embedding space and model the relations as translations in the embedding space with the constraint of $e_h + e_r \approx e_t$. A ranking loss is employed to prioritize an observed triplet with higher score than a randomly chosen negative triplet:

$$\mathcal{L}_{\text{KGE}} = \sum_{(h,r,t,t') \in \mathcal{T}} -\ln \sigma \left(g\left(h, r, t'\right) - g(h, r, t) \right), \quad (1)$$

$$g(h, r, t) = \left\| e_h + e_r - e_t \right\|_2^2 . \quad (2)$$

where $\mathcal{T} = \{(h, r, t, t') \mid (h, r, t) \in \mathcal{G}, (h, r, t') \notin \mathcal{G}\}$ and t' is a randomly sampled entity to build the negative pair; σ is the sigmoid function.

3.2. KGAT-based Recipe Recommendation (KGAT-Rec)

In the collaborative knowledge graph, an entity embedding is updated by aggregating the rich semantic information from its neighbor triplets. As shown in Figure 2, by attending the target entity e_h to its neighbor $\mathcal{N}_h$ with the relational attention

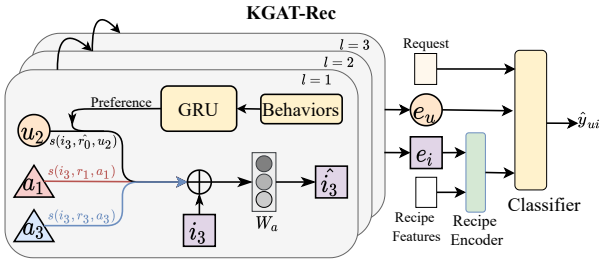


Fig. 2. Illustration of the KGAT-based Recipe Recommender.

function $s(h, r, t)$, the normalized attentive weights are generated to measure the contribution of each neighbor triplet:

$$s(h, r, t) = (e_t)^\top \tanh(e_h + e_r), \quad (3)$$

$$\alpha_{h,r,t} = \frac{\exp(s(h, r, t))}{\sum_{(h,r',t') \in \mathcal{N}_h} \exp\left(s\left(h, r', t'\right)\right)}, \quad (4)$$

$$e_{\mathcal{N}_h} = \sum_{(h,r,t) \in \mathcal{N}_h} \alpha_{h,r,t} e_t . \quad (5)$$

However, existing methods [10, 11] treat all user-recipe interactions the same, failing to differentiate user preferences on different recipes. For example, some users show less interest by browsing recipe casually and checking key ingredients, while some users cook along with a recipe step by step, which reflects stronger user preferences. To alleviate the pain point, we propose to learn different user preferences from her sequential behaviors towards each interacted recipe by a one-layer bidirectional Gated Recurrent Unit (GRU) [13] as:

$$h_{pref} = GRU([b_1, b_2, ..., b_k]), \quad (6)$$

$$e_r^* = W_p([e_r, h_{pref}]). \quad (7)$$

where $[b_1, b_2, ..., b_k]$ are the user sequential behaviors, represented by their behavior type embeddings. The learnt preference vector h_{pref} (the last hidden state in GRU) is concatenated with the uniform edge embedding e_r and passed into a fully-connected layer to obtain the preference-aware edge embedding e_r^*, which replaces e_r in Equation 3 to compute the attentive weights for the edges connecting users and recipes; while for edges between recipes and attributes entities, we still learn the attentive weights with original Equation 3. By doing so, the model is capable of learning better user and recipe embeddings for recommendation by measuring the impact of each neighbor triplet more accurately.

We update the target entity embedding $\hat{e_h}$ by aggregating its entity embedding e_h and embeddings of its neighbors with LeakyReLU [14] as the non-linear function:

$$\hat{e_h} = \text{LeakyReLU}\left(W_a \left(e_h + e_{\mathcal{N}_h}\right)\right) \quad (8)$$

We stack l layers of knowledge graph attention to capture l-order proximity among entities. The user and recipe collaborative vector $\hat{e_u}$ and $\hat{e_i}$ are obtained by concatenating the user and recipe representations from each layer. We then adopt BERT (frozen during training) to obtain user request

vector h_{req} by taking the first hidden vector corresponding to the [CLS] token [15]. Finally, a classifier f_c takes as input request vector, user vector, and recipe vector to predict probability logits $\hat{y}_{ui}$. The model is trained with the BPR loss [4] that an observed interaction (u, i) has a higher score than its sampled unobserved counterpart (u, j):

$$\mathcal{L}_{\text{BPR}} = \sum_{(u,i)\in\mathcal{Y}^+,(u,j)\notin\mathcal{Y}^+} -\log\sigma\left(\hat{y}_{ui} - \hat{y}_{uj}\right), \quad (9)$$

$$\hat{y}_{ui} = f_c(h_{req}, \hat{e_u}, f_{enc}(\hat{e_i}, e_f)). \quad (10)$$

where e_f is the recipe feature embedding by concatenating the textual representation and categorical feature embeddings. f_{enc} is a MLP-based recipe encoder that combine the collaborative recipe embedding and recipe content representation.

3.3. Incorporating Contrastive Learning

3.3.1. Graph Augmentation

In the graph augmentation (GA) stage, different views of the input graph are generated to expose novel patterns of representations to improve the model generalization [16, 17, 18].

In this work, we present graph augmentation strategies to generate different graph views that contain incomplete information about *node embedding* and *node topology* to boost downstream contrastive learning. Essentially, we apply the same following operations on the input CKG graph $\mathcal{G}$ independently twice to generate two different graph views $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$: (1)**Node Embedding Dropout**: We randomly set some embedding values as zero with a prior probability p to create graph views that contain incomplete node embedding. (2) **Edge Dropout**: To generate incomplete node topology information, we randomly discard some edges with the same p.

3.3.2. Unsupervised Contrastive Learning

Recent works explore unsupervised contrastive learning (UCL) on graph to alleviate the label-sparsity issue and improve model robustness [16, 19, 20] given a pair of generated graph views. In detail, the user node representations $e_u^{(1)}$ and $e_u^{(2)}$ are obtained by passing two generated views, $\mathcal{G}^{(1)}$ and $\mathcal{G}^{(2)}$, into the same KGAT model. Then, InfoNCE loss is adopted [21] to pull the different views of the same user entity close and push those of different user entities away:

$$\mathcal{L}_{UCL}(\mathcal{U}^{(1)},\mathcal{U}^{(2)}) = \sum_{u\in\mathcal{U}} -\log\frac{\exp\left(f\left(e_u^{(1)}, e_u^{(2)}\right)/\tau\right)}{\sum_{v\in\mathcal{U}}\exp\left(f\left(e_u^{(1)}, e_v^{(2)}\right)/\tau\right)} \quad (11)$$

where $f(\cdot,\cdot)$ indicates the cosine similarity function. Different views of the same user ($e_u^{(1)}$, $e_u^{(2)}$) are considered as a positive pair, while views of different user nodes ($e_u^{(1)}$, $e_v^{(2)}$) are viewed as a negative pair. τ is the temperature hyper-parameter. We obtain recipe-side unsupervised contrastive learning loss $\mathcal{L}_{UCL}(\mathcal{I}^{(1)},\mathcal{I}^{(2)})$ similarly.

3.3.3. Supervised Contrastive Learning

As demonstrated in Figure 1, in addition to unsupervised contrastive learning on $(e_u^{(1)}, e_u^{(2)})$ and $(e_i^{(1)}, e_i^{(2)})$, we propose to perform supervised contrastive learning (SCL) on the available user-recipe interactions $(e_u^{(2)}, e_i^{(1)})$, enforcing the model to effectively leverage the user-recipe proximity across different incomplete graph views, improving the robustness and generalization of the model. Given an observed user-recipe interaction y_{ui}, we encourage the agreement between the user representation $e_u^{(2)}$ and the recipe representation $e_i^{(1)}$ generated from different views. Meanwhile, we minimize the agreement between unobserved user-recipe pairs $(e_u^{(2)}, e_q^{(1)})$ by sampling recipes for user u:

$$\mathcal{L}_{SCL}(\mathcal{U}^{(2)},\mathcal{I}^{(1)}) = \sum_{(u,i)\in\mathcal{Y}^+} -\log\frac{\exp\left(f\left(e_u^{(2)}, e_i^{(1)}\right)/\tau\right)}{\sum_{q\in\mathcal{Q}}\exp\left(f\left(e_u^{(2)}, e_q^{(1)}\right)/\tau\right)} \quad (12)$$

where $\mathcal{Q}$ indicates the recipe set that includes one observed recipe and sampled unobserved recipes for user u.

The total CL loss is the summation of symmetrical unsupervised contrastive learning on user nodes and recipe nodes and supervised contrastive learning:

$$\begin{aligned}\mathcal{L}_{CL} =& \mathcal{L}_{UCL}(\mathcal{U}^{(1)},\mathcal{U}^{(2)}) + \mathcal{L}_{UCL}(\mathcal{U}^{(2)},\mathcal{U}^{(1)}) \\ &+\mathcal{L}_{UCL}(\mathcal{I}^{(1)},\mathcal{I}^{(2)}) + \mathcal{L}_{UCL}(\mathcal{I}^{(2)},\mathcal{I}^{(1)}) \\ &+\mathcal{L}_{SCL}(\mathcal{U}^{(1)},\mathcal{I}^{(2)}) + \mathcal{L}_{SCL}(\mathcal{U}^{(2)},\mathcal{I}^{(1)}) \end{aligned} \quad (13)$$

The model is trained by alternatively minimizing the final recommendation loss $\mathcal{L}_{rec} = \mathcal{L}_{BPR} + \lambda\mathcal{L}_{CL}$ and KGE loss L_{KGE} during each epoch.

4. EXPERIMENTS

To validate the superiority of our proposed model, we conduct extensive experiments to answer the following research questions: (**RQ1**) How does the proposed model perform compared with other SOTA models? (**RQ2**) How do the components in the proposed model improve the performance for recipe recommendation? (**RQ3**) What benefits does our proposed model bring for recipe recommendation?

4.1. Datasets

Table 1. Statistics of the datasets.

		Recipe-Voice	Recipe-Touch
Recipe	#Entity	50,070	18,740
Knowledge	#Relation Types	4	4
Graph	#Triplets	1,486,858	518,735

Customers can interact with devices equipped with screens by vocal request in Recipe-Voice dataset and touching the screen in Recipe-Touch dataset. To ensure the data quality, we take the $3(10)$-core subset for the two datasets, where each user or recipe has at least 3 (10) interactions, respectively. All users in the datasets are de-identified. The statistics of the recipe knowledge graph associated with the datasets are reported in Table 1. We sort the interactions of each user chronologically and split the interactions into training, validation, and testing set with ratios of 70%, 10%, and 20%, respectively. During testing, given a user's request, the model recommends top-K recipes by ranking the predicted probability of each recipe.

4.2. Baselines

In the experiment, we mainly adopt two types of content-based CF models as baselines for overall performance comparison: (1) Non graph-based CF: **YoutubeDNN** [2] utilizes DNN model to learn the recipe vector by concatenating ID embedding and the textual and categorical features; (2) graph-based CF: **LightGCN** [7] is a state-of-the-art model that learns user and recipe representation with a light graph convolutional operation. **KGCN** [11] runs on the subgraphs sampled from the collaborative knowledge graph to learn user and recipe recipe embeddings, and **KGAT** [10] learns user and recipe embeddings with knowledge graph attention mechanism.

4.3. Main Results (RQ1)

Table 2. Overall Model Performance Comparison.

Datasets	Recipe-Voice		Recipe-Touch	
Models	Recall@5	NDCG@5	Recall@5	NDCG@5
YoutubeDNN	-5.5%	-3.0%	-3.9%	-4.1%
LightGCN	-4.8%	-2.1%	-1.5%	-2.4%
KGCN	-2.5%	-1.3%	-0.8%	-1.5%
KGAT	0.0%	0.0%	0.0%	0.0%
C-KGAT	**+5.2%**	**+7.4%**	**+4.9%**	**+5.8%**

We first compare performance of different models on testing set in terms of widely-used metrics for top-k recommendation, including Recall@K and NDCG@K [22], where $K = 5$. Table 2 shows the relative performance improvement afforded by our method compared to state-of-the-art models. Looking at the results, LightGCN achieves better results than YoutubeDNN by capturing higher-order proximity among users and recipes. KGCN and KGAT outperform other baselines by leveraging knowledge graph that alleviates label sparsity issue. The proposed model consistently yields the best performances on both datasets due to its capability of modeling diversified user's preferences over different recipes given her sequential behaviors towards recipe and leveraging contrastive learning effectively. We further conduct significance testing (t-test) on the improvements of our approaches with fives runs and obtain $p < 0.01$ compared with the best-performed KGAT in terms of Recall@5/NDCG@5 on both datasets, respectively, demonstrating the significance of the improvement.

4.4. Ablation Study (RQ2)

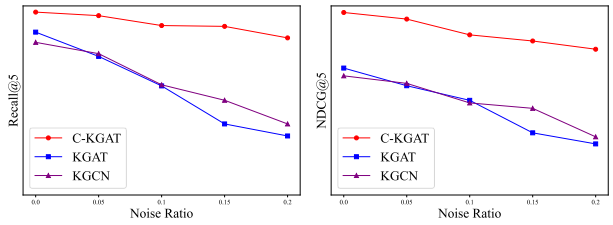
We further conduct ablation study to quantify the impact of the components in our proposed model and report the corresponding degradations in Table 3, including the GRU for preference vector, contrastive learning (CL) that includes both UCL and SCL, supervised contrastive learning (SCL), node embedding dropout (NED), and edge dropout (ED). Each proposed component contributes in a certain degree to the improvement of model performance, which demonstrates the reasonability of our proposed model. Taking a close look at the results, removing CL hurts the performance the most, which verifies the effectiveness of contrastive learning.

Table 3. Ablation Study.

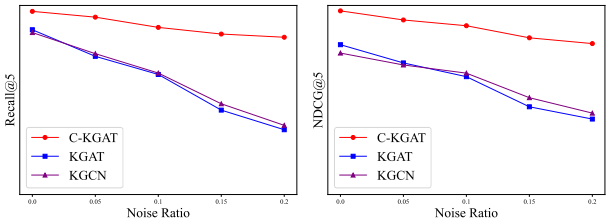
Datasets	Recipe-Voice		Recipe-Touch	
Models	Recall@5	NDCG@5	Recall@5	NDCG@5
C-KGAT	0.0%	0.0%	0.0%	0.0%
-GRU	-2.8%	-4.2%	-1.4%	-2.0%
-CL	-4.1%	-6.1%	-2.7%	-3.1%
-SCL	-2.1%	-3.2%	-1.3%	-1.8%
-NED	-1.4%	-2.6%	-0.9%	-1.4%
-ED	-1.0%	-2.1%	-0.6%	-0.9%

4.5. Benefits of the Proposed Model (RQ3)

As mentioned above, contrastive learning is expected to improve model robustness by learning robust embeddings from different incomplete graph views. To validate this, we train models with different ratios of additional noise data sampled from unobserved interactions and compare their performances as plotted in Figure 3. Injecting additional noise data into training set does harm model performance on different levels, while the proposed C-KGAT has a much less degradation rate than the two strongest baselines, demonstrating the superiority of the proposed model in terms of robustness.



(a) On Recipe-Voice dataset.



(b) On Recipe-Touch dataset.

Fig. 3. Model performance with different noise ratios.

5. CONCLUSION

We propose a contrastive knowledge graph attention network for user request-based recipe recommendation. The proposed model not only boosts performance by modeling user preferences towards different recipes but also integrates unsupervised and supervised contrastive learning to improve model robustness. In the future, we plan to improve the model performance with advanced negative sampling strategies and transfer learning for cross-domain recommendation.

6. REFERENCES

[1] Heng-Tze Cheng, L. Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, H. Aradhye, Glen Anderson, G. Corrado, Wei Chai, M. Ispir, Rohan Anil, Zakaria Haque, L. Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah, "Wide & deep learning for recommender systems," *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, 2016.

[2] Paul Covington, Jay K. Adams, and Emre Sargin, "Deep neural networks for youtube recommendations," *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016.

[3] F. Ricci, L. Rokach, and Bracha Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*, 2011.

[4] S. Rendle, C. Freudenthaler, Zeno Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI*, 2009.

[5] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua, "Neural collaborative filtering," *Proceedings of the 26th International Conference on World Wide Web*, 2017.

[6] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua, "Neural graph collaborative filtering," *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019.

[7] Xiangnan He, Kuan Deng, X. Wang, Y. Li, Yongdong Zhang, and Meng Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

[8] Yu Liu, Shuai Wang, M. S. Khan, and Jieyue He, "A novel deep hybrid recommender system based on auto-encoder with neural collaborative filtering," *Big Data Min. Anal.*, vol. 1, pp. 211–221, 2018.

[9] Javier Maroto, Cl'ement Vignac, and P. Frossard, "Modurec: Recommender systems with feature and time modulation," *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 3615–3619, 2021.

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

[11] Hongwei Wang, M. Zhao, Xing Xie, Wenjie Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," *The World Wide Web Conference*, 2019.

[12] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, J. Weston, and Oksana Yakhnenko, "Translating embeddings for modeling multi-relational data," in *NIPS*, 2013.

[13] Kyunghyun Cho, B. V. Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *EMNLP*, 2014.

[14] Bing Xu, Naiyan Wang, Tianqi Chen, and Mu Li, "Empirical evaluation of rectified activations in convolutional network," *ArXiv*, vol. abs/1505.00853, 2015.

[15] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[16] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie, "Self-supervised graph learning for recommendation," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021, pp. 726–735.

[17] Xiao Wang and Guo-Jun Qi, "Contrastive learning with stronger augmentations," *ArXiv*, vol. abs/2104.07713, 2021.

[18] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick, "Momentum contrast for unsupervised visual representation learning," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9726–9735, 2020.

[19] Yinwei Wei, Xiang Wang, Qi Li, Liqiang Nie, Yan Li, Xuanping Li, and Tat-Seng Chua, "Contrastive learning for cold-start recommendation," *arXiv preprint arXiv:2107.05315*, 2021.

[20] Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong, "Contrastive learning for recommender system," *arXiv preprint arXiv:2101.01317*, 2021.

[21] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *AISTATS*, 2010.

[22] K. Järvelin and Jaana Kekäläinen, "Ir evaluation methods for retrieving highly relevant documents," *SIGIR Forum*, vol. 51, pp. 243–250, 2017.