Conversational Recommender Systems

Week 9 Lecturers: Henghchang Hu, Victor Li, Jiaqi Zheng, Jiahao Liu, Hansheng Ren
Week9 Authors: Ran Yiding,Sunil Kumar Yadav,Zijin Kong, Yajing Yang, Jiaqi Zheng
Week 10 Lecturers: Sunil Kumar Yadav
Week10 Authors:Xu Yiqing, Lu Xinyuan, Chen Zitai, Ren Hansheng

Slide Link: Conversational Recommender Systems Week 9
Scribe: Conversational Recommender Systems Week 9 and 10
Other documents in this series at http://bit.ly/cs6101-2010-notes

Week 9 Papers:

1. Q&R: A Two-Stage Approach toward
Interactive Recommendation http://alexbeutel.com/papers/q-and-r-kdd2018.pdf
2. Towards Question-based Recommender Systems (SIGIR' 20) https://arxiv.org/pdf/2005.14255.pdf
3. Estimation-Action-Reflection: Towards Deep Interaction Between Conversational and Recommender Systems. (https://dl.acm.org/doi/pdf/10.1145/3336191.3371769)
4. Dynamic Online Conversation Recommendation, ACL 2020 http://www4.comp.polyu.edu.hk/~jing1li/publication/zeng2020dynamic.pdf
5. Chen, Z.(2020). Towards Explainable Conversational Recommendation. IJCAI. Chicago. https://www.ijcai.org/Proceedings/2020/0414.pdf

Week 9

Introduction to conversational recommendation system (CRS):
What is CRS?
- Generate more and items after conversation with

Why CRS:
- Information asymmetry
    - Gap between training data VS real time data after deployed online
    - Cold start problem: encountered when recommending to new user without historical interaction
    - Shift of user preference over time

CRS as intersection of many disciplines:
- Traditional RS: FM, CF, neural variants
- Dialogue system: pipeline design, end2end neural models
- Human-computer interactions
- User simulations: need to be fast, broad coverage and unbiased

Overview of today's reading group:

- Direction1: question-driven approach
    - Ask better question to get users' preference
- Direction2: multi-turn CRS strategy
    - Increase efficiency in asking the right question to get users' preference
- Direction3: Dialogue understanding and generation
    - Understand users' intent and preference through natural language

Question Based Recommender Systems

Paper 1: A Two Stage Approach towards Interactive Recommendation

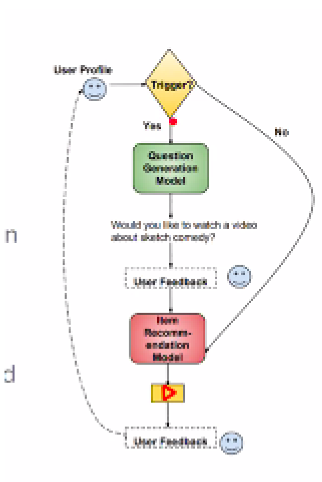
Approach:

Question and recommendation consists of two components

- Question asking component
- Item recommendation component
    - Video recommendation model used in this paper

One round of the conversation between RS & user

- Decide what to ask user
- Decide how to adapt the response and change its model about the user according to the user's provided feedback
    - User feedback used to enhance the recommendation component to generate more accurate recommendations

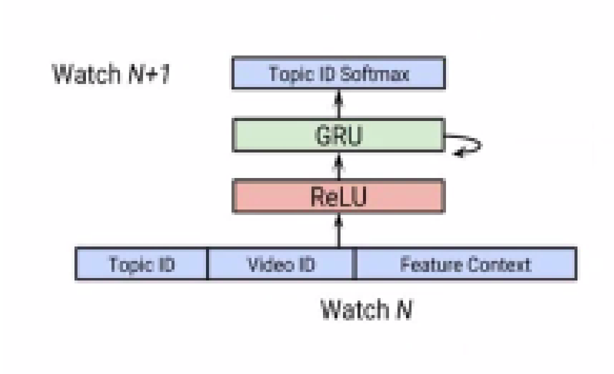


Task:

What is the topic of the next video a user would want to watch
- Capture user interest
- Ask question to cover large interest space
- Data:

Given the most relevant topic of the to-be-watched video, what video will the user be most interested in?

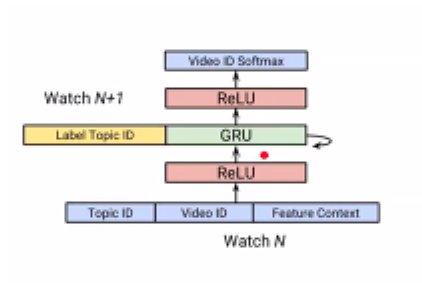
Model-question ranking model



Input: past watching history up to T
Output:
Training
Inference:

Model-video response model:
- Imagine user has selected the topic q they are interested in
- Item recommendations

Model-video response model



Input:
- Watch history
- Topic information as post-fusion

Training:
Inference:

Experiments-offline comparative analysis

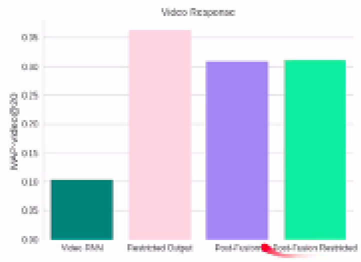


Figure 5: MAP-video@20 video response results.

- Compare different models
- Metric: MAP@k -> mean over the average precisions of the users

Experiments on YouTube

- Results on Youtube homepage: production baseline: a highly optimized baselines, including RNNs

Experiments on User Onboarding

Conclusion:

- First work on learned interactive recommendation demonstrated in a large-scale industrial setting

Discussion:

- Advantages:
    - Large-scale learned interactive RS
    - RNN-based two-factored recommendation

Disadvantages

Important to talk about the trigger

- Based on the trigger to decide whether to ask or recommend
    - System taking initiative to recommend to ask clarification question

  Factors can be used to make the decision
- Whether system is confident in recommending
    - If system is not reaching the confidence level -> continue asking
    - Trade off between exploration and exploitation -> pursue optimal policy or optimal policy is not informed by evidence
- More can be found in EAR paper

Baseline

- Important to introduce baseline in the paper
- Can be a warning sign if baseline is not included
    - New area baseline not existing
    - Not credible enough is baseline is not included -> see whether can replicate the results

- Potential baseline:
    - always recommend (i.e. ignore trigger)
    - CRS (not including conversation aspect) -> conversation is treated as a novel thing to be included in RS

Paper 2: Towards question-based recommender systems

Problems Formalisation:

Historical Data - Conversation - Recommendation

Receive feedback from user to update -> more than one round can exist
-> ask questions to narrow down the recommended scope

WIth fixed #questions, agent expected to give recommendations asap
- From historical data: not sure on specific preference

Framework of Qrec

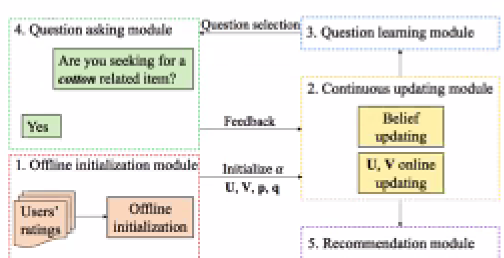
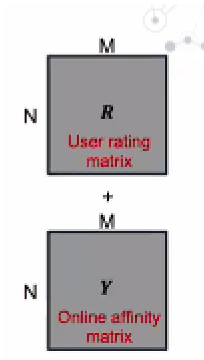


Figure 1: Framework of our proposed question-based recommendation model, Qrec. *Cotton* is an extracted entity (informative term), U, V, p, q are model variables, and α is a hyper-parameter of user belief.

components:
Latent factor building
Question generation
Final recommender output

How to adjust strategy round by round

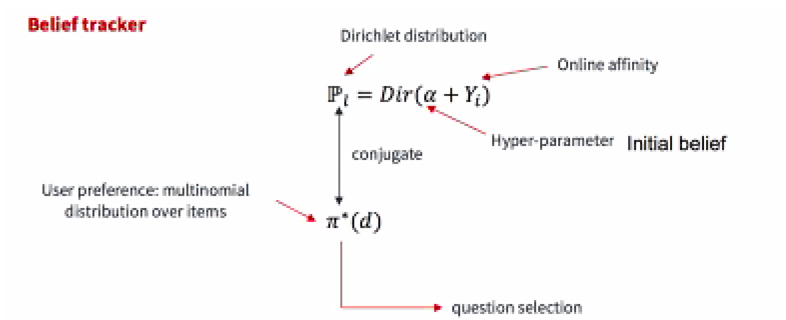
Latent factor recommendation:
- Probabilistic MF
    - The observation noise is Gaussian distributed
    - User attribute U and item attribute V
- Problem statement

- 

- R: historical interaction
- N: online feedback from conversation

Loss function:
Rating matrix error + online affinity matrix error
Find parameters by minimizing loss function

*Question generation:
Question asked is template-based
Attribute choosing criteria
Assume a prior belief P (Dirichlet distribution) over the user preference distribution



- Initial belief calculated from offline matrix

Select entity -> narrow down scope
- Based on entity and user preference
- Min |sum(entity * user preference|
  Used general idea of binary search

Answer: yes/no/unsure

Experiment
- Dataset: 2 from Amazon
- Give similar results

- Evaluation:
    - simulating users: assume users with full knowledge
        - Question generated based on preference

    -
- Performance comparison
    - Static baseline s
    - Offline MF in Qrec
    - Interaction baselines
    - Highest result achieved when compared with interactive vaselines
    - Performs better than NMF with less than 5 questions

    - Cold start performance analysis
        - Still high performance even for cold start user/item

    - Contribution of offline initialization

    - Online user study
        - 489 conversations collected, 21 crowd workers on 33 target items
        - #questions actual user willing to answer: 15
        - Results in agreement with simulated user

Conclusion
- Novelty: incorporating offline and online MF training
    - Useful for other works as well

Q&A
VS paper1
- Incorporate user feedback directly into recommendation generation
- MF: transparent algo -> directly update latent factor representation/distributional parameters online
    - Update user belief on item

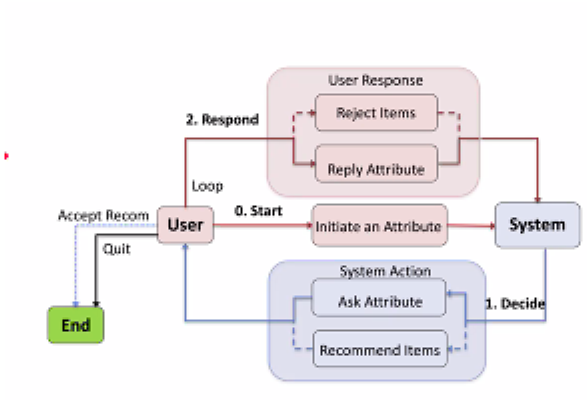
RS: think it as search engine give you recommendations
- Main difference: search engine is not taking into consideration historical search while recommendation system relies heavily on historical interaction
    - observed query: Latent need/intention
- RS: users are not giving query
- Shift of control of information gathering
- Do we end up with a bossy engine that tells you what to do or does the user take the control back?
    - How to find the balance

So far not much about NLP processing in RS

Multi Turn Conversational Strategy

Paper 3: Estimation-Action-Reflection: Towards Deep Interaction Between Conversational and Recommender Systems

Multi-round scenario: ask or recommend



- When system does not get enough information -> recommendation might be rejected
- Too many questions asked -> impatient user
  Balance is important

Fundamental problems in multi-round CRS
What attribute to ask
- What question to ask to shorten rounds of questions

When to recommend items
- How can we know the proper time to push recommendations
- If the candidate space is small enough -> recommend
- If asking additional questions is not useful -> recommend

How to adapt to users' online feedback
- yes/no to the queried attribute
- accept/reject to the recommended item

Approach:
- Conversational component (CC), recommender component (RC)
- 3 stages
    - Estimation: build predictive models to estimate user preference on both items and attributes
    - Action: dialogue policy to determine whether to ask or recommend based on Estimation stage and conversation history
    - Reflection: updates recommender when user rejects recommendation

Estimation:

- Estimate user preference on both items and attributes
- Predictive model used: FM
    - $$\hat{y}(u, v, \mathcal{P}_u) = \mathbf{u}^T \mathbf{v} + \sum_{p_i \in \mathcal{P}_u} \mathbf{v}^T \mathbf{p}_i$$
    - User embedding item embedding
    - Trained using Pairwise Bayesian personalized ranking (BPR) objective
    - $$L_{bpr} = \sum_{(u,v,v') \in \mathcal{D}_1} -\ln\sigma(\hat{y}(u, v, \mathcal{P}_u) - \hat{y}(u, v', \mathcal{P}_u)) + \lambda_\Theta \|\Theta\|^2$$
      $$\mathcal{D}_1 := \{(u, v, v') \mid v' \in \mathcal{V}_u^-\}$$
      $$\mathcal{V}_u^- := \mathcal{V} \backslash \mathcal{V}_u^+$$
    - V’: non-interacted item
    - V: interacted item
        - Traditional: score of v > score of v’
    - Attribute-aware BPR for item prediction.
      $$\mathcal{V}_u^- := \mathcal{V} \backslash \mathcal{V}_u^+ \qquad \widehat{\mathcal{V}}_u^- := \mathcal{V}_{cand} \backslash \mathcal{V}_u^+$$
      $$\mathcal{D}_1 := \{(u, v, v') \mid v' \in \mathcal{V}_u^-\} \qquad \mathcal{D}_2 := \{(u, v, v') \mid v' \in \widehat{\mathcal{V}}_u^-\}$$
      The new loss function for items:
      $$L_{item} = \sum_{(u,v,v') \in \mathcal{D}_1} -\ln\sigma(\hat{y}(u, v, \mathcal{P}_u) - \hat{y}(u, v', \mathcal{P}_u)) + \sum_{(u,v,v') \in \mathcal{D}_2} -\ln\sigma(\hat{y}(u, v, \mathcal{P}_u) - \hat{y}(u, v', \mathcal{P}_u)) + \lambda_\Theta \|\Theta\|^2$$
    - This model: incorporate attribute-aware
    - Attribute preference will be asked -> need to train as well
    - Attribute Preference Prediction.
      $$\hat{g}(p \mid u, \mathcal{P}_u) = \mathbf{u}^T \mathbf{p} + \sum_{p_i \in \mathcal{P}_u} \mathbf{p}^T \mathbf{p}_i$$
      The loss function for attribute:
      $$L_{attr} = \sum_{(u,p,p') \in \mathcal{D}_3} -\ln\sigma(\hat{g}(p \mid u, \mathcal{P}_u) - \hat{g}(p' \mid u, \mathcal{P}_u)) + \lambda_\Theta \|\Theta\|^2$$
      Multi-task Training:
      $$L = L_{item} + L_{attr}$$
    - Attribute of ground truth should be ranked higher
    - When to recommend: Reinforcement learning model

Action

Reflection

- If recommendation rejected: assign lower score of those item
    - Update the recommender component
- Optimize BPR lss online

Experiment

	LastFM		Yelp	
	SR@15	**AT**	**SR@15**	**AT**
Abs Greedy	0.209	13.63	0.271	12.26
Max Entropy	0.290	13.61	0.919	5.77
CRM	0.325	13.43	0.923	5.33
EAR	**0.429***	**12.45***	**0.971***	**4.71***

- Dataset:
    - Yelp: business recommendation
    - LastFM: music artist recommendation
- Metric:
    - SR@t: ratio of successful conversation by turn t
    - AT: average turns needed to end the session
- Baseline:
    - Abs greedy
    - Max entropy: push recommendation when candidate space is small
    - CRM: similar to EAR, state of the art
- Performance of EAR:
    - Significant improvement in both metric
- Abs greedy is better at the beginning as other approaches ask question at the beginning
    - Gradually fall behind after several rounds
    - Asking question is beneficial in multi-round setting
- EAR is more efficient in a complicated multi-round scenario

Q&A

- The most important/interesting idea from this paper VS other papers?
    - [Answered by primary author Yisong] Recently the approach is tried on Kuaishou (Chinese video sharing mobile app) clear problem formalization: what to ask, when to recommend and how to adapt to feedback (favored by biz partner)
- When to ask: are there instances where the RS is like a pushy person to just recommend without asking more
    - Baseline: abs greedy works in a similar way
    - For EAR, it appears to be more patient, ask 2-3 questions before recommending
    - Stakes differ across scenario: restaurant recommendation VS date recommendation
        - Dynamic of human environment
        - Possible recommendation agent runs into user preference agent
            - How much of user preference gets into recommendation system
- User simulation: people with different level of patience -> distribution of patience level can be added in when simulating users
    - Model it as a latent variable
    - Paper published by Baidu in ACL 2020: when to stop, regulate #rounds
    - context /platform matters as well: video on Youtube VS buying an expensive item

Paper 4: Dynamic Online Conversation Recommendation

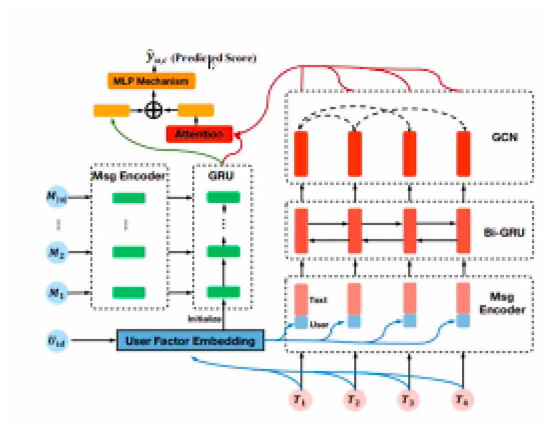
Motivation

- Trending content on social media involves over time -> crucial to understand social media users and their interpersonal communications in a dynamic manner
- Existing work in CRS assume static user interests
- Better handle cold start problem, where conversation and users are new and unseen in training

Approach:

- Capture user interests from both what they said in the past and how they interacted with each other in conversation structure
    - Capture time-variant representations from user chatting history
    - Model user interaction in the conversation context
    - Propose a user-aware attention to convert dynamics of user interest
    - Recommend

Overall structure



- Model user interest dynamics, together with conversation representations derived are used to produce final prediction
- Prediction score y(u,c): how likely you will engage in c
- Msg encoder: mainly contains two layers: word embedding layer and CNN modeling layer

Approach:

- Message-level modeling
    - Given u's historical message m, first use a pre-trained word embedding layer to map word to vector space -> employ convolutional NN encoder to model
- User-aware conversation modeling
    - Turn t: in form of continuous word sequence w in one author id
    - Encode work occurrence in each turn via turn-level modeling
    - Encode interaction between conversation turns
    - User-ware attention over turn s

        - Conversation-level modeling
            - Chronological order
            - Replying structure
- Experiments
    - Training: Jan - April
    - Validation + testing: May
    - Simple baselines:
        - Popularity
        - Topicrank
    - Conversation cold start
        - Separate test set into future conversations and existing ones
        - Baseline: CRIM, LC-RNN
    - Ablation study
        - User factor embedding and user-ware attention contribute most to model outputs
        -

Core innovation:

- When users change their mind, a dynamic system will be particularly helpful
    - Different from other e-commerce directed recommendations
    - Changes: new participants, new subjects -> dynamic interest
    - Topic detection and dynamic shift are important

Paper 5: Chen, Z.(2020). Towards Explainable Conversational Recommendation

Introduction

- Explanation in RS
    - RS predicts personalized preference s
    - Focus on accuracy and explainability
    - Explainability -> user satisfaction, trust
    - Trigger feedback
    - Help user understand working mechanisms
    - Tune RS: why recommendation is wrong, immediate updates and communication, interaction
    - More data collection for future
- Integrate user feedback and
- Provide alternative interaction paradigm: what - whether

Problem Formulation

- Explanations are given to model during training
- Concepts, Explanations and Feedbacks are critical parts of ERS
- Negative feedback is last recommendation

Model Description

- [Insert Figure]
- Context aware concept embedding
- Co-Attentive Concept Importance Modeling [Insert equations]
    - Co attentive weighting matrix
    - User concept importance vector
    - Concept level feedback
- Local Propagation of user concept interest
    - Aggregate embedding
    - Preference score beta
    - Choose the local concept importance
- Multi view concept selection
    - Global FM based recomm: based on user feedback
    - Local estimation of useritem interest
    - Multi view combination:
    - Constrained explanation generation
        - Select concept must appear in explanations using two GRUS
        - Concept relevance loss: punish the model if other selected concepts are not selected

Results:

- Outperforms other models [Fill details]

Week 10

Week 10 Papers:

1) Zhang et. al., 2020, Conversational Contextual Bandit: Algorithm and Application https://arxiv.org/abs/1906.01219
2) Lei et. al, 2020, "Interactive Path Reasoning on Graph for Conversational Recommendation", https://arxiv.org/pdf/2007.00194.pdf
3) Liu et. al, 2020, "Towards Conversational Recommendation over Multi-Type Dialogs", (https://arxiv.org/pdf/2005.03954.pdf)
4) Chen et al., 2019, "Towards Knowledge-Based Recommender Dialog System", https://www.aclweb.org/anthology/D19-1189.pdf

Paper 1: Conversational Contextual Bandit: Algorithm and Application

1/ Classical Bandit Framework

Multi arm bandit problem: A gambler chooses the slot machine which will maximise the rewards out of K **non identical** slot machines in the least amount of time.

Goal: find the best arm as soon as possible and keep gambling with the arm.

Connect to NLP => Contextual bandit problem: Item recommendation

Goal: to learn item recommendation[arm selection] to optimize user's feedback in the long run.
Item [action] : arm in bandit
Contextual vector of arm: user features [shared] + item features[action specific]
Dilemma for Algorithm: leverage user's already known preference versus revealing unknown preferences: **Exploitation vs Exploration Dilemma**
Issue: **Extensive exploration** is needed to accumulate sufficient feedbacks

2/ Why is contextual bandit problem hard? => Exploitation vs Exploration Dilemma

Greedily select the optimal option based on existing knowledge
Vs
Explorating new actions (unknown optimality) in the hope that you can get better reward or knowledge to make better decision

3/ Problem Formulation:

Setting:

1/ Finite set of N arms: a_1, a_2, ……., a_N together forming set A

2/ Rounds T

3/ A_t is a subset of A given to the agent at round t.

4/ Each arm a from At has is associated with d dimensional contextual vector $x_{a,t}$ which contains user as well as item information at round t.

5/ An arm is chosen and reward is received, $r_{at,t}$

Goal:
minimize regret where regret is defined as cumulative difference of expected value of optimal reward and received reward till rounds T.

$$R(T) \triangleq \sum_{t=1}^{T} \left(\mathbb{E}[r_{a_t^*,t}] - \mathbb{E}[r_{a_t,t}] \right).$$

Challenge:
Agent needs to make a trade off between the best arm based on feedback versus the arms agent is unsure of. (E-E Dilemma)

4/ Proposed Upper Confidence Bound Algorithm:

Based on selected arms till now a_1, a_2 … a_{t-1} and received rewards $r_{a1,1}$...$r_{at-1,t-1}$, estimate rewards $R_{a,t}$ for arms and select a new arm. (Basic Algorithm)

$$a_t = \arg\max_{a \in \mathcal{A}_t} R_{a,t} + C_{a,t}$$

Where C_{a,t} is the confidence interval of the arm **a** at round **t**.

$$r_{a_t,t} = \boldsymbol{x}_{a_t,t}^T \boldsymbol{\theta} + \epsilon_t$$

Where the reward function takes different forms depending upon the algorithm. In LinUCB, above linear form is used and Θ is d dimensional parameter vector to be learned.

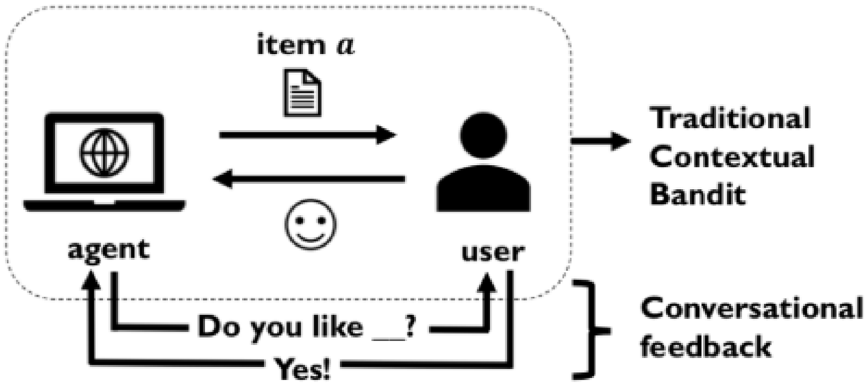


Figure 1: Conversational contextual bandit. The part in dashed box corresponds to traditional contextual bandit.

Key idea:
Conversational feedback is added to the problem to accelerate bandit learning.

But How do we decide when to converse?
=> conversation frequency model!

Idea: we only allow the system to make a fixed number of conversations for a number of rounds.

Details:

q(t) = 1 if b(t) - b(t-1) > 0 else q(t) = 0 where b(t) is the number of conversations upto round t.

Say, b(t) = k ceil(t/m), m >=1, k>=1 then the agent makes k conversations in every m rounds.

Also we assume key term level conversations are less than arm level interactions.

i.e. b(t) <= t

rk,tis the user's preference for key k at round t.

Overall algorithm:

Algorithm 1: General algorithm of ConUCB

Input: arms $\mathcal{A}$, key-terms $\mathcal{K}$, graph $(\mathcal{A}, \mathcal{K}, \boldsymbol{W})$, $b(t)$.

1 **for** $t = 1, \ldots, T$ **do**

2 observe contextual vector $\boldsymbol{x}_{a,t}$ of each arm $a \in \mathcal{A}_t$;

3 If conversation is allowed at round t, i.e., $q(t) = 1$, select key-terms to conduct conversations and receive conversational feedbacks $\{\tilde{r}_{k,t}\}$;

4 select an arm $a_t = \arg\max_{a \in \mathcal{A}_t} \tilde{R}_{a,t} + C_{a,t}$;

5 receive a reward $r_{a_t,t}$;

6 update model ;

Two key components:
1/ key selection [line 3]
2/ arm selection [line 4]

Intuition:

1/ Key selection aims to ask relevant questions to gain more knowledge for (promising) exploration. Thus the loss for key selection tasks is to choose keys that are close match to the user’s current interest, with a regularization term.

2/ Arm selection balances the current optimal choice with the exploration term that depends on the answers to the key question [from the previous step]

3/ Theoretical guarantees are derived

5/ Experiments

Experiment 1: Synthetic
Each arm is associated with d dimensional feature vector xa and a set of key terms with equal weight.

x_a generation:

Pseudo feature vector x_{kdot} for each key term with each dimension from U(-1,1) iid.

For each arm sample, n_a key terms uniformly from K without replacement as its related key terms with weight $1/n_a$

Each dimension of x_a is drawn from ***N*** (mean value of x_{kdot}(i), sig^2_g)

N users with ground truth of user's preferences Θ drawn from U(-1,1)

True arm level reward and key level reward:

D:50, K:500, sig:0.1,M:5

Baselines: LinUCB, Arm-Con, Var-RS, Var-MRC, Var-LCR

$\mathbf{A_t}$ pool is selected randomly without replacement and presented to all algos.

10 times run and averaged.

Result:

Cumulative Regret is minimum in case of ConUCB for various factors.

CR increases with pool size as it becomes difficult to select the best arm with large pool size.

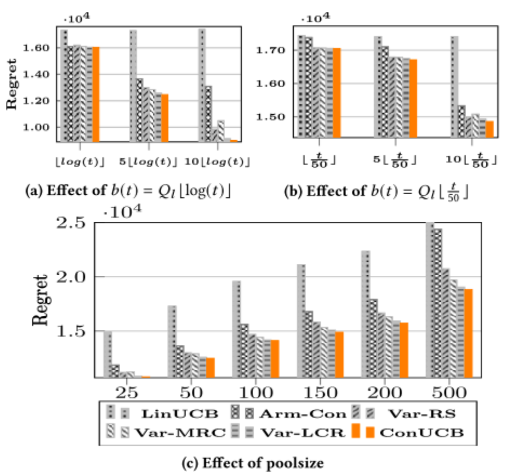


Figure 2: Effect of various factors: Figures (a) & (b) show the effect of $b(t)$; Figure (c) shows the effect of poolsize.

Experiment 2: Toutiao

2000 users , 1.7 million articles and 8.4 million interaction records.

Article is an arm

Categories: e.g. “Article”: {“news_car”} : 573

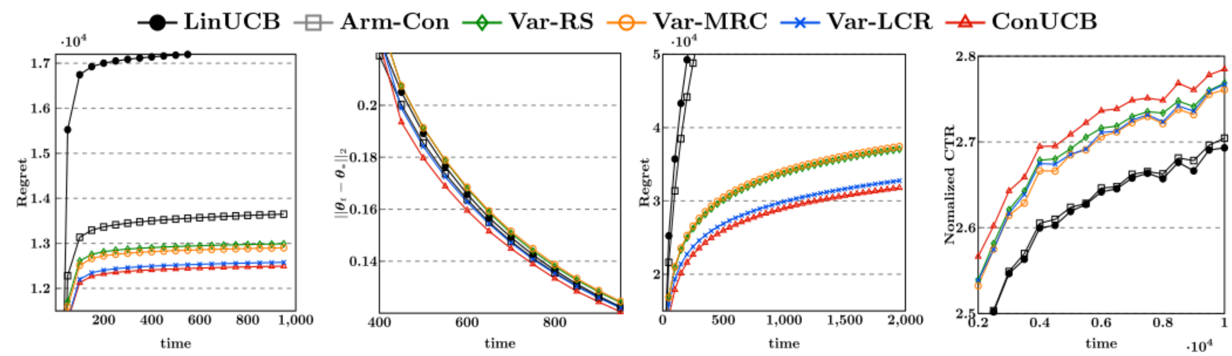
Keywords: 2384

Contextual vector is 100 dimensional [based on PCA on 3469 features]

User reads then feedback is 1 else 0, also taken as reward

User preference is generated based on interaction records [ridge regression]

Result:



(a) Cumulative regret on synthetic dataset (b) Accuracy of learned parameters (c) Cumulative regret on Yelp dataset (d) Normalized CTR on Toutiao dataset

6. Contribution

Conversational feedback based on key terms to accelerate bandit learning.

Paper 2: Interactive Path Reasoning on Graph for Conversational Recommendation

1/ Workflow of Multi-round Conversational Recommendation

- Session starts by user choosing a desired attribute
- The system decides whether to **ask** attribute or **recommend** items to user
- The user gives response
    - Reply when asking attribute
    - Accepting or rejecting the recommendation
- Repeat the second and third steps until:
    - Recommendation successful
    - User quit

2/ Objectives

There are 3 objectives of MCR

- Decides which attributes to ask
- Decides which item to recommend
- Decides what action will be performed (either asking or recommending)

3/ EAR, prior works of MCR

EAR divides their system into 2 parts,

- **Recommendation component (RC)**. Responsibility: decides items and attributes to recommend/ask.
- **Conversational component (CC)**. Responsibility: decides action to be performed

RC and CC are helping each other. CC uses statistics data from RC to choose the action, and RC uses the history of the dialog from CC to recommend/ask better items.

However, EAR has 2 limitations

- EAR considers a large number of actions space. The large number of actions space could make the policy model hard to train.
- EAR does not consider the structural information of user-item-attributes

4/ SCPR Framework and Model

At CPR, at each path/timestep, it employs 3 jobs:

- **Reasoning**. Scoring the candidate items and candidate attributes
- **Consultation**. An RL policy model to choose which action to perform at this time step.
- **Transition**. Updating the states (e.g the user-preferred-attirbutes, candidate items and candidate attributes) based upon the user feedback.

CPR Framework

- **Reasoning**
  - Item score:
    $$s_v = f(v, u, \mathcal{P}_u)$$
  - Attribute score:
    $$s_p = g(u, p, \mathcal{V}_{cand})$$
- **Consultation**
  - Policy network
- **Transition**
  - Extended path
    $$P = p_0, p_1, p_2 \dots p_t$$
  - Update candidate item/attribute

The difference between Simple CPR (SCPR) and the CPR framework is SCPR incorporates the knowledge-graph structures. SCPR uses the knowledge-graph structures for:

1. **Limiting** the candidate attributes.
   In EAR, they treat all the attributes as the candidate attributes. While in SCPR, it only considers the attributes which are **adjacent** with all the user-preferred attributes.
2. The reasoning scores.
   In SCPR, the scoring attributes strategy is to find attributes which eliminate the uncertainty of the candidate items. These candidate items are collected by using the knowledge graph.

   Message propagation from items to attributes

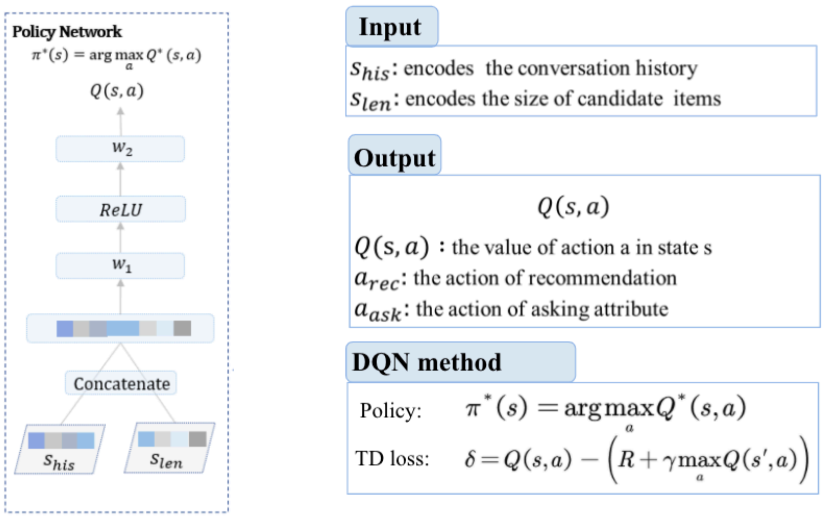
   Information entropy strategy

   - Weighted attribute information entropy

   $$g(u, p, \mathcal{V}_{cand}) = -\mathrm{prob}(p) \cdot \log_2(\mathrm{prob}(p)),$$

   $$\mathrm{prob}(p) = \frac{\sum_{v \in \mathcal{V}_{cand} \cap \mathcal{V}_p} \sigma(s_v)}{\sum_{v \in \mathcal{V}_{cand}} \sigma(s_v)}$$
3. The output action of RL Policy model.

In EAR, the output action of the policy function not only decides which action (ask/recommend), but also decides which attributes to ask. Thus output space is linear with the number of the available attributes. However in SCPR, the output space is a binary, which is whether to ask or recommend. Later, SCPR will rank the reasoning scores to decide which attributes to ask or which items to recommend.



5/ Experiments

Baselines:

- Max Entropy: rule-based
- Abs Greedy: only recommends
- CRM: state-of-the-art CRS
- EAR: state-of-the-art method on Multi-round CRS

Dataset:

- LastFM and Yelp. (merged the attributes into small numbers of attributes)
- LastFM* and Yelp* (do not merge the attributes). They are using this because it is more practical for real usage. Merging the attributes is an expensive task.

Dateset		LastFM	Yelp
User-Item Interaction	#Users	1,801	27,675
	#Items	7,432	70,311
	#Interactions	76,693	1,368,606
	#attributes	**33**	**29**
Graph	#Entities	9,266	98,605
	#Relations	4	3
	#Triplets	138,217	2,884,567
Relations	**Description**	**Number of Relations**	
Interact	user ⟺ *item*	76,696	1,368,606
Friend	user ⟺ *user*	23,958	688,209
Like	user ⟺ *attribute*	7,276	*
Belong_to	item ⟺ *attribute*	30,290	350,175

Dateset		LastFM*	Yelp*
User-Item Interaction	#Users	1,801	27,675
	#Items	7,432	70,311
	#Interactions	76,693	1,368,606
	#attributes	**8,438**	**590**
Graph	#Entities	17,671	98,576
	#Relations	4	3
	#Triplets	228,217	2,533,827
Relations	**Description**	**Number of Relations**	
Interact	user ⟺ *item*	76,696	1,368,606
Friend	user ⟺ *user*	23,958	688,209
Like	user ⟺ *attribute*	33,120	*
Belong_to	item ⟺ *attribute*	94,446	477,012

Comparison in performance

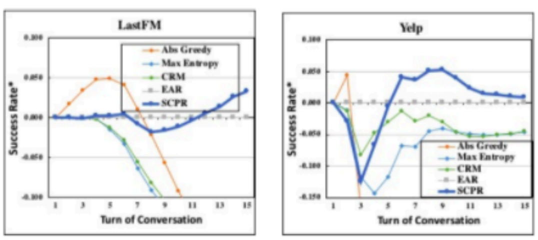


Figure 3: Success Rate* of compared methods at different turns on LastFM and Yelp (RQ1).

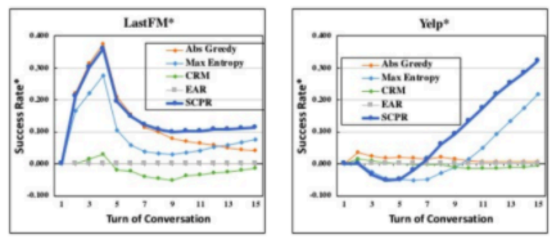


Figure 4: Success Rate* of compared methods at different conversation turns on LastFM* and Yelp*(RQ1).

- Looking at the * dataset, SCPR achieves more advantage by using having more attributes in the dataset.
- Both EAR and CRM are outperformed by Abs-greedy and Max entropy on the few first rounds on * dataset. It is because EAR and CRM are not good given large output space.

Ablation study

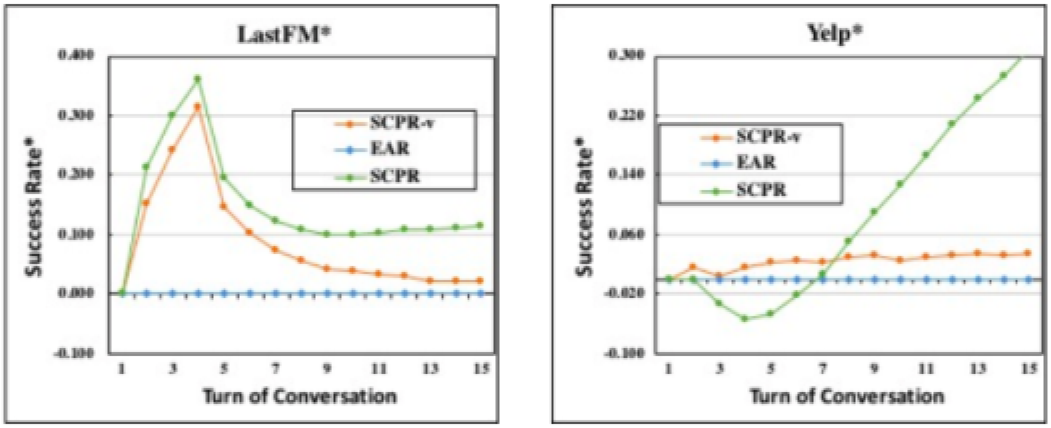


Figure 6: Success Rate* of compared methods at different conversation turns on LastFM* and Yelp*(RQ2).

- Creates SCPR-v = the same as SCPR, but the RL model has the same action space as in EAR (it also decides which attributes to ask).
- The SCPR-v is like an intermediate version between EAR and SCPR:
    - For SCPR: the action space is not so focused
    - For EAR: can be helped by the KG constraints.
- The SPCR-v performs better at the first few turns, but falls behind in the future.
    - The SCPR-v model will ask **a few attributes** and **recommend items at earlier turns**.
    - Recommend items at earlier turns make the success rate of the first few turns higher compared to SCPR.
    - With a few attributes, it will has more candidate items to recommend, making the model harder to give the best items.

6/ Conclusion

- Knowledge graph helps:
    - better question
    - coherent conversation
    - Explainable!
- Reducing RL output spaces helps:
    - RL to have better decision making

Paper 3: Towards Conversational Recommendation over Multi-Type Dialogs

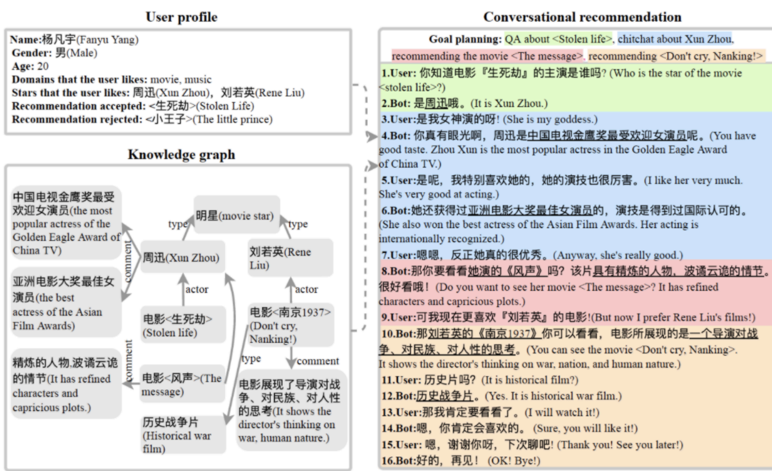
1. Introduction

-Existing task oriented recommendation conversations mostly focus on one single task.
- Current systems assume that both sides of users are aware of the conversation goal from the beginning.
-No application for multi-types conversations: Chitchat+Task-oriented+QA

2. Contributions of this paper

-Identify multi-type conversations
-Propose new dataset DuRecDial
-Propose novel model MGCG(multi-goal driven conversation generation framework)
-Actively lead the conversation

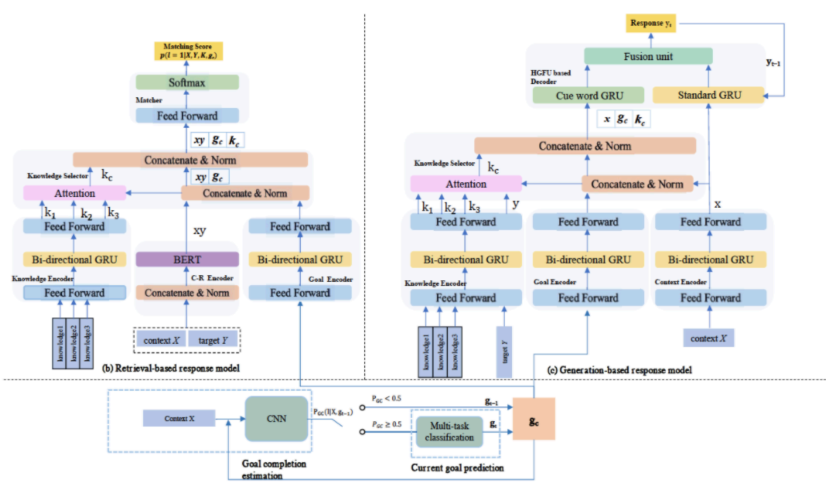
3. Example of Conversational Recommendation over multi-type dialogs



4. Data Collection:

- Seeker Profile+KG+Task templates+Dialog Recommendations

5. MGCG Framework



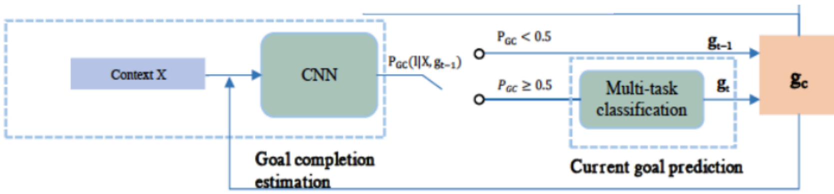
5.1 Goal Planning

-Goal Completion Estimation & Current Goal Prediction

$$g_t = \arg\max_{g^{ty}, g^{tp}} P_{GP}(g^{ty}, g^{tp} | X, \mathcal{G}', \mathcal{P}_i^{sk}, \mathcal{K}),$$

$$P_{GC}(l = 1 | X, g_{t-1}).$$

$$g_c = g_t,$$



(a) Goal-planning module

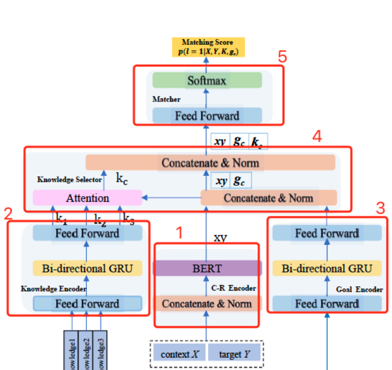
5.2 Retrieved-based Response Model

-Attention distribution->Matching probability

$$p(k_i | x, y, g_c) = \frac{exp(\mathbf{MLP}([xy; g_c]) \cdot k_i)}{\sum_j exp(\mathbf{MLP}([xy; g_c]) \cdot k_j)}$$

-Matching probability:

$$p(l = 1 | X, Y, K, g_c) = softmax(\mathbf{MLP}([xy; k_c; g_c]))$$



(b) Retrieval-based response model

5.3 Generation-based Response Model

-KL Div Loss

$$p(k_i | x, y, g_c) = \frac{exp(k_i \cdot \mathbf{MLP}([x; y; g_c]))}{\sum_{j=1}^{N} exp(k_j \cdot \mathbf{MLP}([x; y; g_c]))}$$

$$p(k_i | x, g_c) = \frac{exp(k_i \cdot \mathbf{MLP}([x; g_c]))}{\sum_{j=1}^{N} exp(k_j \cdot \mathbf{MLP}([x; g_c]))}$$

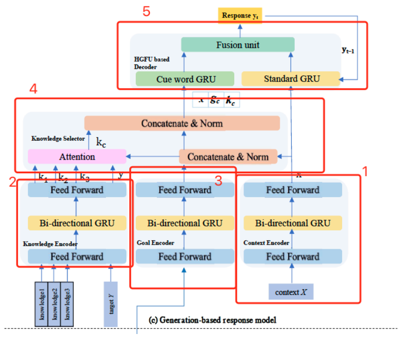
$$L_{KL}(\theta) = \frac{1}{N} \sum_{i=1}^{N} p(k_i | x, y, g_c) log \frac{p(k_i | x, y, g_c)}{p(k_i | x, g_c)}$$

-BOW Loss

$$L_{BOW}(\theta) = -\frac{1}{m} \sum_{t=1}^{m} log p(y_t | k_c)$$

-Loss Function

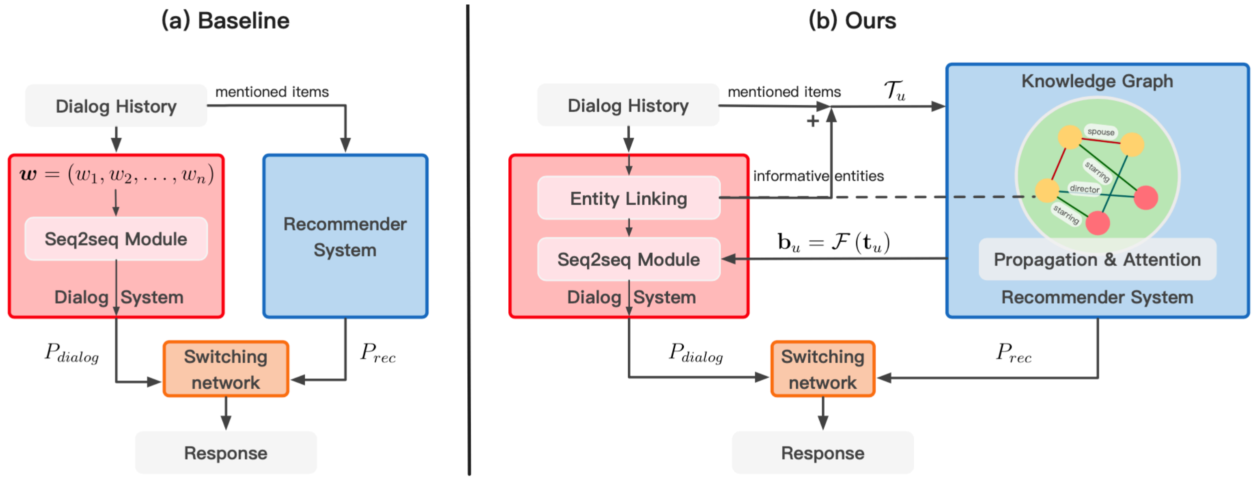
$$L(\theta) = \alpha \cdot L_{KL}(\theta) + \alpha \cdot L_{NLL}(\theta) + L_{BOW}(\theta)$$



6. Experiment Results
   -MGCG outperforms by large margin in terms of all metrics
   -Methods using goals and knowledge outperform those without goals and knowledge
   -Human Evaluation: Retrieval-based model performs better in terms of fluency; Produce more appropriate and informative response.

Paper 4: Towards Knowledge-Based Recommender Dialog System

1. Summary:
   In this paper, the authors propose a novel end-to-end framework Knowledge-Based Recommender Dialog (KBRD) system, combining the recommender system and the dialog generation system. Through the interaction between these two systems to introduce user preferences and provide recommendation-aware vocabulary bias.
2. Motivation: Compare to the model ReDial (NeurIPS2018)
   a. Only mentioned items are used for recommenders.
   b. Recommenders cannot help generate better dialog.
3. Model:



a. Knowledge-graph information (contextual information)
   i. Knowledge graph G consisting of triples (h,r,t), h,t in E and r in R.
   ii. User representation: T_u = {e_1,e_2, …, e_{|T_u|}}, combining the item link and entity link, containing item information and non-item information in the dialog contents.
   iii. Relational graph propagation with Relational Graph Convolutional Networks (R-GCNs): to learn the representation of every node in the knowledge graph, we can combine the neighbors representation h_w with weight W_r to learn a new representation of the central node h_v.

$$h_v^{(l+1)} = \sigma\left(\sum_{r\in\mathcal{R}}\sum_{w\in\mathcal{N}_v^r}\frac{1}{c_{v,r}}W_r^{(l)}h_w^{(l)} + W_0^{(l)}h_v^{(l)}\right)$$

   iv. Entity attention: to learn the final representation of user, we use the attention mechanism w_{a1,a2} to combine the entity representation with learned weights alpha to emphasize the preference of user:

$$\mathbf{H}_u = (h_1, h_2, \cdots, h_{|\mathcal{T}_u|})$$

$$\alpha_u = \mathrm{softmax}\left(w_{a2}\tanh\left(W_{a1}\mathbf{H}_u^T\right)\right)$$

$$t_u = \alpha_u \mathbf{H}_u$$

b. Recommend system P_rec

$$P_{\mathrm{rec}} = \mathrm{softmax}(\mathrm{mask}(t_u\mathbf{H}^T))$$

c. Dialog generation system P_dialog
   i. Transformer based model with encoder layer and decoder layer

$$P_{\mathrm{dialog}} = \mathrm{softmax}\left(Wo + b\right)$$

   ii. Improve with the user information t_u from knowledge graph

$$b_u = \mathcal{F}(t_u)$$

$$P_{\mathrm{dialog}} = \mathrm{softmax}\left(Wo + b + b_u\right)$$

d. End-to-end system: A switching mechanism controls the decoder to decide whether it should generate a word from the vocabulary or an item from the recommender output at a certain time step *s*.

$$P(w) = p_s P_{\text{dialog}}(w) + (1 - p_s)\, P_{\text{rec}}(w)$$

$$p_s = \sigma\left(w_s o + b_s\right)$$

4. Experiment
   a. Dataset:
      i. Conversational recommendation: Recommendations through Dialog (ReDial)
      ii. Knowledge Graph: DBpedia (containing movies and relevant entities, such as director and style)
   b. Metrics:
      i. For Dialog: automatic evaluation (perplexity and distinct 3&4-gram) & human evaluation(10 annotators score the candidates)
      ii. For Recommendation: Recall@K, top-k item with groundtrue
   c. Baselines: ReDial, and Transformer in ReDial
   d. Result: all are improved
      i. Dialog results:

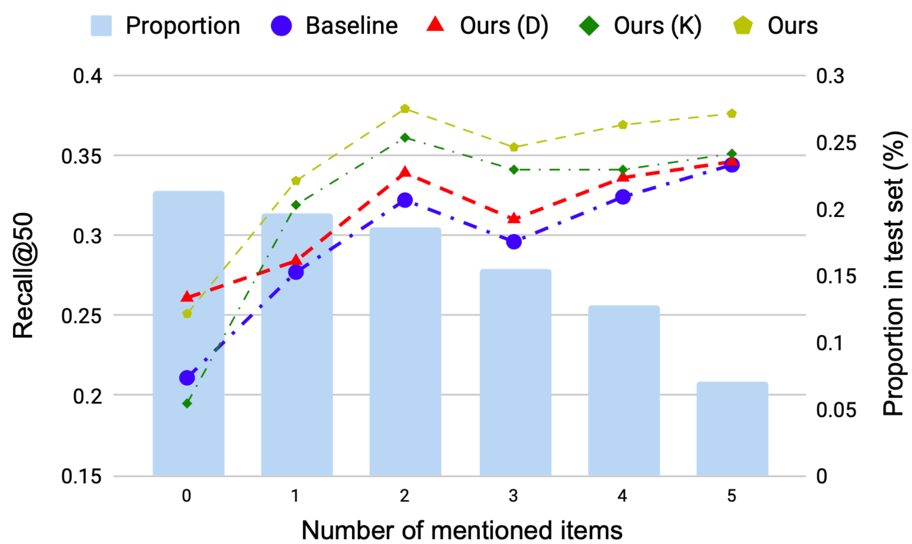
Model	PPL	Dist-3	Dist-4	CSTC
REDIAL	28.1	0.11	0.13	1.73
Transformer	18.0	0.27	0.39	-
KBRD	**17.9**	**0.30**	**0.45**	**1.99**

Consistency with dialog history, KBRD better than ReDial with 15% increase. The baseline REDIAL does not have a strong connection between the dialog system and user representation. Instead, in KBRD, the recommender system provides the recommendation-aware vocabulary bias b_u, which is based on the user representation t_u, to the dialog system. Thus the dialog system gains knowledge about the user's preference and generates a consistent response.

      ii. Recommendation results:

Model	R@1	R@10	R@50
REDIAL	2.3±0.2	12.9±0.7	28.7±0.9
KBRD (D)	2.7±0.2	14.0±0.6	30.6±0.7
KBRD (K)	2.6±0.2	14.4±0.9	31.0±1.2
KBRD	**3.0**±0.2	**16.3**±0.3	**33.8**±0.7

5. Research Questions:
   a. Does dialog help recommendation?



This represents the efficiency of the recommender system, which can save users' time and efforts. On average, the system with both information sources performs the best. Dialog introduces contextual information and knowledge introduces movie features and structural connection with other movies.

b. Does recommendation help dialog?

Movie	1	2	3	4	5	6	7	8
Star Wars	space	alien	sci-fi	star	sci	robot	smith	harry
The Shining	creepy	stephen	gory	horror	scary	psychological	haunted	thriller
The Avengers (2012)	marvel	superhero	super	dc	wait	batman	thor	take
Beauty and the Beast	cute	disney	animated	live	music	child	robin	kids

It can be found that the interaction with the recommendation system can enhance the performance of the dialog system in both automatic evaluation and human evaluation. From several examples shown in above Table, we observe that the words are highly related to the mentioned movies. Therefore, it can be suggested that the recommendation system conveys important information to the dialog system in the form of a vocabulary bias.

6. Conclusion:
The authors propose a novel end-to-end framework, KBRD, which bridges the gap between the recommender system and the dialog system via knowledge propagation. Through a series of experiments, we show that KBRD can reach better performances in both recommendation and dialog generation in comparison with the baselines.

Resources List to Shortlist From: TBDeleted or Cleaned After Week 10 Scribing

https://arxiv.org/abs/2002.09102
https://arxiv.org/abs/2007.00194
For other candidate paper, I recommend a very comprehensive tutorial in SIGIR 2020, you guys can select from it: http://staff.ustc.edu.cn/~hexn/papers/sigir20-tutorial.pdf
Slides here: http://staff.ustc.edu.cn/~hexn/slides/sigir20-tutorial-CRS-slides.pdf
The SIGIR tutorial has listed 4 mainstream directions in CRS, I suggest we may select 1-2 papers from each direction respectively. On S12 of the slides, it listed:
1. Question Driven Approaches
2. Multi-turn Conversational Recommendation Strategy (2 of my nominated paper)
3. Exploitation-Exploration Trade-offs for Cold Users
4. Dialogue Understanding and Generation

    [SIGIR'18] Conversational recommender system
    [20] A Survey on Conversational Recommender Systems
    [WWW'20] Latent Linear Critiquing for Conversational Recommender Systems
    [WSDN'20] Estimation–Action–Reflection: Towards Deep Interaction Between Conversational and Recommender Systems
    Deep Conversational Recommender System: A New Frontier for Goal-Oriented Dialogue Systems

And some recent related good works:

    Towards Question-based Recommender Systems (Very novel idea of modifying the MF parameter during chat)
    Towards Conversational Recommendation over Multi-Type Dialogs (A Industrial level work)