Incorporating Commonsense Knowledge into Abstractive Dialogue Summarization via Heterogeneous Graph Networks

Xiachong Feng[1], **Xiaocheng Feng**[1,2]*, **Bing Qin**[1,2]
[1]Harbin Institute of Technology, China
[2]Peng Cheng Laboratory, China
{xiachongfeng,xcfeng,bqin}@ir.hit.edu.cn

Abstract

Abstractive dialogue summarization is the task of capturing the highlights of a dialogue and rewriting them into a concise version. In this paper, we present a novel multi-speaker dialogue summarizer to demonstrate how large-scale commonsense knowledge can facilitate dialogue understanding and summary generation. In detail, we consider utterance and commonsense knowledge as two different types of data and design a Dialogue Heterogeneous Graph Network (D-HGN) for modeling both information. Meanwhile, we also add speakers as heterogeneous nodes to facilitate information flow. Experimental results on the SAMSum dataset show that our model can outperform various methods. We also conduct zero-shot setting experiments on the Argumentative Dialogue Summary Corpus, the results show that our model can better generalized to the new domain.

1 Introduction

Automatic summarization is a fundamental task in Natural Language Processing, which aims to condense the original input into a shorter version covering salient information and has been continuously studied for decades (Paice, 1990; Kupiec et al., 1999). Recently, online multi-speaker dialogue/meeting has become one of the most important ways for people to communicate with each other in their daily works. Especially due to the spread of COVID-19 worldwide, people are more dependent on online communication. In this paper, we focus on dialogue summarization, which can help people quickly grasp the core content of the dialogue without reviewing the complex dialogue context.

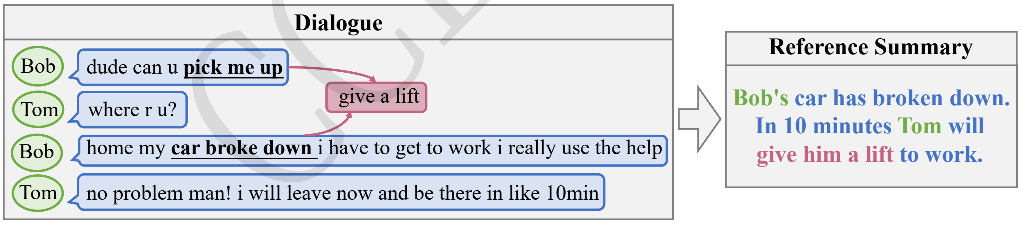

Dialogue
Bob
dude can u pick me up
Tom
where r u?
give a lift
Bob
home my car broke down i have to get to work i really use the help
Tom
no problem man! i will leave now and be there in like 10min
Reference Summary
Bob's car has broken down. In 10 minutes Tom will give him a lift to work.

Figure 1: An example of dialogue-summary pair. Green for speakers, blue for utterances, and pink for commonsense knowledge. In order to generate "give a lift" in the reference summary, the summarization model needs to understand the commonsense knowledge behind "pick up" and "car broke down".

Recent works that incorporate additional commonsense knowledge in the dialogue generation (Zhou et al., 2018) and dialogue context representation learning (Wang et al., 2020) show that even though neural models have strong learning capabilities, explicit knowledge can still improve response generation quality. It is because that a dialog system can understand conversations better and thus respond more properly if it can access and make full use of large-scale commonsense knowledge. However, current dialogue summarization systems (Ganesh and Dingliwal, 2019; Li et al., 2019; Liu et al., 2019a; Zhu et

*Corresponding author.

al., 2020; Chen and Yang, 2020) ignore the exploration of commonsense knowledge, which may limit the performance. In this work, we examine the benefit of incorporating commonsense knowledge in the dialogue summarization task and also address the question of how best to incorporate this information. Figure 1 shows a positive example to illustrate the effectiveness of commonsense knowledge in the dialogue summarization task. Bob asks Tom for help because his car has broken down. On the one hand, by introducing commonsense knowledge according to the *pick up* and *car broke down*, we can know that Bob expects Tom to *give him a lift*. On the other hand, commonsense knowledge can serve as a bridge between non-adjacent utterances that can help the model better understanding the dialogue.

In this paper, we follow the previous setting (Zhou et al., 2018) and also use ConceptNet (Speer and Havasi, 2012) as a large-scale commonsense knowledge base, while the difference is that we regard knowledge and text(utterance) as heterogeneous data in a real multi-speaker dialogue. We propose a model named **D**ialogue **H**eterogeneous **G**raph **N**etwork (D-HGN) for incorporating commonsense knowledge by constructing the graph including both utterance and knowledge nodes. Besides, our heterogeneous graph also contains speaker nodes at the same time, which has been proved to be a useful feature in dialogue modeling. In particular, we equip our heterogeneous graph network with two additional designed modules. One is called message fusion, which is specially designed for utterance nodes to better aggregate information from both speakers and knowledge. The other one is called node embedding, which can help utterance nodes to be aware of position information. Compared to homogeneous graph network in related works (Ganesh and Dingliwal, 2019; Li et al., 2019; Liu et al., 2019a; Zhu et al., 2020), we claim that the heterogeneous graph network can effectively fuse information and contain rich semantics in nodes and links, and thus more accurately encode the dialogue representation.

We conduct experiments on the SAMSum corpus (Gliwa et al., 2019), which is a large-scale chat summarization corpus. We analyze the effectiveness of integration of knowledge and heterogeneity modeling. The human evaluation also shows that our approach can generate more abstractive and correct summaries. To evaluate whether commonsense knowledge can help our model better generalize to the new domain, we also perform zero-shot setting experiments on the Argumentative Dialogue Summary Corpus (Misra et al., 2015), which is a debate summarization corpus. In the end, we give a brief summary of our contributions: (1) We are the first to incorporate commonsense knowledge into dialogue summarization task. (2) We propose a D-HGN model to encode the dialogue by viewing utterances, knowledge and speakers as heterogeneous data. (3) Our model can outperform various methods.

2 Heterogeneous Dialogue Graph Construction

In this section, we describe the graph notation and the graph construction process, which consists of three steps, including (1) utterance-knowledge bipartite graph construction, (2) speaker-utterance bipartite graph construction and (3) heterogeneous dialogue graph construction.

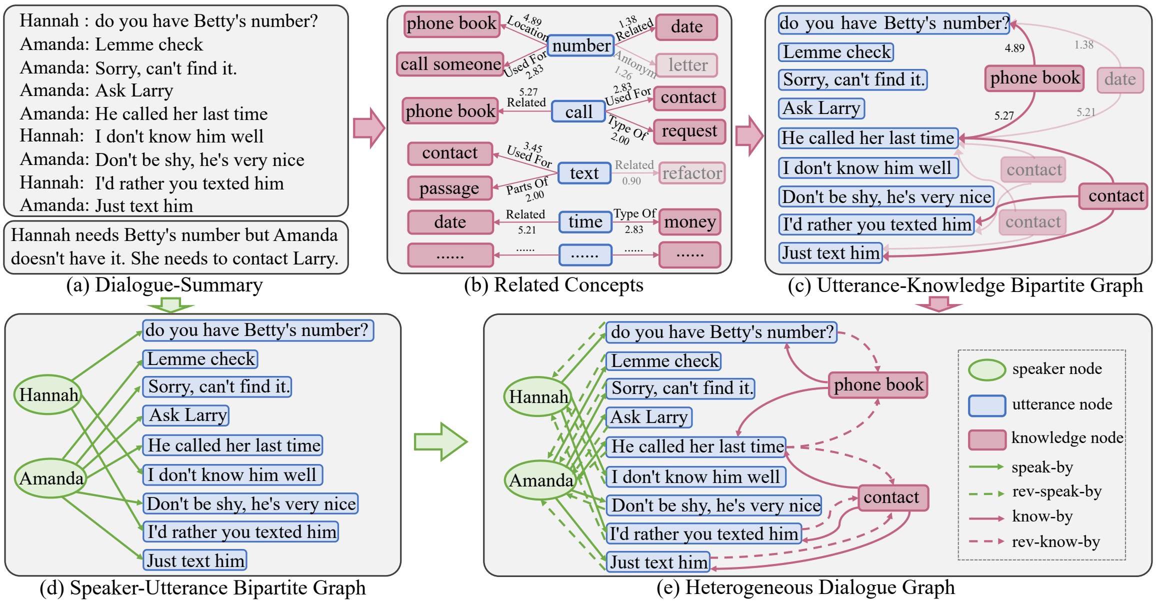
2.1 Graph Notation

Our heterogeneous dialogue graph (HDG) is defined as a directed graph $G = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{R})$, where each node $v \in \mathcal{V}$ and each edge $e \in \mathcal{E}$. Different types of nodes and edges are associated with their type mapping functions $\tau(v) : \mathcal{V} \rightarrow \mathcal{A}$ and $\phi(e) : \mathcal{E} \rightarrow \mathcal{R}$.

2.2 Utterance-Knowledge Bipartite Graph Construction

Current dialogue summarization corpus has no knowledge annotations. To ground each dialogue to commonsense knowledge, we make use of ConceptNet (Speer and Havasi, 2012) to incorporate knowledge. ConceptNet is a semantic network that contains 34 relations in total and represents each knowledge tuple by $R = (h, r, t, w)$ meaning that head concept h and tail concept t have a relation r with a weight of w. It contains not only world facts such as "*Paris is the capital of France*" that are constantly true, but also informal relations that are part of daily knowledge such as "*Call is used for Contact*".

We use each word in the utterance as a query to retrieve a one-hop graph from ConceptNet, as done by Guan et al. (2019). We only consider nouns, verbs, adjectives, and adverbs. We filter out tuples



(a) Dialogue-Summary

(b) Related Concepts

(c) Utterance-Knowledge Bipartite Graph

(d) Speaker-Utterance Bipartite Graph

(e) Heterogeneous Dialogue Graph

Figure 2: Illustration of Heterogeneous Dialogue Graph Construction Process.

where (1) r is in a pre-defined list of useless relations[0] (e.g. “number” is antonym of “letter”), (2) the weight of r is less than 1 (e.g. “text” is related to “refactor”, weight: 0.9). Finally, we can get related concepts for the dialogue, as shown in Figure 2(b). We construct utterance-knowledge bipartite graph by viewing utterances and knowledge as different types of nodes. As shown in Figure 2(c), we connect two utterances to one tail concept t using edge *know-by* if they both have the same tail concept t. Note that two utterances may connect to multiple tail concepts, we choose the one with the highest average weight of relations (e.g. “phone book” is better than “date”). If there are multiple identical knowledge nodes, we also combine them to a single one (e.g. two “contact” nodes are combined into one node).

2.3 Speaker-Utterance Bipartite Graph Construction

Given multiple speakers and corresponding utterances in a dialogue, we construct the speaker-utterance bipartite graph by viewing speakers and utterances as different types of nodes. As shown in Figure 2(d), we construct *speak-by* edges from speakers to utterances based on who said the utterances.

2.4 Heterogeneous Dialogue Graph Construction

We combine the utterance-knowledge bipartite graph and the speaker-utterance bipartite graph as our heterogeneous dialogue graph, as shown in Figure 2(e). Additionally, we add a reverse edge *rev-know-by* and *rev-speak-by* to facilitate information flow over the graph. Finally, there are three types of nodes, where $\mathcal{A}$ becomes *speaker*, *utterance*, and *knowledge* and four types of edges, where $\mathcal{R}$ becomes *speak-by*, *know-by*, *rev-speak-by* and *rev-know-by*.

3 Dialogue Heterogeneous Graph Network

In this section, we describe the details of our dialogue heterogeneous graph network (D-HGN), including three components: node encoder, graph encoder and pointer decoder. The model is shown in Figure 3.

3.1 Node Encoder

The role of node encoder is to give each node $v_i \in \mathcal{V}$ an initial representation $h^0_{v_i}$, where v_i consists of $|v_i|$ words $[w_{i,1}, w_{i,2}, ...w_{i,|v_i|}]$. Note that speaker and knowledge may have multiple words. We employ a Bi-LSTM as the node encoder that encodes input node forwardly and backwardly to generate two

[0] We pre-define the useless relation list, including Antonym, EtymologicallyDerivedFrom, NotHasProperty, DistinctFrom, NotCapableOf, EtymologicallyRelatedTo and NotDesires.

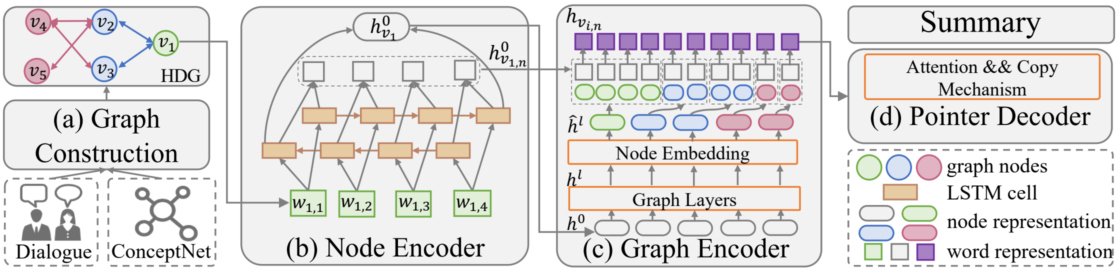


Figure 3: Illustration of our D-HGN model. (a) Graph construction receives a dialogue and ConceptNet and outputs a heterogeneous dialogue graph (HDG). (b) Node encoder receives a sequence of words for a node and produces initial node and word representations. (c) Graph encoder first conducts graph operations for initial node representations. Then a node embedding module is added after graph layers to make nodes to be aware of position information. Finally, the initial word representations and corresponding updated node representations are concatenated as final word representations. (d) Pointer decoder can either generate summary words from the vocabulary or copy from the input words.

sequences of hidden states $\left(\overrightarrow{h_1}, \overrightarrow{h_2}, \ldots, \overrightarrow{h_{|v_i|}}\right)$ and $\left(\overleftarrow{h_1}, \overleftarrow{h_2}, \ldots, \overleftarrow{h_{|v_i|}}\right)$.

$$\begin{aligned}\overrightarrow{h_n} &= \text{LSTM}_f\left(x_n, \overrightarrow{h_{n-1}}\right)\\ \overleftarrow{h_n} &= \text{LSTM}_b\left(x_n, \overleftarrow{h_{n+1}}\right)\end{aligned} \tag{1}$$

x_n denotes the embedding of $w_{i,n}$. The forward and backward hidden states are concatenated as the initial node representation $h^0_{v_i} = [\overrightarrow{h_{|v_i|}}; \overleftarrow{h_1}]$ and initial word representation $h^0_{v_i,n} = [\overrightarrow{h_n}; \overleftarrow{h_n}]$. $h^0_{v_i}$ will be passed to the graph encoder to learn high-level representations. $h^0_{v_i,n}$ will be concatenated with updated node representations to get final word representations.

3.2 Graph Encoder

Graph encoder is used to digest the structural information and get updated node representations. We employ Heterogeneous Graph Transformer (Hu et al., 2020) as our graph encoder, which models heterogeneity by type-dependent parameters and can be easily applied to our graph. It includes: (a) heterogeneous mutual attention, which calculates attention scores $\text{Attn}(s, e, t)$ between source nodes and the target node. (b) heterogeneous message passing, which prepares the message vector $\text{Msg}(s, e, t)$ for each source node and (c) target-specific aggregation, which aggregates messages from source nodes to the target node using attention scores as the weight. Specifically, we design two modules named message fusion and node embedding to make the learning process more effective for our graph.

Heterogeneous Mutual Attention Given an edge $e = (s, t)$ with their node and edge type mapping functions τ and ϕ, we first project source and target node representations from $(l\text{-}1)$-th layer $h_s^{(l-1)}$ and $h_t^{(l-1)}$ into key vector $k_s^{(l)}$ and query vector $q_t^{(l)}$ with type-dependent linear projection.

$$\begin{aligned}k_s^{(l)} &= \text{K\_Linear}^{(l)}_{\tau(s)}\left(h_s^{(l-1)}\right)\\ q_t^{(l)} &= \text{Q\_Linear}^{(l)}_{\tau(t)}\left(h_t^{(l-1)}\right)\end{aligned} \tag{2}$$

Next, to integrate edge type information, we calculate unnormalized score $\alpha(s, e, t)$ between t and s by adding a edge-based matrix $W^{ATT}_{(l),\phi(e)}$. Finally, for each target node t, we conduct Softmax for all $s \in N(t)$ to get the final normalized attention scores $\text{Attn}^{(l)}(s, e, t)$, where $N(t)$ denotes neighbors of target node t. Note that if target node is of utterance type and source node is of speaker type, we do not calculate the attention score between these two types of nodes. See more detail at *message fusion*

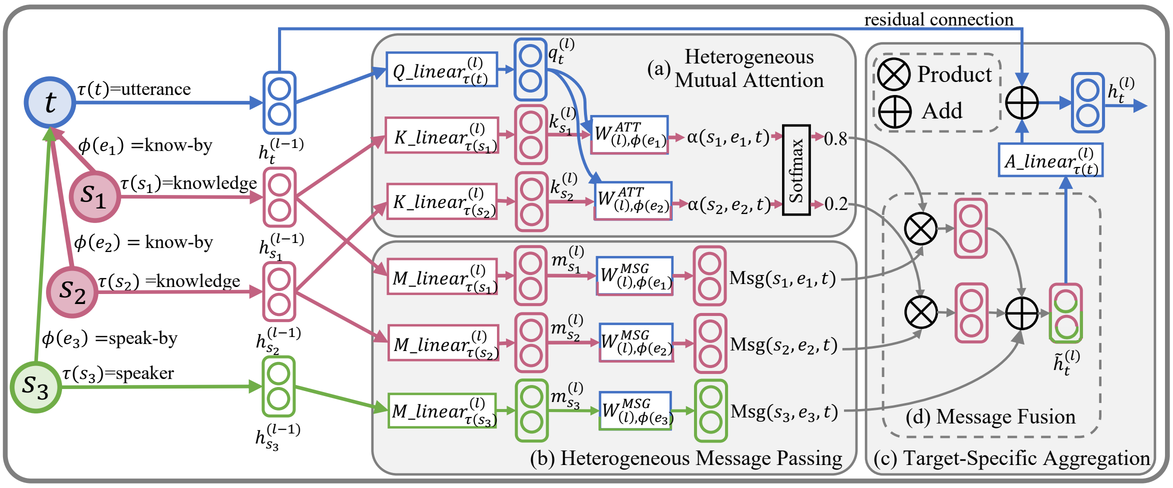

residual connection
t
$\tau(t)$=utterance
$h_t^{(l-1)}$
$\phi(e_1)$ =know-by
s_1
$\tau(s_1)$=knowledge
$h_{s_1}^{(l-1)}$
$\phi(e_2)$ = know-by
s_2
$\tau(s_2)$ =knowledge
$h_{s_2}^{(l-1)}$
$\phi(e_3)$ =speak-by
s_3
$\tau(s_3)$=speaker
$h_{s_3}^{(l-1)}$
$Q\_linear_{\tau(t)}^{(l)}$
$q_t^{(l)}$
(a) Heterogeneous Mutual Attention
$K\_linear_{\tau(s_1)}^{(l)}$
$k_{s_1}^{(l)}$
$W_{(l),\phi(e_1)}^{ATT}$
$\alpha(s_1,e_1,t)$
$K\_linear_{\tau(s_2)}^{(l)}$
$k_{s_2}^{(l)}$
$W_{(l),\phi(e_2)}^{ATT}$
$\alpha(s_2,e_2,t)$
Softmax
0.8
0.2
$M\_linear_{\tau(s_1)}^{(l)}$
$m_{s_1}^{(l)}$
$W_{(l),\phi(e_1)}^{MSG}$
$\mathrm{Msg}(s_1,e_1,t)$
$M\_linear_{\tau(s_2)}^{(l)}$
$m_{s_2}^{(l)}$
$W_{(l),\phi(e_2)}^{MSG}$
$\mathrm{Msg}(s_2,e_2,t)$
$M\_linear_{\tau(s_3)}^{(l)}$
$m_{s_3}^{(l)}$
$W_{(l),\phi(e_3)}^{MSG}$
$\mathrm{Msg}(s_3,e_3,t)$
(b) Heterogeneous Message Passing
⊗ Product
⊕ Add
$h_t^{(l)}$
$A\_linear_{\tau(t)}^{(l)}$
$\tilde{h}_t^{(l)}$
(d) Message Fusion
(c) Target-Specific Aggregation

Figure 4: Illustration of one graph layer. Given a target node of utterance type and source nodes of knowledge and speaker type. Firstly, we use (a) heterogeneous mutual attention to calculate the attention scores by type-dependent linear projection. Secondly, we use (b) heterogeneous message passing to prepare the message vector for each source node. Thirdly, we use (c) target-specific aggregation to aggregate messages to the target node. Specifically, we propose a message fusion module that uses attention scores as the weight to average the knowledge vectors and add speaker information additionally.

module. The process is shown in Figure 4(a).

$$\begin{aligned}\alpha(s,e,t) &= \left(k_s^{(l)} W_{(l),\phi(e)}^{ATT} {q_t^{(l)}}^{\top}\right)\\ \mathrm{Attn}^{(l)}(s,e,t) &= \underset{\forall s\in N(t)}{\mathrm{Softmax}}\left(\alpha(s,e,t)\right)\end{aligned} \tag{3}$$

Heterogeneous Message Passing We first project source node representation $h_s^{(l-1)}$ into the vector $m_s^{(l)} = \mathrm{M\_Linear}_{\tau(s)}^{(l)}\left(h_s^{(l-1)}\right)$ with type-dependent linear projection and then followed by a edge-based matrix $W_{(l),\phi(e)}^{MSG}$ to get the message vector. The process is shown in Figure 4(b).

$$\mathrm{Msg}^{(l)}(s,e,t) = m_s^{(l)} W_{(l),\phi(e)}^{MSG} \tag{4}$$

Target-Specific Aggregation We divide this process into two cases based on the type of target node: (1) $\tau(t) \neq utterance$, (2) $\tau(t) = utterance$. For the first case, We use attention vector as the weight to average messages: $\widetilde{h}_t^{(l)} = \oplus_{\forall s\in N(t)}\left(\mathrm{Attn}^{(l)}(s,e,t) \otimes \mathrm{Msg}^{(l)}(s,e,t)\right)$. For the second case, we design a Message Fusion module to aggregate messages to utterance node more effectively. After getting aggregated message vector $\widetilde{h}_t^{(l)}$, we maps it back to $\tau(t)$-type distribution with a linear projection followed by residual connection to get the updated representation $h_t^{(l)}$, as shown in Figure 4(c).

$$h_t^{(l)} = \mathrm{A\_Linear}_{\tau(t)}^{(l)}\left(\mathrm{Sigmoid}\left(\widetilde{h}_t^{(l)}\right)\right) + h_t^{(l-1)} \tag{5}$$

Message Fusion Dialogue summaries often describe "*who did what*", thus speaker information is required for utterances. However, if target node of utterance type aggregates messages from source nodes of knowledge and speaker type, it will prefer more to the speaker node while giving up using knowledge nodes, since attention is a normalized distribution. Therefore, in our message fusion module, we use attention weights for knowledge nodes to average corresponding messages and add speaker information additionally. The process is shown in Figure 4(d).

$$\begin{aligned}&\mathrm{s_k} = (\forall s \in N(t) \wedge \tau(s) = knowledge), \mathrm{s_s} = (\forall s \in N(t) \wedge \tau(s) = speaker)\\ &\widetilde{h}_t^{(l)} = \underset{s\in \mathrm{s_k}}{\oplus}\left(\mathrm{Attn}^{(l)}(s,e,t) \otimes \mathrm{Msg}^{(l)}(s,e,t)\right) + \mathrm{Msg}^{(l)}(\mathrm{s_s},e,t)\end{aligned} \tag{6}$$

Node Embedding In this section, a module named Node Embedding is designed to make utterance nodes to be aware of position information in source dialogue. This is because original heterogeneous graph cannot directly model the chronological order between utterances, while an ideal dialogue summary needs to refer to the order of corresponding dialogue utterances. In detail, for speaker and knowledge nodes, we fix their position to 0. For each utterance node v_i, it associates with a position p_{v_i}, which is the ranking of utterances in the original dialogue. As shown in Figure 3(c), we add position information for each node: $\hat{h}_{v_i}^{(l)} = h_{v_i}^{(l)} + W^{pos}[p_{v_i}]$, where W^{pos} denotes a learnable node embedding matrix. After getting the output representation $\hat{h}^{(l)}$ for each node, we concatenate updated node representation $\hat{h}_{v_i}^{(l)}$ and corresponding initial word representations $h_{v_i,n}^{0}$ followed by a linear projection F_Linear to get final word representations $h_{v_i,n}$.

$$h_{v_i,n} = \text{F\_Linear}([\hat{h}_{v_i}^{(l)}; h_{v_i,n}^{0}]) \tag{7}$$

3.3 Pointer Decoder

We employ a LSTM with attention and copy mechanism to generate summaries. At each decoding time step t, the LSTM reads the previous word embedding x_{t-1} and previous context vector c_{t-1} as inputs to compute the new hidden state $s_t = \text{LSTM}\left(x_{t-1}, c_{t-1}, s_{t-1}\right)$. We use the average of all word representations s_0 in the graph to initialize the decoder.

$$s_0 = \text{Average}(\sum\nolimits_{v_i \in G} \sum\nolimits_{n \in [1,|v_i|]} h_{v_i,n}) \tag{8}$$

The context vector c_t is computed as in Bahdanau et al. (2015), which is then used to calculate generation probability p_{gen} and the final probability distribution $P(w)$, as done by See et al. (2017).

3.4 Training

For each heterogeneous dialogue graph G that is paired with a ground truth summary $Y^* = [y_1^*, y_2^*, ..., y_{|Y^*|}^*]$, we minimize the negative log-likelihood of the target words sequence.

$$L = -\sum_{t=1}^{|Y^*|} \log p\left(y_t^* | y_1^* \ldots y_{t-1}^*, G\right) \tag{9}$$

4 Experiments

Dataset Following the latest works (Gliwa et al., 2019; Ganesh and Dingliwal, 2019), we conduct experiments on two different settings. Firstly, we train and evaluate our model on the SAMSum corpus (Gliwa et al., 2019), which contains dialogues around chit-chats topics. Secondly, we train using SAMSum corpus and use the Argumentative Dialogue Summary Corpus (ADSC) (Misra et al., 2015) as the test set to perform zero-shot setting experiments. Each dialogue in ADSC dataset owns 5 different summaries and is mainly around debate topics. Table 1 shows the knowledge related statistics of two datasets.

Dataset	Split	#	Coverage	Average Know
SAMSum	Train	14732	94.43%	19.60
	Valid	818	95.72%	18.23
	Test	819	93.89%	19.77
ADSC	Full	45	100%	6.50

Table 1: Knowledge related statistics on SAMSum and ADSC datasets. # is the number of dialogues. Coverage represents the percentage of dialogues with at least one knowledge node. Average Know represents the average number of knowledge nodes per dialogue.

Implementation Details The word embedding size is set to 100 and initialized with the pre-trained GloVe vector. The dimension of node encoder and pointer decoder is set to 300. The dimension of graph encoder is set to 200. The graph layer number is set to 1. Dropout is set to 0.5. We use Adam with the learning rate of 0.001 and use gradient clipping with a maximum gradient norm of 2. In the test process, beam size is set to 10, minimum decoded length is 19 [1].

Evaluation Metrics We employ the standard F_1 scores for ROUGE-1, ROUGE-2, and ROUGE-L metrics (Lin, 2004) to measure summary qualities. These three metrics evaluate the accuracy on unigrams, bigrams, and longest common subsequence between the groundtruth and the generated summary [2].

Baseline Models We compare our model with several baselines.

- **LONGEST-3** chooses the longest three utterances as the summary.
- **TextRank** (Mihalcea and Tarau, 2004) is a graph-based extractive method.
- **SummaRunner** (Nallapati et al., 2017) extract utterances based on a hierarchical RNN model.
- **Transformer** (Vaswani et al., 2017) is a Seq2Seq model that utilizes self-attention operations.
- **PGN** (See et al., 2017) is a Seq2Seq model equipped with copy mechanism.
- **HRED** (Serban et al., 2016) is a hierarchical Seq2Seq model.
- **Abs RL** (Chen and Bansal, 2018) is a pipeline model that first selects salient utterances based on a extractive model then produces the summary based on a abstractive model using diversity beam search. The extractive model is trained using utterance-level extraction labels. The overall model is jointly trained using reinforcement learning.
- **Abs RL Enhance** (Gliwa et al., 2019) is based on **Abs RL**, which appends all speakers after each utterance, because the original model may select utterances of a single speaker that will lead to no other speaker information.
- **D-GAT**, **D-GCN** and **D-RGCN** are variants of our model that replace heterogeneous graph layers with homogeneous graph layers, including GAT (Velickovic et al., 2018), GCN (Kipf and Welling, 2017) and RGCN (Schlichtkrull et al., 2018) [3].

4.1 Automatic Evaluation

Table 2 shows the results on SAMSum corpus. The D-HGN stands for our full model, which outperforms various baselines. Compared with HRED that uses no additional auxiliary information such as commensence knowledge or utterance-level extraction labels, D-RGCN that uses commensence knowledge can achieve 0.97% improvement on ROUGE-1, 0.94% on ROUGE-2, 1.28% on ROUGE-L, which shows the effectiveness of knowledge integration. Compared with homogeneous networks like D-RGCN, D-HGN that based on heterogeneous graph networks can achieve 0.67% improvement on ROUGE-1, 1.00% on ROUGE-2, 0.63% on ROUGE-L, which verifies the effectiveness of heterogeneity modeling.

4.2 Human Evaluation

We conduct human evaluation to verify the quality of the generated summaries, including abstractiveness (contains higher-level conceptual words), informativeness (covers adequate information) and correctness (associates right names with actions). We randomly sample 50 dialogues with corresponding generated summaries to conduct the evaluation. We hired five graduates to perform human evaluation. For each metric, the score ranges from 1 (worst) to 5 (best). The results are shown in Table 3.

[1] Our codes are available at: `https://github.com/xcfcode/DHGN`.

[2] https://pypi.org/project/pyrouge/

[3] Note that D-GAT also use message fusion module to update representations for utterance nodes.

Type	Model	Know.	Heter.	Utter.	RL	R-1	R-2	R-L
Extractive	LONGEST-3	✗	✗	✗	✗	32.46	10.27	29.92
	TextRank	✗	✗	✗	✗	29.27	8.02	28.78
	SummaRunner	✗	✗	✗	✗	33.76	10.28	28.69
Abstractive	Transformer	✗	✗	✗	✗	36.62	11.18	33.06
	PGN	✗	✗	✗	✗	40.08	15.28	36.63
	HRED	✗	✗	✗	✗	40.39	16.13	37.65
Pipeline	Abs RL	✗	✗	✓	✓	40.96	17.18	39.05
	AbsRL Enhance	✗	✗	✓	✓	41.95	18.06	39.23
Ours	D-GCN	✓	✗	✗	✗	41.33	16.98	38.70
	D-GAT	✓	✗	✗	✗	41.08	16.89	38.61
	D-RGCN	✓	✗	✗	✗	41.36	17.07	38.93
	D-HGN	✓	✓	✗	✗	**42.03**	**18.07**	**39.56**

Table 2: Test set results on the SAMSum Dataset, where “R-1” is short for “ROUGE-1”, “R-2” for “ROUGE-2”, “R-L” for “ROUGE-L”. “Know.”, “Heter.”, “Utter.” and “RL” indicate whether knowledge, heterogeneity modeling, utterance-level extraction labels and reinforcement learning are used or not.

Model	Abstractiveness	Informativeness	Correctness
PGN	2.70	2.68	2.49
AbsRL Enhance	2.94	3.23	2.43
D-HGN	**3.26**	**3.25**	**2.92**
w/o *knowledge*	3.09	3.16	2.80
w/o *speaker*	3.23	3.21	2.60

Table 3: Human evaluation results.

Our model achieves higher scores. Compared with D-HGN, D-HGN(w/o knowledge) gets a lower score in abstractiveness, which indicates knowledge incorporation can help our model express deeper meanings. D-HGN(w/o speaker) performs worse than D-HGN in correctness, which shows effectiveness of heterogeneity modeling by viewing speakers as heterogeneous data. AbsRL Enhance performs worst in correctness, which may due to the utterances extraction will break the coherence of dialogue contexts.

4.3 Ablation Study

We conduct two types of ablation studies to verify the effectiveness of different types of nodes and two modules we propose. As shown in Table 4, without knowledge integration(w/o knowledge), the model suffers the performance drop, which shows incorporating knowledge can help our model better modeling the dialogue context. For speaker nodes, directly remove them in the graph will lead to no speaker in the final summary. Instead, we append the speakers in front of utterances(w/o speaker). The results show that modeling speakers as heterogeneous data will do good the final summary generation process.

Model	ROUGE-1	ROUGE-2	ROUGE-L
D-HGN	**42.03**	**18.07**	**39.56**
w/o *message fusion*	41.29	17.09	38.74
w/o *node embedding*	41.99	17.85	38.89

Table 4: Ablation Study for Two Modules

As shown in Table 5, we remove the message fusion module(w/o message fusion), the results show that it is worth to design specific message fusion method according to different types of nodes. Besides, without taking position information into account(w/o node embedding), our model will lose some performance.

Model	ROUGE-1	ROUGE-2	ROUGE-L
D-HGN	**42.03**	**18.07**	**39.56**
w/o *knowledge*	41.52	17.38	38.76
w/o *speaker*	41.06	17.17	38.92

Table 5: Ablation Study for Different Types of Nodes

4.4 Zero-shot Setting

To verify whether knowledge can help our model better generalize to the new domain, we directly test models on the ADSC Corpus. The results are shown in Table 6. The homogeneous model D-GAT that uses knowledge can get better results than other baselines. The D-HGN gets the best score. We contribute this to the fact that knowledge can help our models better understand the dialogue in the new domain.

Model	ROUGE-1	ROUGE-2	ROUGE-L
PGN	28.69	4.77	22.39
AbsRL Enhance	30.00	4.87	22.27
D-GAT	32.90	5.46	22.47
D-HGN	**33.55**	**5.68**	**22.75**

Table 6: ROUGE F_1 results on the Argumentative Dialogue Summary Corpus.

4.5 Visualization

To examine whether our D-HGN can learn easily distinguishable representations, we extract node representations from the last graph layer for the SAMSum test set. We apply t-SNE (van der Maaten, 2014) to these vectors. The results are shown in Figure 5. We find that our model can generate more discrete and easily distinguishable representations. Besides, D-GAT also tends to separate representations of different types of nodes, which indicates explicitly heterogeneity modeling is a more reasonable approach.

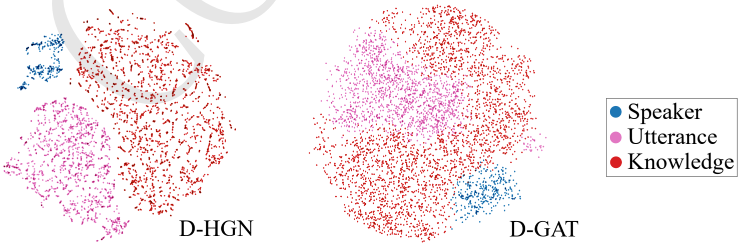
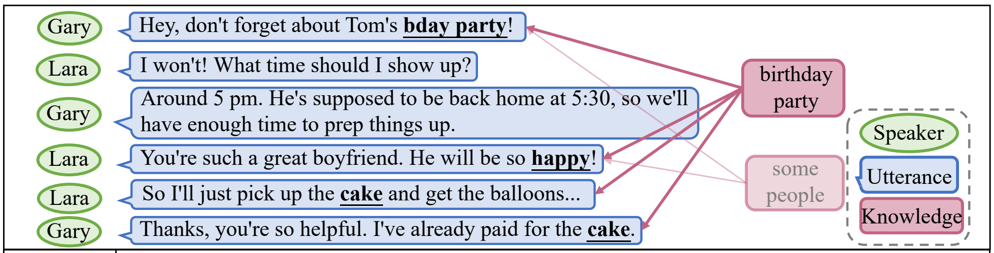


Figure 5: Visualization of node representations generated by the last graph layer of D-HGN and D-GAT.

4.6 Case Study

Figure 6 shows summaries generated by different models and the visualization of knowledge-to-utterance attention weights learned by our D-HGN model, the darker the color, the higher the weights. Our model incorporates two knowledge nodes, one is *birthday party* according to "bday party", "happy" and "cake", the other one is *some people* according to "Tom" and "boyfriend". We can see that our D-HGN model pays more attention to *birthday party* rather than *some people*. On the one hand, incorporating *birthday party* helps our model generate a more formal summary (using birthday rather than bday). On the other hand, *birthday party* connects non-adjacent utterances around the birthday topic, which helps our model generate a more informative and detailed summary (including cake).


Gary
Hey, don't forget about Tom's **bday party**!
Lara
I won't! What time should I show up?
Gary
Around 5 pm. He's supposed to be back home at 5:30, so we'll have enough time to prep things up.
Lara
You're such a great boyfriend. He will be so **happy**!
Lara
So I'll just pick up the **cake** and get the balloons...
Gary
Thanks, you're so helpful. I've already paid for the **cake**.
birthday party
some people
Speaker
Utterance
Knowledge

PGN	Gary and Lara will meet at 5 pm for Tom's **bday party.**
AbsRL Enhance	Gary and Lara are supposed to be back Tom's home at 5 pm for his **bday party.**
D-HGN	Gary and Lara are going to Tom's **birthday party** at 5 pm. Lara will pick up the **cake**.
Golden	It's Tom's **birthday**. Lara and Gary will come to Tom's place about 5 pm to prepare everything. Gary has already paid for the **cake** Lara will pick it.

Figure 6: Example summaries generated by different models for one dialogue.

5 Related Work

Previous works used feature engineering (Xie et al., 2008), template-based (Oya et al., 2014) and graph-based (Bui et al., 2009) methods for extractive dialogue summarization. Although extractive methods are widely used, the results tend to be incoherent and poorly readable. Therefore, current works mainly focus on abstractive methods, which can produce more readable and fluency summaries. They tend to incorporate additional auxiliary information to help better modeling the dialogue. Goo and Chen (2018) incorporated dialogue acts to model the interactive status of the meeting. Liu et al. (2019a) tackled the problem of customer service summarization, which first produced a sequence of pre-defined keywords then generated the summary. Liu et al. (2019b) generated summaries for nurse-patient conversation by incorporating topic information. Ganesh and Dingliwal (2019) first removed useless utterances by utilizing discourse labels and then generated summaries. Li et al. (2019) combined vision and textual features in a unified hierarchical attention framework to generate meeting summaries. Zhu et al. (2020) employed a hierarchical transformer framework and incorporated part-of-speech and entity information for meeting summarization. Chen and Yang (2020) incorporated topic and stage information to model the dialogue. Zhao et al. (2020) used topic words to alleviate the factual inconsistency problem. Feng et al. (2020) used dialogue discourse to model the interaction between utterances. In this paper, we facilitate dialogue summarization task by incorporating commonsense knowledge and further model utterances, commonsense knowledge and speakers as heterogeneous data.

6 Conclusion

In this paper, we improve abstractive dialogue summarization by incorporating commonsense knowledge. We first construct a heterogeneous dialogue graph by introducing knowledge from a large-scale commonsense knowledge base. Then we present a Dialogue Heterogeneous Graph Network (D-HGN) for this task by viewing utterances, knowledge and speakers in the graph as heterogeneous nodes. We additionally design two modules named message fusion and node embedding to facilitate information flow. Experiments on the SAMSum dataset show the effectiveness of our model that can outperform various methods. Zero-shot setting experiments on the Argumentative Dialogue Summary Corpus show that our model can better generalized to the new domain.

Acknowledgments

This work is supported by the National Key R&D Program of China via grant 2020AAA0106502 and National Natural Science Foundation of China (NSFC) via grant 61906053 and Natural Science Foundation of Heilongjiang via grant YQ2019F008.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Trung Bui, Matthew Frampton, John Dowding, and Stanley Peters. 2009. Extracting decisions from multi-party dialogue using directed graphical models and semantic similarity. In *Proceedings of the SIGDIAL 2009 Conference*, pages 235–243, London, UK. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–686, Melbourne, Australia. Association for Computational Linguistics.

Jiaao Chen and Diyi Yang. 2020. Multi-view sequence-to-sequence models with conversational structure for abstractive dialogue summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4106–4118, Online. Association for Computational Linguistics.

Xiachong Feng, Xiaocheng Feng, Bing Qin, Xinwei Geng, and Ting Liu. 2020. Dialogue discourse-aware graph convolutional networks for abstractive meeting summarization. *arXiv preprint arXiv:2012.03502*.

Prakhar Ganesh and Saket Dingliwal. 2019. Abstractive summarization of spoken and written conversation. *arXiv preprint arXiv:1902.01615*.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.

Chih-Wen Goo and Yun-Nung Chen. 2018. Abstractive dialogue summarization with sentence-gated modeling optimized by dialogue acts. In *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 735–742. IEEE.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6473–6480. AAAI Press.

Ziniu Hu, Yuxiao Dong, Kuansan Wang, and Yizhou Sun. 2020. Heterogeneous graph transformer. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2704–2710. ACM / IW3C2.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1999. A trainable document summarizer. *Advances in Automatic Summarization*, pages 55–60.

Manling Li, Lingyu Zhang, Heng Ji, and Richard J. Radke. 2019. Keep meeting summaries on topic: Abstractive multi-modal meeting summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2190–2196, Florence, Italy. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chunyi Liu, Peng Wang, Jiang Xu, Zang Li, and Jieping Ye. 2019a. Automatic dialogue summary generation for customer service. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 1957–1965. ACM.

Zhengyuan Liu, Angela Ng, Sheldon Lee, Ai Ti Aw, and Nancy F Chen. 2019b. Topic-aware pointer-generator networks for summarizing spoken conversations. *arXiv preprint arXiv:1910.01335*.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain. Association for Computational Linguistics.

Amita Misra, Pranav Anand, Jean E. Fox Tree, and Marilyn Walker. 2015. Using summarization to discover argument facets in online idealogical dialog. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 430–440, Denver, Colorado. Association for Computational Linguistics.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3075–3081. AAAI Press.

Tatsuro Oya, Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. A template-based abstractive meeting summarization: Leveraging summary and source text relationships. In *Proceedings of the 8th International Natural Language Generation Conference (INLG)*, pages 45–53, Philadelphia, Pennsylvania, U.S.A. Association for Computational Linguistics.

Chris D Paice. 1990. Constructing literature abstracts by computer: techniques and prospects. *Information Processing & Management*, 26(1):171–186.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*, pages 593–607. Springer.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Robyn Speer and Catherine Havasi. 2012. Representing general relational knowledge in ConceptNet 5. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 3679–3686, Istanbul, Turkey. European Language Resources Association (ELRA).

Laurens van der Maaten. 2014. Accelerating t-sne using tree-based algorithms. *J. Mach. Learn. Res.*, 15:3221–3245.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Tianyi Wang, Yating Zhang, Xiaozhong Liu, Changlong Sun, and Qiong Zhang. 2020. Masking orchestration: Multi-task pretraining for multi-role dialogue representation learning.

Shasha Xie, Yang Liu, and Hui Lin. 2008. Evaluating the effectiveness of features and sampling in extractive meeting summarization. In *2008 IEEE Spoken Language Technology Workshop*. IEEE.

Lulu Zhao, Weiran Xu, and Jun Guo. 2020. Improving abstractive dialogue summarization with graph structures and topic words. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 437–449, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Hao Zhou, Tom Young, Minlie Huang, Haizhou Zhao, Jingfang Xu, and Xiaoyan Zhu. 2018. Commonsense knowledge aware conversation generation with graph attention. In Jérôme Lang, editor, *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4623–4629. ijcai.org.

Chenguang Zhu, Ruochen Xu, Michael Zeng, and Xuedong Huang. 2020. A hierarchical network for abstractive meeting summarization with cross-domain pretraining. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 194–203.