Commonsense Knowledge Aware Conversation Generation with Graph Attention

Hao Zhou[1], **Tom Young**[2], **Minlie Huang**[1*], **Haizhou Zhao**[3], **Jingfang Xu**[3] and **Xiaoyan Zhu**[1]
[1] Conversational AI Group, AI Lab., Dept. of Computer Science, Tsinghua University
Beijing National Research Center for Information Science and Technology, China
[2] School of Information and Electronics, Beijing Institute of Technology, China
[3] Sogou Inc., Beijing, China
tuxchow@gmail.com , tom@sentic.net, aihuang@tsinghua.edu.cn

Abstract

Commonsense knowledge is vital to many natural language processing tasks. In this paper, we present a novel open-domain conversation generation model to demonstrate how large-scale commonsense knowledge can facilitate language understanding and generation. Given a user post, the model retrieves relevant knowledge graphs from a knowledge base and then encodes the graphs with a *static graph attention* mechanism, which augments the semantic information of the post and thus supports better understanding of the post. Then, during word generation, the model attentively reads the retrieved knowledge graphs and the knowledge triples within each graph to facilitate better generation through a *dynamic graph attention* mechanism. This is the first attempt that uses large-scale commonsense knowledge in conversation generation. Furthermore, unlike existing models that use knowledge triples (entities) separately and independently, our model treats each knowledge graph as a whole, which encodes more structured, connected semantic information in the graphs. Experiments show that the proposed model can generate more appropriate and informative responses than state-of-the-art baselines.

1 Introduction

Semantic understanding, particularly when facilitated by commonsense knowledge or world facts, is essential to many natural language processing tasks [Wang *et al.*, 2017; Lin *et al.*, 2017], and undoubtedly, it is a key factor to the success of dialogue or conversational systems, as conversational interaction is a semantic activity [Eggins and Slade, 2005]. In open-domain conversational systems, commonsense knowledge is important for establishing effective interactions, since socially shared commonsense knowledge is the set of background information people intended to know and use during conversation [Minsky, 1991; Marková *et al.*, 2007; Speer and Havasi, 2012; Souto, 2015].

*Corresponding author: Minlie Huang.

Recently, a variety of neural models has been proposed for conversation generation [Ritter *et al.*, 2011; Shang *et al.*, 2015]. However, these models tend to generate generic responses, which are unable to respond appropriately and informatively in most cases, because it is challenging to learn semantic interactions *merely* from conversational data [Ghazvininejad *et al.*, 2017] without deep understanding of user input, and the background knowledge and the context of conversation. A model can understand conversations better and thus respond more properly if it can access and make full use of large-scale commonsense knowledge. For instance, to understand a post-response pair "*Don't order drinks at the restaurant , ask for free water*" and "*Not in Germany. Water cost more than beer. Bring you own water bottle*", we need commonsense knowledge such as (*water, AtLocation, restaurant*), (*free, RelatedTo, cost*), etc.

Some prior studies have been conducted to introduce external knowledge in conversation generation [Han *et al.*, 2015; Ghazvininejad *et al.*, 2017; Zhu *et al.*, 2017]. The knowledge used in these models is either unstructured texts [Ghazvininejad *et al.*, 2017] or domain-specific knowledge triples [Zhu *et al.*, 2017]. Therefore, such models face with two issues when they are applied to open-domain, open-topic conversation generation. **First**, they are highly dependent on the quality of unstructured texts or limited by the small-scale, domain-specific knowledge. **Second**, they usually make use of knowledge triples (entities) separately and independently, instead of treating knowledge triples as a whole in a graph. Thus, they are unable to represent the semantics of a graph via linked entities and relations.

To address the two issues, we propose a commonsense knowledge aware conversational model (CCM) to facilitate language understanding and generation in open-domain conversational systems. We use a large-scale commonsense knowledge [Speer and Havasi, 2012] to help understand the background information of a given post, and to facilitate response generation with such knowledge. The model retrieves a few knowledge graphs for each post and then use the graphs to respond more informatively and appropriately, as shown in Figure 1. To fully leverage the retrieved graphs in conversation generation, two novel graph attention mechanisms are designed. A static graph attention mechanism encodes

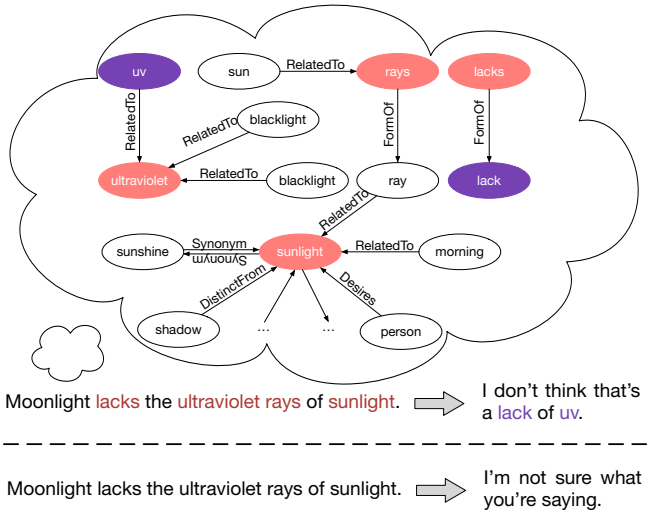

Figure 1: (Better viewed in color) Two response examples by our model (the first line) and Seq2Seq (second) with/without considering commonsense knowledge, respectively.

the retrieved graphs for a post to augment the semantic representation of the post, which can help understand the post. A dynamic graph attention mechanism attentively reads the knowledge graphs and the triples in each graph, and then uses the semantic information from the graphs and triples for better response generation.

In summary, this paper makes the following contributions:

- This work is the first attempt that uses large-scale commonsense knowledge in neural conversation generation. Supported by such knowledge, our model can understand the dialogue better and thus respond more appropriately and informatively.
- Instead of treating knowledge triples (or entities) separately and independently, we devise static and dynamic graph attention mechanisms to treat the knowledge triples as a graph, from which we can better interpret the semantics of an entity from its neighboring entities and relations.

2 Related Work

Open-domain Conversational Models

Recently, sequence-to-sequence models [Sutskever *et al.*, 2014; Bahdanau *et al.*, 2014] have been successfully applied to large-scale conversation generation, including neural responding machine [Shang *et al.*, 2015], hierarchical recurrent models [Serban *et al.*, 2015], and many others [Sordoni *et al.*, 2015]. These models developed various techniques to improve the content quality of generated responses, including diversity promotion [Li *et al.*, 2016; Shao *et al.*, 2017], considering additional information [Xing *et al.*, 2017; Mou *et al.*, 2016], and handling unknown words [Gu *et al.*, 2016]. However, generic or meaningless responses are still commonly seen in these models due to the inability of good understanding of the user input or other context.

Unstructured Texts Enhanced Conversational Models

Several studies incorporated unstructured texts as external knowledge into conversation generation [Ghazvininejad *et al.*, 2017; Long *et al.*, 2017]. [Ghazvininejad *et al.*, 2017] used memory network which stores unstructured texts to improve conversation generation. [Long *et al.*, 2017] applied a convolutional neural network to extract knowledge from unstructured texts to generate multi-turn conversations. However, these models largely depend on the quality of unstructured texts, which may introduce noise in conversation generation if the texts are irrelevant.

Structured Knowledge Enhanced Conversational Models

There exist some models that introduced high-quality structured knowledge for conversation generation [Han *et al.*, 2015; Zhu *et al.*, 2017; Xu *et al.*, 2017]. [Xu *et al.*, 2017] incorporated a structured domain-specific knowledge base into conversation generation with a recall-gate mechanism. [Zhu *et al.*, 2017] presented an end-to-end knowledge grounded conversational model using a copy network [Gu *et al.*, 2016]. However, these studies are somehow limited by the small domain-specific knowledge base, making them not applicable for open-domain, open-topic conversation generation. By contrast, our model applies a large-scale commonsense knowledge base to facilitate both the understanding of a post and the generation of a response, with novel graph attention mechanisms.

3 Commonsense Conversational Model

3.1 Background: Encoder-decoder Framework

First of all, we introduce a general encoder-decoder framework which is based on sequence-to-sequence (seq2seq) learning [Sutskever *et al.*, 2014]. The encoder represents a post sequence $X = x_1 x_2 \cdots x_n$ with hidden representations $\boldsymbol{H} = \boldsymbol{h}_1 \boldsymbol{h}_2 \cdots \boldsymbol{h}_n$[1], which is briefly defined as below:

$$\boldsymbol{h}_t = \mathbf{GRU}(\boldsymbol{h}_{t-1}, \boldsymbol{e}(x_t)), \tag{1}$$

where $\boldsymbol{e}(x_t)$ is the embedding of the word x_t, and GRU is gated recurrent unit [Cho *et al.*, 2014].

The decoder takes as input a context vector $\boldsymbol{c}_t$ and the embedding of a previously decoded word $\boldsymbol{e}(y_{t-1})$, and updates its state $\boldsymbol{s}_t$ using another GRU:

$$\boldsymbol{s}_t = \mathbf{GRU}(\boldsymbol{s}_{t-1}, [\boldsymbol{c}_{t-1}; \boldsymbol{e}(y_{t-1})]), \tag{2}$$

where $[\boldsymbol{c}_{t-1}; \boldsymbol{e}(y_{t-1})]$ is the concatenation of the two vectors, serving as input to the GRU network. The context vector $\boldsymbol{c}_{t-1}$ is an attentive read of $\boldsymbol{H}$, which is a weighted sum of the encoder's hidden states as $\boldsymbol{c}_{t-1} = \sum_{k=1}^{n} \alpha_k^{t-1} \boldsymbol{h}_k$, and α_k^{t-1} measures the relevance between state $\boldsymbol{s}_{t-1}$ and hidden state $\boldsymbol{h}_k$. Refer to [Bahdanau *et al.*, 2014] for more details.

The decoder generates a token by sampling from the output probability distribution which can be computed as follows:

$$\begin{aligned} y_t \sim \boldsymbol{o}_t &= P(y_t \mid y_{<t}, \boldsymbol{c}_t) \\ &= \mathrm{softmax}(\mathbf{W_o} \boldsymbol{s}_t). \end{aligned} \tag{3}$$

where $y_{<t} = y_1 y_2 \cdots y_{t-1}$, the words already generated.

[1] Throughout the paper, a normal letter denotes a discrete symbol while a bolded letter denotes a vector.

3.2 Task Definition and Overview

Our problem is formulated as follows: Given a post $X = x_1 x_2 \cdots x_n$ and some commonsense knowledge graphs $G = \{g_1, g_2, \cdots, g_{N_G}\}$, the goal is to generate a proper response $Y = y_1 y_2 \cdots y_m$. Essentially, the model estimates the probability: $P(Y|X, G) = \prod_{t=1}^{m} P(y_t|y_{<t}, X, G)$. The graphs are retrieved from a knowledge base using the words in a post as queries, and each word corresponds to a graph in G [2]. Each graph consists of a set of triples $g_i = \{\tau_1, \tau_2, \cdots, \tau_{N_{g_i}}\}$ and each triple (head entity, relation, tail entity) is denoted as $\tau = (h, r, t)$.

We adopt TransE [Bordes *et al.*, 2013] to represent the entities and relations in the knowledge base. In order to bridge the representation gap between knowledge base and unstructured conversational texts, we adopt a MLP for the purpose: a knowledge triple τ is represented by $\boldsymbol{k} = (\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) = \mathbf{MLP}(\mathrm{TransE}(h, r, t))$, where $\boldsymbol{h}/\boldsymbol{r}/\boldsymbol{t}$ are the transformed TransE embeddings for $h/r/t$ respectively.

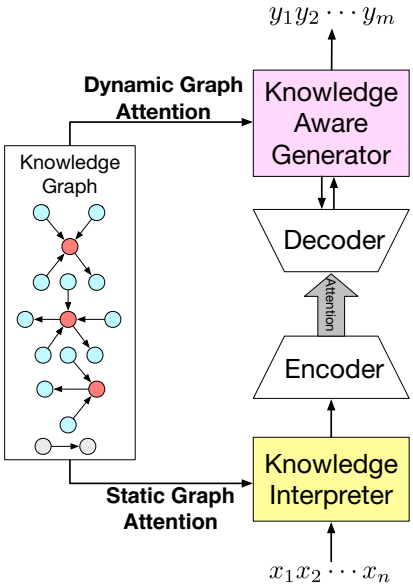

Figure 2: Overview of CCM.

The overview of our commonsense conversational model (CCM) is presented in Figure 2. The knowledge interpreter takes as input a post $X = x_1 x_2 \cdots x_n$ and retrieved knowledge graphs $G = \{g_1, g_2, \cdots, g_{N_G}\}$ to obtain knowledge-aware representations at each word position, by concatenating a word vector and its corresponding knowledge graph vector. A knowledge graph vector represents a knowledge graph for the corresponding word in X through a static graph attention mechanism. The knowledge aware generator generates a response $Y = y_1 y_2 \cdots y_m$ with our dynamic graph attention mechanism. At each decoding position, it attentively reads the retrieved graphs and the entities in each graph, and then generates a generic word in the vocabulary or an entity in the knowledge graphs. The entity is selected by attending on the graphs and the triples within each graph.

[2] For a word without any match, there is a special graph denoted as *Not_A_Fact*.

3.3 Knowledge Interpreter

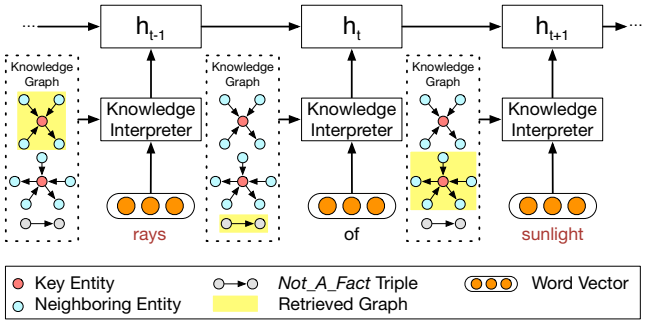

Figure 3: Knowledge interpreter concatenates a word vector and the graph vector of the corresponding retrieved graph. In this example, word *rays* (also key entity) corresponds to the first graph, and *sunlight* to the second one. Each graph is represented by a graph vector. A key entity is an entity which occurs in the post.

The knowledge interpreter is designed to facilitate the understanding of a post. It augments the semantics of a word by including the corresponding graph vector for the word, as shown in Figure 3. The knowledge interpreter uses each word x_t in a post as the key entity to retrieve a graph $g_i = \{\tau_1, \tau_2, \cdots, \tau_{N_{g_i}}\}$ (the yellow parts) from the entire commonsense knowledge base. Each retrieved graph consists of a key entity (the red dots), its neighboring entities (the blue dots) and relations between entities. For common words (e.g., *of*) which match no entity in the commonsense knowledge graph, a knowledge graph that contains a special symbol *Not_A_Fact* (the grey dots) is used. Then, the knowledge interpreter computes the graph vector $\boldsymbol{g}_i$ of the retrieved graph using the static graph attention mechanism. After concatenating the word vector $\boldsymbol{w}(x_t)$ and the knowledge graph vector $\boldsymbol{g}_i$, the concatenated vector $\boldsymbol{e}(x_t) = [\boldsymbol{w}(x_t); \boldsymbol{g}_i]$ is obtained and fed to the GRU cell of the encoder (see Eq. 1).

Static Graph Attention

The static graph attention mechanism is designed to generate a representation for a retrieved knowledge graph, inspired by [Velickovic *et al.*, 2017]. The major difference to [Velickovic *et al.*, 2017] lies in that our graph attention encodes more structured semantic information by considering not only all nodes in a graph but also relations between nodes.

The static graph attention generates a static representation for a graph, which will be used to augment the semantics of a word in a post.

Formally, the static graph attention mechanism takes as input the knowledge triple vectors $\boldsymbol{K}(g_i) = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \cdots, \boldsymbol{k}_{N_{g_i}}\}$ in the retrieved knowledge graph g_i, to produce a graph vector $\boldsymbol{g}_i$ as follows:

$$\boldsymbol{g}_i = \sum_{n=1}^{N_{g_i}} \alpha_n^s [\boldsymbol{h}_n; \boldsymbol{t}_n], \quad (4)$$

$$\alpha_n^s = \frac{\exp(\beta_n^s)}{\sum_{j=1}^{N_{g_i}} \exp(\beta_j^s)}, \quad (5)$$

$$\beta_n^s = (\mathbf{W_r} \boldsymbol{r}_n)^\top \tanh(\mathbf{W_h} \boldsymbol{h}_n + \mathbf{W_t} \boldsymbol{t}_n), \quad (6)$$

where $(\boldsymbol{h_n}, \boldsymbol{r_n}, \boldsymbol{t_n}) = \boldsymbol{k_n}$, $\mathbf{W_h}, \mathbf{W_r}, \mathbf{W_t}$ are weight matrices for head entities, relations, and tail entities, respectively. The attention weight measures the association of a relation r_n to a head entity h_n and a tail entity t_n.

Essentially, a graph vector $\boldsymbol{g}_i$ is a weighted sum of the head and tail vectors $[\boldsymbol{h}_n; \boldsymbol{t}_n]$ of the triples contained in the graph.

3.4 Knowledge Aware Generator

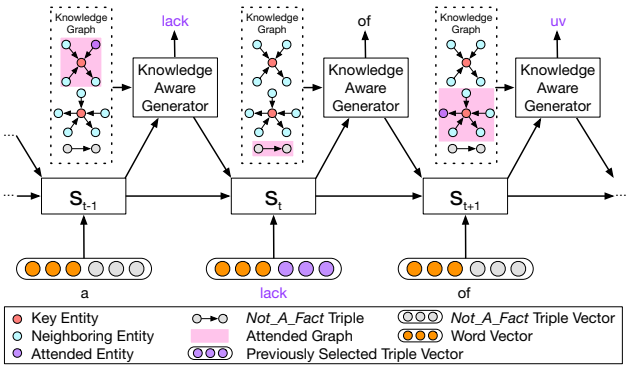

Figure 4: Knowledge aware generator dynamically attends on the graphs (the pink graph is mostly attended) . It then attentively reads the triples in each graph to estimate the probability of selecting a triple, where the triple's neighboring entity (purple dots/words) is used for word generation.

The knowledge aware generator is designed to generate a response through making full use of the retrieved knowledge graphs, as shown in Figure 4. The knowledge aware generator plays two roles: 1) attentively reading the retrieved graphs to obtain a graph-aware context vector, and using the vector to update the decoder's state; 2) adaptively choosing a generic word or an entity from the retrieved graphs for word generation.Formally, the decoder updates its state as follows:

$$\boldsymbol{s}_{t+1} = \mathbf{GRU}(\boldsymbol{s}_t, [\boldsymbol{c}_t; \boldsymbol{c}_t^g; \boldsymbol{c}_t^k; \boldsymbol{e}(y_t)]), \quad (7)$$

$$\boldsymbol{e}(y_t) = [\boldsymbol{w}(y_t); \boldsymbol{k}_j], \quad (8)$$

where $\boldsymbol{e}(y_t)$ is the concatenation of the word vector $\boldsymbol{w}(y_t)$ and the previous knowledge triple vector $\boldsymbol{k}_j$ from which the previous word (y_t) is selected,

$\boldsymbol{c}_t$ is the context vector as used in Eq. 2, $\boldsymbol{c}_t^g$ and $\boldsymbol{c}_t^k$ are context vectors attended on knowledge graph vectors $\{\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_{N_G}\}$ and knowledge triple vectors $\{\boldsymbol{K}(g_1), \boldsymbol{K}(g_2), \cdots, \boldsymbol{K}(g_{N_G})\}$ respectively.

Dynamic Graph Attention

The dynamic graph attention mechanism is a hierarchical, top-down process. It first attentively reads all the knowledge graphs and then attentively reads all the triples in each graph for final word generation. Given the decoder state $\boldsymbol{s}_t$, it first attends on the knowledge graph vectors $\{\boldsymbol{g}_1, \boldsymbol{g}_2, \cdots, \boldsymbol{g}_{N_G}\}$ to compute the probability of using of each graph g_i, which is defined as below:

$$\boldsymbol{c}_t^g = \sum_{i=1}^{N_G} \alpha_{ti}^g \boldsymbol{g}_i, \quad (9)$$

$$\alpha_{ti}^g = \frac{\exp(\beta_{ti}^g)}{\sum_{j=1}^{N_G} \exp(\beta_{tj}^g)}, \quad (10)$$

$$\beta_{ti}^g = \boldsymbol{V}_b^\top \tanh(\mathbf{W_b} \boldsymbol{s}_t + \mathbf{U_b} \boldsymbol{g}_i), \quad (11)$$

where $\boldsymbol{V}_b/\boldsymbol{W}_b/\boldsymbol{U}_b$ are parameters, and α_{ti}^g is the probability of choosing knowledge graph g_i at step t. The graph context vector $\boldsymbol{c}_t^g$ is a weighted sum of the graph vectors, and the weight measures the association between the decoder's state $\boldsymbol{s}_t$ and a graph vector $\boldsymbol{g}_i$.

The model then attends on the knowledge triple vectors $\boldsymbol{K}(g_i) = \{\boldsymbol{k}_1, \boldsymbol{k}_2, \cdots, \boldsymbol{k}_{N_{g_i}}\}$ within each graph g_i to calculate the probability of selecting a triple for word generation, formally as follows:

$$\boldsymbol{c}_t^k = \sum_{i=1}^{N_G} \sum_{j=1}^{N_{g_i}} \alpha_{ti}^g \alpha_{tj}^k \boldsymbol{k}_j, \quad (12)$$

$$\alpha_{tj}^k = \frac{\exp(\beta_{tj}^k)}{\sum_{n=1}^{N_{g_i}} \exp(\beta_{tn}^k)}, \quad (13)$$

$$\beta_{tj}^k = \boldsymbol{k}_j^\top \mathbf{W_c} \boldsymbol{s}_t, \quad (14)$$

where β_{tj}^k can be viewed as the similarity between each knowledge triple vector $\boldsymbol{k}_j$ and the decoder state $\boldsymbol{s}_t$, α_{tj}^k is the probability of choosing triple τ_j from all triples in graph g_i at step t.

Finally, the knowledge aware generator selects a generic word or an entity word [3] with the following distributions:

$$\boldsymbol{a}_t = [\boldsymbol{s}_t; \boldsymbol{c}_t; \boldsymbol{c}_t^g; \boldsymbol{c}_t^k], \quad (15)$$

$$\gamma_t = \text{sigmoid}(\mathbf{V_o}^\top \boldsymbol{a}_t), \quad (16)$$

$$P_c(y_t = w_c) = \text{softmax}(\mathbf{W_o} \boldsymbol{a}_t), \quad (17)$$

$$P_e(y_t = w_e) = \alpha_{ti}^g \alpha_{tj}^k, \quad (18)$$

$$y_t \sim \boldsymbol{o}_t = P(y_t) = \begin{bmatrix} (1-\gamma_t) P_g(y_t = w_c) \\ \gamma_t P_e(y_t = w_e) \end{bmatrix}, \quad (19)$$

where $\gamma_t \in [0, 1]$ is a scalar to balance the choice between an entity word w_e and a generic word w_c, P_c/P_e is the distribution over generic/entity words respectively. The final distribution $P(y_t)$ is a concatenation of two distributions.

3.5 Loss Function

The loss function is cross entropy between the predicted token distribution $\boldsymbol{o}_t$ and the reference distribution $\boldsymbol{p}_t$ in the

[3] Entity words are taken from the neighboring entities of the knowledge triples.

training corpus. Additionally, we apply supervised signals on the knowledge aware generator layer to teacher-force the selection of an entity or a generic word. The loss on one sample $< X, Y >$ ($X = x_1 x_2 \cdots x_n$, $Y = y_1 y_2 \cdots y_m$) is defined as:

$$L(\theta) = -\sum_{t=1}^{m} \boldsymbol{p}_t \log(\boldsymbol{o}_t) - \sum_{t=1}^{m} (q_t \log(\gamma_t) + (1-q_t)\log(1-\gamma_t)), \tag{20}$$

where γ_t is the probability of selecting an entity word or a generic word, and $q_t \in \{0, 1\}$ is the true choice of an entity word or a generic word in $\boldsymbol{Y}$. The second term is used to supervise the probability of selecting an entity word or a generic word.

4 Experiments

4.1 Dataset

Commonsense Knowledge Base

ConceptNet[4] is used as the commonsense knowledge base. It contains not only world facts such as "*Paris is the capital of France*" that are constantly true, but also informal relations between common concepts that are part of daily knowledge such as "*A dog is a pet*". This feature is desirable in our experiments, because the ability to recognize the informal relations between common concepts is necessary in the open-domain conversation setting. For simplicity, we removed triples containing multi-word entities, and 120,850 triples were retained with 21,471 entities and 44 relations.

Commonsense Conversation Dataset

We adopted 10M reddit single-round dialogs from the site[5]. Since we target at using commonsense knowledge to facilitate language understanding and generation, we filtered the original corpus with the knowledge triples. If a post-response pair can not be connected by any triple (that is, one entity appears in the post and the other in the response), the pair will be removed. The statistics can be seen in Table 1.

We randomly sampled 10,000 pairs for validation. To test how commonsense knowledge can help understand common or rare concepts in a post, we constructed four test sets: high-frequency pairs in which each post has all top 25% frequent words, medium-frequency pairs where each post contains at least one word whose frequency is within the range of 25%-75%, low-frequency pairs within the range of 75%-100%, and OOV pairs where each post contains out-of-vocabulary words. Each test set has 5,000 pairs randomly sampled from the dataset[6].

4.2 Implementation Details

Our model was implemented with Tensorflow[7]. The encoder and decoder have 2-layer GRU structures with 512 hidden cells for each layer and they do not share parameters. The word embedding size is set to 300. The vocabulary size is limited to 30,000. We used TransE [Bordes *et al.*, 2013] to obtain entity and relation representations. The embedding size of entities and relations is set to 100.

We used the Adam optimizer with a mini-batch size of 100. The learning rate is 0.0001. The models were ran at most 20 epoches, and the training stage of each model took about a week on a Titan X GPU machine. Our code is available at: `https://github.com/tuxchow/ccm`.

Conversational Pairs		Commonsense KB	
Training	3,384,185	Entity	21,471
Validation	10,000	Relation	44
Test	20,000	Triple	120,850

Table 1: Statistics of the dataset and the knowledge base.

4.3 Baselines

We chose several suitable baselines:

- A seq2seq model (Seq2Seq) [Sutskever *et al.*, 2014], which is widely used in open-domain conversational systems.
- A knowledge-grounded model (MemNet) adapted from [Ghazvininejad *et al.*, 2017], where the memory units store TransE embeddings of knowledge triples.
- A copy network (CopyNet) model [Zhu *et al.*, 2017], which copies a word from knowledge triples or generates a word from the vocabulary.

4.4 Automatic Evaluation

Metrics: We adopted *perplexity* [Serban *et al.*, 2015] to evaluate the model at the content level (whether the content is grammatical and relevant in topic). We also calculated the number of entities per response to measure the model's ability to select the concepts from the commonsense knowledge base in generation. This metric is denoted by *entity score*.

Results: As shown in Table 2, CCM obtains the lowest perplexity on all the test sets, indicating that CCM can understand users' posts better and generate more grammatical responses. Moreover, CCM selects the most entities from the commonsense knowledge among the models during generation, demonstrating that commonsense knowledge can truly facilitate response generation.

More interestingly, commonsense knowledge is more frequently used by CCM in low-frequency posts than in high-frequency ones (*entity score*: **1.196 vs. 1.156**). This is in line with our intuition that rare concepts need more background knowledge to understand and respond. For the *perplexity*, the score for high-frequency posts is lower than low-frequency ones (**35.36 vs. 40.67**) since the probabilities for the common words can be more sufficiently trained.

4.5 Manual Evaluation

We resorted to a crowdsourcing service, Amazon Mechanical Turk, for manual annotation. 400 posts were randomly sampled for manual annotation. We conducted pair-wise comparison between the response generated by CCM and the one by

[4] `https://conceptnet.io`

[5] `https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/`

[6] Our data are available at: `http://coai.cs.tsinghua.edu.cn/hml/dataset/#commonsense`

[7] `https://github.com/tensorflow/tensorflow`

Model	Overall		High Freq.		Medium Freq.		Low Freq.		OOV	
	ppx.	ent.	ppx.	ent.	ppx.	ent.	ppx.	ent.	ppx.	ent.
Seq2Seq	47.02	0.717	42.41	0.713	47.25	0.740	48.61	0.721	49.96	0.669
MemNet	46.85	0.761	41.93	0.764	47.32	0.788	48.86	0.760	49.52	0.706
CopyNet	40.27	0.96	36.26	0.91	40.99	0.97	42.09	0.96	42.24	0.96
CCM	**39.18**	**1.180**	**35.36**	**1.156**	**39.64**	**1.191**	**40.67**	**1.196**	**40.87**	**1.162**

Table 2: Automatic evaluation with *perplexity* (ppx.), and *entity score* (ent.).

Model	Overall		High Freq.		Medium Freq.		Low Freq.		OOV	
	app.	inf.	app.	inf.	app.	inf.	app.	inf.	app.	inf.
CCM vs. Seq2Seq	0.616	0.662	0.605	0.656	0.549	0.624	0.636	0.650	0.673	0.716
CCM vs. MemNet	0.602	0.647	0.593	0.656	0.566	0.640	0.622	0.635	0.626	0.657
CCM vs. CopyNet	0.600	0.640	0.606	0.669	0.586	0.619	0.610	0.633	0.596	0.640

Table 3: Manual evaluation with *appropriateness* (app.), and *informativeness* (inf.). The score is the percentage that CCM wins its competitor after removing "Tie" pairs. CCM is significantly better (sign test, p-value < 0.005) than all the baselines on all the test sets.

a baseline for the same post. In total, there are 1,200 pairs since we have three baselines. For each response pair, seven judges were hired to give a preference between the two responses, in terms of the following two metrics. Tie was allowed. Notice that system identifiers were masked during annotation.

Metrics: We defined two metrics: *appropriateness* at the content level (whether the response is appropriate in grammar, topic, and logic); and *informativeness* at the knowledge level (whether the response provides new information and knowledge in addition to the post).

Annotation Statistics: We calculated the agreements to measure inter-rater consistency. For *appropriateness*, the percentage of the pairs that at least 4 judges gave the same label (4/7 agreement) amounts to 96.3%, and the percentage for at least 5/7 agreement is 60.2%. For *informativeness*, the percentage for at least 4/7 agreement is 90.4% and that for at least 5/7 agreement is 55.1%.

Results: The results are shown in Table 3. The score is the percentage that CCM wins a baseline after removing "Tie" pairs. CCM outperforms all the baselines significantly in terms of both metrics (sign test, p-value < 0.005) on all the test sets. Furthermore, CCM has about 60% chances to win the strongest baseline, CopyNet, which makes use of knowledge triples (entities) separately and independently. This demonstrates the effectiveness of our graph attention mechanisms.

Noticeably, the probabilities that CCM wins Seq2Seq on the OOV dataset are remarkably higher than those on the high-frequency dataset (**0.673 vs. 0.605** in *appropriateness*, and **0.716 vs. 0.656** in *informativeness*). This further indicates that commonsense knowledge is more useful in understanding rare concepts in post since Seq2Seq has no ability to use such knowledge. For MemNet and CopyNet, we did not observe such differences because the two baselines have the ability of using knowledge more or less.

4.6 Case Study

A sample conversation is shown in Table 4. The red-colored word "*breakable*" in the post is an entity word in the knowledge base as well as an out-of-vocabulary word for all the models. Without the access to commonsense knowledge, the Seq2Seq model is unable to understand the post because of the out-of-vocabulary word, "*breakable*", thereby generating **OOV** words. MemNet can generate some meaningful words as it reads the triple embeddings in its memory, but still outputs **OOV**. CopyNet can read and copy words from knowledge triples. However, CopyNet generated fewer entity words than ours (see Table 2), as it only deals with separate knowledge triples. Instead, CCM treats the knowledge graph as a whole and encodes more structured, connected information via linked entities and relations. It thus generates more reasonable responses through better use of knowledge. This simple example shows that CCM can generate more appropriate and informative responses than the baselines.

Post	Why are you so **breakable**?
Knowledge	(**glass**, RelatedTo, **breakable**), (**brittle**, RelatedTo, **breakable**), (**rule**, RelatedTo, **breakable**)
Seq2Seq	I'm not a **OOV** , I'm just a **OOV**.
MemNet	I'm not **OOV**. I'm just a really nice person.
CopyNet	I'm not. I'm just a lurker.
CCM	Because I'm a **brittle** man .

Table 4: Sample responses generated by all the models.

5 Conclusion and Future Work

In this paper, we present a commonsense knowledge aware conversational model (CCM) to demonstrate how commonsense knowledge can facilitate language understanding and generation in open-domain conversational systems. Automatic and manual evaluation show that CCM can generate more appropriate and informative responses than state-of-the-art baselines.

As future work, our graph attention mechanisms may inspire other tasks to use commonsense knowledge.

Acknowledgments

This work was partly supported by the National Science Foundation of China under grant No.61272227/61332007 and the National Basic Research Program (973 Program) under grant No. 2013CB329403.

References

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2014.

[Bordes *et al.*, 2013] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *NIPS*, pages 2787–2795, 2013.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van 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*, pages 1724–1734, 2014.

[Eggins and Slade, 2005] Suzanne Eggins and Diana Slade. *Analysing casual conversation*. Equinox Publishing Ltd., 2005.

[Ghazvininejad *et al.*, 2017] Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen-tau Yih, and Michel Galley. A knowledge-grounded neural conversation model. *CoRR*, abs/1702.01932, 2017.

[Gu *et al.*, 2016] Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. Incorporating copying mechanism in sequence-to-sequence learning. In *ACL*, pages 1631–1640, 2016.

[Han *et al.*, 2015] Sangdo Han, Jeesoo Bang, Seonghan Ryu, and Gary Geunbae Lee. Exploiting knowledge base to generate responses for natural language dialog listening agents. In *SIGDIAL*, pages 129–133, 2015.

[Li *et al.*, 2016] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In *NAACL*, pages 110–119, 2016.

[Lin *et al.*, 2017] Hongyu Lin, Le Sun, and Xianpei Han. Reasoning with heterogeneous knowledge for commonsense machine comprehension. In *EMNLP*, pages 2032–2043, 2017.

[Long *et al.*, 2017] Yinong Long, Jianan Wang, Zhen Xu, Zongsheng Wang, Baoxun Wang, and Zhuoran Wang. A knowledge enhanced generative conversational service agent. In *DSTC6 Workshop*, 2017.

[Marková *et al.*, 2007] Ivana Marková, Per Linell, Michèle Grossen, and Anne Salazar Orvig. *Dialogue in focus groups: Exploring socially shared knowledge*. Equinox publishing, 2007.

[Minsky, 1991] Marvin Minsky. Society of mind: a response to four reviews. *Artificial Intelligence*, 48(3):371–396, 1991.

[Mou *et al.*, 2016] Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation. In *COLING*, pages 3349–3358, 2016.

[Ritter *et al.*, 2011] Alan Ritter, Colin Cherry, and William B. Dolan. Data-driven response generation in social media. In *EMNLP*, pages 583–593, 2011.

[Serban *et al.*, 2015] Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. Hierarchical neural network generative models for movie dialogues. *CoRR*, abs/1507.04808, 2015.

[Shang *et al.*, 2015] Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *ACL*, pages 1577–1586, 2015.

[Shao *et al.*, 2017] Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. Generating long and diverse responses with neural conversation models. *CoRR*, abs/1701.03185, 2017.

[Sordoni *et al.*, 2015] Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. In *NAACL*, pages 196–205, 2015.

[Souto, 2015] Patrícia Cristina Nascimento Souto. Creating knowledge with and from the differences: the required dialogicality and dialogical competences. *RAI Revista de Administração e Inovação*, 12(2):60–89, 2015.

[Speer and Havasi, 2012] Robert Speer and Catherine Havasi. Representing general relational knowledge in conceptnet 5. In *LREC*, pages 3679–3686, 2012.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Velickovic *et al.*, 2017] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *CoRR*, abs/1710.10903, 2017.

[Wang *et al.*, 2017] Bingning Wang, Kang Liu, and Jun Zhao. Conditional generative adversarial networks for commonsense machine comprehension. In *IJCAI*, pages 4123–4129, 2017.

[Xing *et al.*, 2017] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. Topic aware neural response generation. In *AAAI*, pages 3351–3357, 2017.

[Xu *et al.*, 2017] Zhen Xu, Bingquan Liu, Baoxun Wang, Chengjie Sun, and Xiaolong Wang. Incorporating loose-structured knowledge into conversation modeling via recall-gate lstm. In *IJCNN*, pages 3506–3513. IEEE, 2017.

[Zhu *et al.*, 2017] Wenya Zhu, Kaixiang Mo, Yu Zhang, Zhangbin Zhu, Xuezheng Peng, and Qiang Yang. Flexible end-to-end dialogue system for knowledge grounded conversation. *CoRR*, abs/1709.04264, 2017.