Self- and Pseudo-self-supervised Prediction of Speaker and Key-utterance for Multi-party Dialogue Reading Comprehension

Yiyang Li[1,2,3] and **Hai Zhao**[1,2,3,*]
[1] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2] Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University
[3] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
eric-lee@sjtu.edu.cn, zhaohai@cs.sjtu.edu.cn

Abstract

Multi-party dialogue machine reading comprehension (MRC) brings tremendous challenge since it involves multiple speakers at one dialogue, resulting in intricate speaker information flows and noisy dialogue contexts. To alleviate such difficulties, previous models focus on how to incorporate these information using complex graph-based modules and additional manually labeled data, which is usually rare in real scenarios. In this paper, we design two labour-free self- and pseudo-self-supervised prediction tasks on speaker and key-utterance to implicitly model the speaker information flows, and capture salient clues in a long dialogue. Experimental results on two benchmark datasets have justified the effectiveness of our method over competitive baselines and current state-of-the-art models.

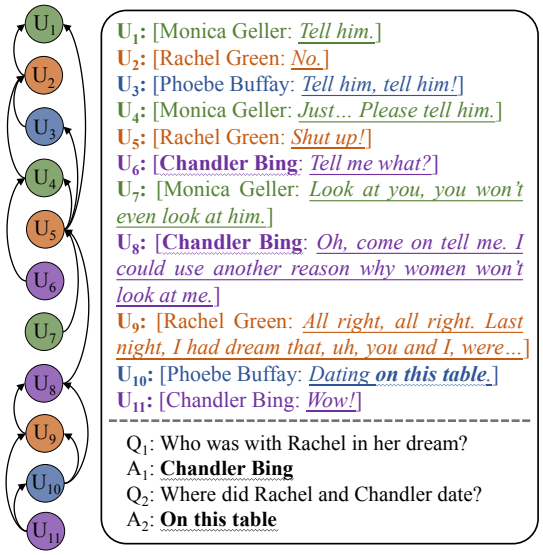


Figure 1: Right part: A dialogue and its corresponding questions from FriendsQA, whose answers are marked with wavy lines. Left part: The speaker information flows of this dialogue.

1 Introduction

Dialogue machine reading comprehension (MRC, Hermann et al., 2015) aims to teach machines to understand dialogue contexts so that solves multiple downstream tasks (Yang and Choi, 2019; Li et al., 2020; Lowe et al., 2015; Wu et al., 2017; Zhang et al., 2018). In this paper, we focus on question answering (QA) over dialogue, which tests the capability of a model to understand a dialogue by asking it questions with respect to the dialogue context. QA over dialogue is of more challenge than QA over plain text (Rajpurkar et al., 2016; Reddy et al., 2019; Yang and Choi, 2019) owing to the fact that conversations are full of informal, colloquial expressions and discontinuous semantics. Among this, multi-party dialogue brings even more tremendous challenge compared to two-party dialogue (Sun et al., 2019; Cui et al., 2020) since it involves multiple speakers at one dialogue, resulting in complicated discourse structure (Li et al., 2020) and intricate speaker information flows. Besides this, Zhang et al. (2021) also pointed that for long dialogue contexts, not all utterances contribute to the final answer prediction since a lot of them are noisy and carry no useful information.

To illustrate the challenge of multi-party dialogue MRC, we extract a dialogue example from FriendsQA dataset (Yang and Choi, 2019) which is shown in Figure 1. This single dialogue involves four different speakers with intricate speaker information flows. The arrows here represent the direction of information flows, from senders to receivers. Let us consider the reasoning process of Q_1: a model should first notice that it is *Rachel* who had a dream and locate U_9, then solve the coreference resolution problem that *I* refers to *Rachel* and *you* refers to *Chandler*. This coreference knowledge must be obtained by considering the information

*Corresponding author. This paper was partially supported by Key Projects of National Natural Science Foundation of China (U1836222 and 61733011).

Findings of the Association for Computational Linguistics: EMNLP 2021, pages 2053–2063
November 7–11, 2021.

flow from U_9 to U_8, which means *Rachel* speaks to *Chandler*. Q_2 follows a similar process, a model should be aware of that U_{10} is a continuation of U_9 and solves the above coreference resolution problem as well.

To tackle the aforementioned obstacles, we design a self-supervised speaker prediction task to implicitly model the speaker information flows, and a pseudo-self-supervised key-utterance prediction task to capture salient utterances in a long and noisy dialogue. In detail, the self-supervised speaker prediction task guides a carefully designed Speaker Information Decoupling Block (SIDB, introduced in Section 3.4) to decouple speaker-aware information, and the key-utterance prediction task guides a Key-utterance Information Decoupling Block (KIDB, introduced in Section 3.3) to decouple key-utterance-aware information. We finally fuse these two kinds of information and make final span prediction to get the answer of a question.

To sum up, the main contributions of our method are three folds:

- We design a novel self-supervised speaker prediction task to better capture the indispensable speaker information flows in multi-party dialogue. Compared to previous models, our method requires no additional manually labeled data which is usually rare in real scenarios.
- We design a novel key-utterance prediction task to capture key-utterance information in a long dialogue context and filter noisy utterances.
- Experimental results on two benchmark datasets show that our model outperforms strong baselines by a large margin, and reaches comparable results to the current state-of-the-art models even under the condition that they utilized additional labeled data.

2 Related work

2.1 Pre-trained Language Models

Recently, pre-trained language models (PrLMs), like BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019), XLNet (Yang et al., 2019) and ELECTRA (Clark et al., 2020), have reached remarkable achievements in learning universal natural language representations by pre-training large language models on massive general corpus and fine-tuning them on downstream tasks (Socher et al., 2013; Wang et al., 2018; Wang et al., 2019; Lai et al., 2017). We argue that the self-attention mechanism (Vaswani et al., 2017) in PrLMs is in essence a variant of Graph Attention Network (GAT, Veličković et al., 2017), which has an intrinsic capability of exchanging information. Compared to vanilla GAT, a Transformer block consisting of residual connection (He et al., 2016) and layer normalization (Ba et al., 2016) is more stable in training. Hence, it is chosen as the basic architecture of our SIDB (Section 3.4) and KIDB (Section 3.3) instead of vanilla GAT.

2.2 Multi-party Dialogue Modeling

There are several previous works that study multi-party dialogue modeling on different downstream tasks such as response selection and dialogue emotion recognition. Hu et al. (2019) utilize the *response to (@)* labels and a Graph Neural Network (GNN) to explicitly model the speaker information flows. Wang et al. (2020) design a pre-training task named Topic Prediction to equip PrLMs with the ability of tracking parallel topics in a multi-party dialogue. Jia et al. (2020) make use of an additional labeled dataset to train a dependency parser, then utilize the dependency parser to disentangle parallel threads in multi-party dialogues. Ghosal et al. (2019) propose a window-based heterogeneous Graph Convolutional Network (GCN) to model the emotion flow in multi-party dialogues.

2.3 Speaker Information Incorporation

In dialogue MRC, speaker information plays a significant role in comprehending the dialogue context. In the latest studies, Liu et al. (2021) propose a Mask-based Decoupling-Fusing Network (MDFN) to decouple speaker information from dialogue contexts, by adding inter-speaker and intra-speaker masks to the self-attention blocks of Transformer layers. However, their approach is restricted to two-party dialogue since they have to specify the sender and receiver roles of each utterance. Gu et al. (2020) propose Speaker-Aware BERT (SA-BERT) to capture speaker information by adding speaker embedding at token representation stage of the Transformer architecture, then pre-train the model using next sentence prediction (NSP) and masked language model (MLM) losses. Nonetheless, their speaker embedding lacks of well-designed pre-training task to refine, resulting in inadequate speaker-specific information. Different from previous models, our model is suitable for the more challenging multi-party dialogue and is equipped with carefully-designed task to better capture the speaker information.

3 Methodology

In this part, we will formulate our task and present our proposed model as shown in Figure 2. There are four main parts in our model, a shared Transformer encoder, a key-utterance information decoupling block, a speaker information decoupling block and a final fusion-prediction layer. In the following sections, we will introduce these modules in detail.

3.1 Task Formulation

Let $\mathbb{C} = \{U_1, U_2, ..., U_N\}$ be a dialogue context with N utterances. Each utterance $U_i = \{S_i, W_i\}$ consists of a speaker S_i specified by a name and a sequence of words W_i speaker S_i utters. W_i can be denoted as a l_i-length sequence $\{w_{i1}, w_{i2}, ..., w_{il_i}\}$. Let a question corresponds to the dialogue context be $\mathbb{Q} = \{q_1, q_2, ..., q_L\}$, where L is the length of the question and each q_i is a token of the question. Given $\mathbb{C}$ and $\mathbb{Q}$, a dialogue MRC model is required to find an answer a for the question, which is restricted to be a continuous span of the dialogue context. In some datasets, a can be an empty string indicating that there is no answer to the question according to the dialogue context.

3.2 Shared Transformer Encoder

To fully utilize the powerful representational ability of PrLMs, we employ a *pack* and *separate* method as Zhang et al. (2021), which is supposed to take advantage of the deep Transformer blocks to make the context and question better interacted with each other. We first pack the context and question as a joint input to feed into the Transformer blocks and separate them according to the position for further interaction.

Given the dialogue context $\mathbb{C}$ and a corresponding question $\mathbb{Q}$, we pack them to form a sequence: $\mathbb{X}$ = {[CLS]$\mathbb{Q}$[SEP]$\mathrm{S_1}$:$\mathrm{U_1}$[SEP]... $\mathrm{S_N}$:$\mathrm{U_N}$ [SEP]}, where [CLS] and [SEP] are two special tokens and each $\mathrm{S_i}$:$\mathrm{U_i}$ pair is the name and utterance of a speaker separated by a colon. This sequence $\mathbb{X}$ is then fed into $L_{all} - L$ layers of Transformer blocks to gain its contextualized representation $\boldsymbol{E} \in \mathcal{R}^{J \times d}$ where J is the length of the sequence after tokenized by Byte-Pair Encoding (BPE) tokenizer (Sennrich et al., 2016) and d is the hidden dimension of the Transformer block. Here L_{all} is the total number of Transformer layers specified by the type of the PrLM, L is a hyper-parameter which means the number of decoupling layers.

3.3 Key-utterance Information Decoupling Block

Given the contextualized representation $\boldsymbol{E}$ from Section 3.2, follow Zhang et al. (2021), we gather the representation of [SEP] tokens from $\boldsymbol{E}$ as the representation of each utterance in the dialogue context. These representations are used to initialize N utterance nodes $\boldsymbol{E_U} = \{\boldsymbol{E_{u_i}} \in \mathcal{R}^d\}_{i=1}^N$ and a question node $\boldsymbol{E_q} \in \mathcal{R}^d$ as illustrated in the middle-upper part of Figure 2. The representations of normal tokens are gathered as token nodes $\boldsymbol{E_T} = \{\boldsymbol{E_{t_i}} \in \mathcal{R}^d\}_{i=1}^n$ where n is the number of normal tokens in the dialogue context. Then, another L layers of multi-head self-attention Transformer blocks are used to exchange information inter- and intra- the three types of nodes:

$$\begin{aligned} &\mathrm{Attn}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V \\ &\mathrm{head_i} = \mathrm{Attn}(EW_i^Q, EW_i^K, EW_i^V) \\ &\mathrm{MultiHead}(E) = [\mathrm{head}_1, \ldots, \mathrm{head}_h]W^O \end{aligned} \tag{1}$$

Here $W_i^Q \in \mathcal{R}^{d \times d_q}$, $W_i^K \in \mathcal{R}^{d \times d_k}$, $W_i^V \in \mathcal{R}^{d \times d_v}$, $W^O \in \mathcal{R}^{hd_v \times d}$ are matrices with trainable weights, h is the number of attention heads and $[;]$ denotes the concatenation operation.

After stacking L layers of multi-head self-attention: $\mathrm{MultiHead}([\boldsymbol{E_U}; \boldsymbol{E_q}; \boldsymbol{E_T}])$ to fully exchange information between these nodes, we get a question representation $\boldsymbol{H_q} \in \mathcal{R}^d$, the utterance representations $\boldsymbol{H_U} = \{\boldsymbol{H_{u_i}} \in \mathcal{R}^d\}_{i=1}^N$, and the token representations $\boldsymbol{H_T} = \{\boldsymbol{H_{t_i}} \in \mathcal{R}^d\}_{i=1}^n$.

$\boldsymbol{H_q}$ is then paired with each $\boldsymbol{H_{u_i}}$ to conduct the key-utterance prediction task. In detail, we use a heuristic matching mechanism proposed by (Mou et al., 2016) to calculate the matching score of the question representation and utterance representation. Here we define a matching function $Match(\boldsymbol{X}, \boldsymbol{Y}, \mathrm{activ})$, where $\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{R}^{d*N}$, as follows:

$$\begin{aligned} &\boldsymbol{G} = [\boldsymbol{X}; \boldsymbol{Y}; \boldsymbol{X} - \boldsymbol{Y}; \boldsymbol{X} \odot \boldsymbol{Y}] \in \mathcal{R}^{4d \times N} \\ &\boldsymbol{P} = \mathrm{activ}(\boldsymbol{a}^T \boldsymbol{G}) \in \mathcal{R}^N \end{aligned} \tag{2}$$

Here $\odot$ denotes element-wise multiplication and $\boldsymbol{a} \in \mathcal{R}^{4d}$ is a vector with trainable weights. The activ is an activation function to get a probability distribution according to the downstream loss function, which can be chosen from $softmax$ and $sigmoid$. In span-based dialogue MRC datasets, we set the pseudo-self-supervised key-utterance target based on the position of the answer span.

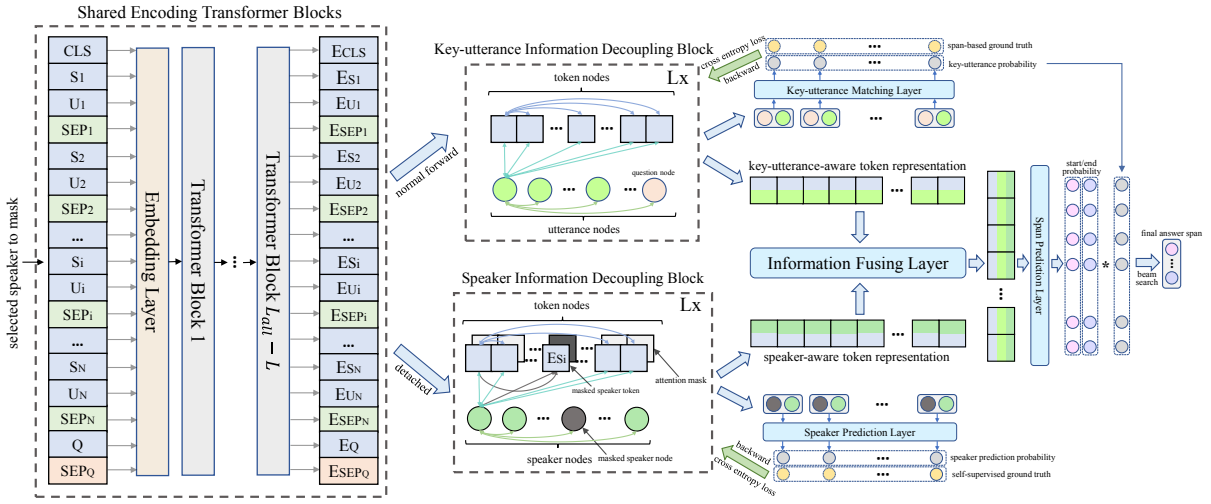


Figure 2: The overview of our model, which contains a shared Transformer encoder, a key-utterance information decoupling block, a speaker information decoupling block and a fusion-prediction layer. In speaker information decoupling block, the bi-directional arrow means that the information flows from and to both sides, the unidirectional arrow means that the information only flows from start nodes to end nodes.

We name it pseudo-self-supervised since it is generated from the original span labels, but requires no additional labeled data. Specifically, we set $p^{target} = i$ where i is the index of the utterance that contains the answer span. Then we calculate the key-utterance distribution by:

$$\begin{aligned} \boldsymbol{H_Q} &= \{\boldsymbol{H_q}\}_{i=1}^{N} \in \mathcal{R}^{d \times N} \\ \boldsymbol{P_U^{pred}} &= Match(\boldsymbol{H_U}, \boldsymbol{H_Q}, softmax) \end{aligned} \tag{3}$$

$\boldsymbol{P_U^{pred}} \in \mathcal{R}^N$ is later expanded to the length of token nodes to get $\boldsymbol{P_U^{expand}} \in \mathcal{R}^n$ which will be put forward to filter noisy utterances in the fusion-prediction layer (introduce in Section 3.5). We adopt cross-entropy loss to compute the loss of this task:

$$\mathcal{L}_U = -log(\boldsymbol{P_U^{pred}}[p^{target}]) \tag{4}$$

The gradient of $\mathcal{L}_U$ will flow backwards to refine the representations of the utterance nodes so that they can decouple key-utterance-aware information from the original representations. After the interaction between token nodes and utterance nodes, the token nodes will gather key-utterance-aware information from the utterance nodes. Therefore, we denote the token representations as key-utterance-aware: $\boldsymbol{H_T^k} = \boldsymbol{H_T} \in \mathcal{R}^{d \times n}$, which will be forwarded to the fusion-prediction layer described in Section 3.5.

3.4 Speaker Information Decoupling Block

This part is the core of our model, which contributes to modeling the complex speaker information flows. In this section, we first introduce the self-supervised speaker prediction task we proposed, then depict the decoupling process of speaker information.

3.4.1 Self-supervised Speaker Prediction

As defined in Section 3.1, we have a dialogue context $\mathbb{C} = \{U_1, U_2, ..., U_N\}$ where each utterance $U_i = \{S_i, W_i\}$ consists of a speaker S_i specified by a name. We randomly choose an $\mathrm{m_{th}}$ utterance and mask its speaker name. Then for every (U_i, U_m) pair where $i \neq m$, the model should determine whether they are uttered by the same speaker, that is to say, whether $S_i = S_m$.

We figure this task a relatively difficult one since it requires the model to have a thorough understanding of the speaker information flows and solve problems such as coreference resolution. Figure 3 is an example of the self-supervised speaker prediction task, where the speaker of the utterance in gray is masked. We human can determine that the masked speaker should be *Emily Waltham* by considering that *Ross* and *Monica* is persuading *Emily* to attend the wedding by showing her the wedding place, and when *Monica* and *Emily* reaches there, it should be *Emily* who is surprised to say *"Oh My God"*. However, it is not that easy for machines to capture these information flows.

3.4.2 Speaker Information Decoupling

To fully utilize the interactive feature of self-attention mechanism (Vaswani et al., 2017) and the powerful representational ability of PrLMs, we

Scene: *Ross and Emily's planned wedding place, Monica is dragging Emily in.*
Emily Waltham: *Monica, why have you brought me here of all places?!*
Monica Geller: *You'll see.*
Emily Waltham: *I tell you, this wedding is not going to happen.*
Scene: *At that Ross plugs in some Christmas lights to light the place up.*
[Masked]: *Oh My God!*
Ross Geller: *Okay? But - but imagine a lot more lights, okay? And - and flowers, and candles...*
Monica Geller: *And the musicians, look, they can go over here, okay? And the chairs can face this way, and... You go.*
Ross Geller: *If you don't love this, we'll do it in any other place at any other time. Really, it's fine, whatever you want.*
Emily Waltham: *It's perfect.*
.....

Figure 3: An example of the speaker prediction task, which involves three speakers. *Scene* here is a narrative description which introduces some additional information about the scene.

also use Transformer blocks to capture the interactive speaker information flows and fulfill this difficult task.

We first detach $\boldsymbol{E}$ from the computational graph to get $\boldsymbol{E^{de}}$, then as what we do in Section 3.3, the representation of [SEP] tokens are gathered from $\boldsymbol{E^{de}}$ to initialize $N-1$ unmasked speaker nodes $\boldsymbol{E_S} = \{\boldsymbol{E_{s_i}} \in \mathcal{R}^d\}_{i=1}^{N-1}$ and a masked speaker node $\boldsymbol{E_{s_m}} \in \mathcal{R}^d$. The representation of normal tokens are gathered as token nodes $\boldsymbol{E_T} = \{\boldsymbol{E_{t_i}} \in \mathcal{R}^d\}_{i=1}^{n}$. Then, we add attention mask to the token nodes corresponding to the selected speaker name before they are forwarded into the speaker information decoupling block, as illustrated in the middle-lower part of Figure 2. The reasons why we use this detach-mask strategy are as follows. First, we mask the selected speaker before the speaker information decoupling block instead of at the very beginning before the encoder since it is better to let the utterance decoupling block see all the speaker names. Based on this point, we detach $\boldsymbol{E}$ from the computational graph and add attention mask to avoid target leakage. If we use a normal forward instead, the encoder would simply attend to the speaker names, which would hurt performance (discuss in detail in Section 5.3). Besides, this strategy also helps the model better decouple the key-utterance-aware and speaker-aware information from the original representations.

In detail, the mask strategy is similar as Liu et al. (2021). We modify Eq. (1) to:

$$\begin{aligned}
&\mathrm{Attn}(Q,K,V,M) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}} + M)V \\
&\mathrm{head_i} = \mathrm{Attn}(EW_i^Q, EW_i^K, EW_i^V, M) \\
&\mathrm{MultiHead}(E,M) = [\mathrm{head}_1, \ldots, \mathrm{head}_h]W^O
\end{aligned} \tag{5}$$

Let the start index and end index of the masked speaker tokens be m_s and m_e, to make the selected speaker name unseen to other nodes, the attention mask is obtained as follows:

$$\boldsymbol{M_S}[i,j] = \begin{cases} -\infty, & \text{if } j \in [m_s, m_e] \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

By adding this mask, other nodes will not attend to the masked token nodes, thus preventing target leakage. On the mean time, the speaker nodes will have to collect clues from other nodes through deep interaction to make prediction, which implicitly models the complex speaker information flows.

After stacking L layers of masked multi-head self-attention: $\mathrm{MultiHead}([\boldsymbol{E_S}; \boldsymbol{E_{s_m}}; \boldsymbol{E_T}], \boldsymbol{M_S})$, we get a masked speaker representation $\boldsymbol{H_{s_m}} \in \mathcal{R}^d$, the normal speaker representation $\boldsymbol{H_S} = \{\boldsymbol{H_{s_i}} \in \mathcal{R}^d\}_{i=1}^{N-1}$, and the token representation $\boldsymbol{H_T} = \{\boldsymbol{H_{t_i}} \in \mathcal{R}^d\}_{i=1}^{n}$.

$\boldsymbol{H_{s_m}}$ is then paired with each $\boldsymbol{H_{s_i}}$ to conduct the self-supervised speaker prediction task. We also adopt the matching function defined in Eq. (2):

$$\begin{aligned}
&\boldsymbol{H_M} = \{\boldsymbol{H_{s_m}}\}_{i=1}^{N-1} \in \mathcal{R}^{d\times(N-1)} \\
&\boldsymbol{P_S^{pred}} = Match(\boldsymbol{H_S}, \boldsymbol{H_M}, sigmoid)
\end{aligned} \tag{7}$$

For convenience and without loss of generality, we make $m = N$ which means we mask the speaker of the $\mathrm{N_{th}}$ utterance, in the following description. We construct the self-supervised target by:

$$p_{s_i}^{target} = \begin{cases} 1, & \text{if } S_i == S_N \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Then binary cross entropy loss is applied here to compute the loss of this task:

$$\begin{aligned}
\mathcal{L}_S = -\frac{1}{N-1}\sum_{i=1}^{N-1}(&p_{s_i}^{target} * log(p_{s_i}^{pred}) \\
&+ (1-p_{s_i}^{target}) * log(1-p_{s_i}^{pred}))
\end{aligned} \tag{9}$$

The gradient of $\mathcal{L}_S$ will flow backwards to refine the representations of speaker nodes so that

they can decouple speaker-aware information from the original representations. After the interaction between token nodes and speaker nodes, the token nodes will gather speaker-aware information from the speaker nodes. Therefore, we denote the token representations as speaker-aware: $\boldsymbol{H}_{\boldsymbol{T}}^{s} = \boldsymbol{H}_{\boldsymbol{T}} \in \mathcal{R}^{d \times n}$, which will be forwarded to the fusion-prediction layer described in next section.

3.5 Fusion-Prediction Layer

Given the key-utterance-aware token representation $\boldsymbol{H}_{\boldsymbol{T}}^{k}$ and the speaker-aware token representations $\boldsymbol{H}_{\boldsymbol{T}}^{s}$, we first fuse these two kinds of decoupled representation using the following transformation:

$$\begin{aligned} \boldsymbol{H}_{\boldsymbol{T}}^{cat} &= [\boldsymbol{H}_{\boldsymbol{T}}^{k}; \boldsymbol{H}_{\boldsymbol{T}}^{s}; \boldsymbol{H}_{\boldsymbol{T}}^{k} - \boldsymbol{H}_{\boldsymbol{T}}^{s}; \boldsymbol{H}_{\boldsymbol{T}}^{k} \odot \boldsymbol{H}_{\boldsymbol{T}}^{s}] \\ \boldsymbol{H}_{\boldsymbol{T}}^{f} &= \text{Tanh}(W^{f} \boldsymbol{H}_{\boldsymbol{T}}^{cat}) \in \mathcal{R}^{d \times n} \end{aligned} \tag{10}$$

where $W^{f} \in \mathcal{R}^{d \times 4d}$ is a linear transformation matrix with trainable weights and Tanh is a non-linear activation function.

Then we compute the start and end distributions over the tokens by:

$$\begin{aligned} \boldsymbol{P}_{start} &= \text{softmax}(\boldsymbol{w}_{start}^{T} \boldsymbol{H}_{\boldsymbol{T}}^{f}) \odot \boldsymbol{P}_{\boldsymbol{U}}^{expand} \\ \boldsymbol{P}_{end} &= \text{softmax}(\boldsymbol{w}_{end}^{T} \boldsymbol{H}_{\boldsymbol{T}}^{f}) \odot \boldsymbol{P}_{\boldsymbol{U}}^{expand} \end{aligned} \tag{11}$$

where $\boldsymbol{w}_{start}$ and $\boldsymbol{w}_{end}$ are vectors of size $\mathcal{R}^{d}$ with trainable weights, $\boldsymbol{P}_{\boldsymbol{U}}^{expand}$ is defined on Section 3.3 and $\odot$ is element-wise multiplication.

Given the ground truth label of answer span $[a_s, a_e]$, cross entropy loss is adopted to train our model:

$$\mathcal{L}_{SE} = -(\log(\boldsymbol{P}_{start}[a_s]) + \log(\boldsymbol{P}_{end}[a_e])) \tag{12}$$

If the dataset contains unanswerable question, the representation of $\boldsymbol{H}_{\boldsymbol{T}}^{f}$ at $[CLS]$ position x is used to predict whether a question is answerable or not:

$$p_a = \text{sigmoid}(\boldsymbol{w}^{T} \boldsymbol{H}_{\boldsymbol{T}}^{f}[x] + \boldsymbol{b}) \tag{13}$$

where $\boldsymbol{w}^{T}$ and $\boldsymbol{b}$ are vectors of size $\mathcal{R}^{d}$ with trainable weights.

Given the ground truth of answerability $t_a \in \{0, 1\}$, binary cross entropy is applied to compute the answerable loss:

$$\begin{aligned} \mathcal{L}_{A} = &- ((1 - t_a) * log(1 - p_a) \\ &+ t_a * log(p_a)) \end{aligned} \tag{14}$$

The final loss is the summation of the above losses:

$$\mathcal{L} = \mathcal{L}_{U} + \mathcal{L}_{S} + \mathcal{L}_{SE}\ (+\mathcal{L}_{A}) \tag{15}$$

4 Experiments

4.1 Benchmark Datasets

We adopt FriendsQA (Yang and Choi, 2019) and Molweni (Li et al., 2020), two span-based extractive dialogue MRC datasets, as the benchmarks. Molweni is derived from the large-scale multi-party dialogue dataset — Ubuntu Chat Corpus (Lowe et al., 2015), whose main theme is technical discussions about problems on Ubuntu system. This dataset features in its informal speaking style and domain-specific technical terms. In total, it contains 10,000 dialogues whose average and maximum number of speakers is 3.51 and 9 respectively. Each dialogue is short in length with the average and maximum number of tokens 104.4 and 208 respectively. Unanswerable questions are asked in this dataset, hence the answerable loss in Eq. (14) is applied. Additionally, this dataset is equipped with discourse parsing annotations which is not used by our model however.

To evaluate our model more comprehensively, another open-domain dialogue MRC dataset FriendsQA is also used to conduct our experiments. FriendsQA excerpts 1,222 scenes and 10,610 open-domain questions from the first four seasons of a well-known American TV show *Friends* to tackle dialogue MRC on everyday conversations. Each dialogue is longer in length and involves more speakers, resulting in more complicated speaker information flows compared to Molweni. For each dialogue context, at least 4 out of 6 types (5W1H) of questions, are generated. This dataset features in its colloquial language style filled with sarcasms, metaphors, humors, etc.

4.2 Implementation Details

We implement our model based on *Transformers* Library (Wolf et al., 2020). The number of information decoupling layers L is chosen from 3 - 5 according to the type of the PrLM in our experiments. For Molweni, we set batch size to 8, learning rate to 1.2e-5 and maximum input sequence length of the Transformer blocks to 384. For FriendsQA, they are 4, 4e-6 and 512 respectively. Note that in FriendsQA, there are dialogue contexts whose length (in tokens) are larger than 512. We split those contexts to pieces and choose the answer with highest span probability $p_{start} * p_{end}$ as the final prediction[1].

[1]Codes and data are available at https://github.com/EricLee8/Multi-party-Dialogue-MRC

4.3 Baseline Models

For FriendsQA, we adopt BERT as the baseline model follow Li and Choi (2020) and Liu et al. (2020). For Molweni, we follow Li et al. (2021) who also employ BERT as the baseline model. In addition, we also adpot ELECTRA (Clark et al., 2020) as a strong baseline in both datasets to see if our model still holds on top of stronger PrLMs.

4.4 Results

Table 1 shows our experimental results on FriendsQA. $\text{BERT}_{\text{ULM+UOP}}$ (Li and Choi, 2020) is a method using pretrain-fine-tune form. They first pre-train BERT on FriendsQA and additional transcripts from Seasons 5-10 of *Friends* using well designed pre-training tasks Utterance-level-Masked-LM (ULM) and Utterance-Order-Prediction (UOP), then fine-tune it on dialogue MRC task. $\text{BERT}_{\text{graph}}$ (Liu et al., 2020) is a graph-based model that integrates relation knowledge and coreference knowledge using Relational Graph Convolution Networks (R-GCNs) (Schlichtkrull et al., 2018). Note that this model utilizes additional labeled data on coreference resolution (Chen et al., 2017) and character relation (Yu et al., 2020). We adopt the same evaluation metrics as Li et al. (2020): exactly match (EM) and F1 score. Our model reaches new state-of-the-art (SOTA) result on EM metric and comparable result on F1 metric, even without any additional labeled data. Besides, our model still gains great performance improvement under ELECTRA-based condition, which demonstrates the effectiveness of our model over strong PrLMs.

Model	EM	F1
$\text{BERT}_{\text{basline}}$	43.3	59.3
$\text{BERT}_{\text{ULM+UOP}}$ (Li and Choi)	46.8	63.1
$\text{BERT}_{\text{graph}}$ (Liu et al.)	46.4	**64.3**
BERT_{our}	**46.9**	63.9
$\text{ELECTRA}_{\text{basline}}$	52.8	70.1
$\text{ELECTRA}_{\text{our}}$	**55.8**	**72.3**

Table 1: Results on FriendsQA

Table 2 presents our experimental results on Molweni. Public Baseline is directly taken from the original paper of Molweni (Li et al., 2020). DADGraph (Li et al., 2021) is the current SOTA model that utilizes Graph Convolution Network (GCN) and the additional discourse annotations in Molweni to explicitly model the discourse structure. We see from the the table that our model outperforms strong baselines and the current SOTA model by a large margin, even under the condition that we do not make use of additional discourse annotations.

Model	EM	F1
$\text{BERT}_{\text{public basline}}$ (Li et al.)	45.3	58.0
$\text{BERT}_{\text{our basline}}$	45.8	60.2
$\text{BERT}_{\text{DADGraph}}$ (Li et al.)	46.5	61.5
BERT_{our}	**49.2**	**64.0**
$\text{ELECTRA}_{\text{basline}}$	56.8	70.6
$\text{ELECTRA}_{\text{our}}$	**58.0**	**72.9**

Table 2: Results on Molweni

5 Analysis

5.1 Performance Gain Analysis

To get more detailed insights on our proposed method, we analyze the results on different question types of FriendsQA over ELECTRA-based model. Also, we compare our model with the baseline model on these types to see where the performance gains come from. Table 3 shows the results of our model on different question types. Dist. means the distribution of each question type, from which we see that the question type of FriendsQA is nearly uniformly distributed.

Performance gains mainly come from question type *Who*, *When* and *What*. We argue that the speaker information decoupling block is the predominant contributor to *Who* question type since answering this type of question requires the model to have a deep understanding of speaker information flows and solve problems like coreference resolution, which is the same as our self-supervised speaker prediction task. For question type *When*, the key-utterance information decoupling block contributes the most. The answer of question type *When* usually comes from a scene description utterance, hence grabbing key-utterance information helps answer this kind of question. Among these improvements, question type *Who* benefits the most from our model, demonstrating the strong capability of the self-supervised speaker prediction task.

5.2 Ablation Study

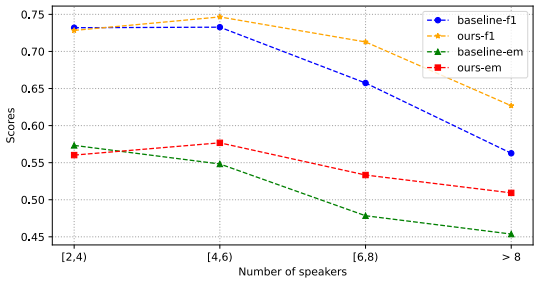
We conduct ablation study to see the contribution of each module. Table 4 shows the results of our ablation study. Here KIDB and SIDB are the abbreviation of Key-utterance Information Decoupling

Type	Dist.	EM	F1
Who	18.82	66.8(↑ **6.2**)	74.6(↑ **4.7**)
When	13.57	63.2(↑ **6.1**)	74.1(↑ **3.3**)
What	18.48	58.6(↑ **5.0**)	76.9(↑ 1.9)
Where	18.16	64.2(↑ 0.9)	79.3(↑ 1.4)
Why	15.65	36.2(↓ 0.5)	62.9(↑ 1.4)
How	15.32	41.3(↓ 0.9)	63.5(↑ 0.1)

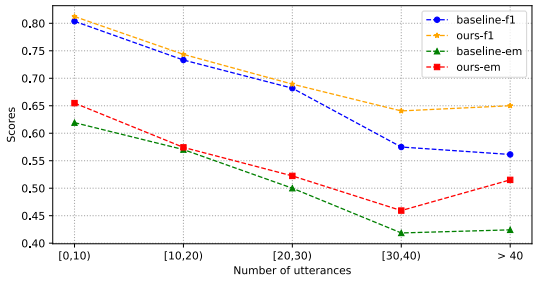
Table 3: Results on different question types, where up arrows↑ represent performance gain and down arrows↓ represent performance drop compared to the baseline model. Significant gains (greater than 3%) are marked as **bold**.

Model	FriendsQA		Molweni	
	EM	F1	EM	F1
Our Model	**55.8**	**72.3**	58.0	**72.9**
w/o KIDB	55.4	71.7	57.7	72.1
w/o SIDB	55.0	71.4	**58.2**	71.8
SpeakerEmb	55.5	71.9	57.5	72.4

Table 4: Results of Ablation Study



(a) Scores vs. Number of Speakers



(b) Scores vs. Number of Utterances

Figure 4: Influence of Speaker and Utterance Numbers

Model	EM	F1	Speaker
Our Model	**55.8**	**72.3**	80.8
w/o Detaching	54.5	70.8	96.8

Table 5: Influence of Detaching Operation

Block and Speaker Information Decoupling Block respectively. We see from the results that both of the two modules contributes to the performance gains of our final model. For FirendsQA, SIDB contributes more and otherwise for Molweni. This is because dialogue contexts in FriendsQA tend to be long, involve more speakers and carry more complex speaker information flows. On the contrary, dialogue contexts in Molweni are short with less turns and most of the questions can be answered by considering only one key-utterance.

To further investigate the effectiveness of our self-supervised speaker prediction task, we design a SpeakerEmb model in which we replace the speaker-aware token representation $\boldsymbol{H}_{\boldsymbol{T}}^{s}$ by speaker representations. The speaker representations are obtained by simply gathering embeddings from a trainable embedding look-up table according to the name of the speaker. Experimental results show that it only makes a slight performance gain compared to SIDB, demonstrating that simply adding speaker information is sub-optimal compared to implicitly modeling speaker information flows using our self-supervised speaker prediction task.

5.3 Influence of Detaching Operation

We conduct experiments to investigate the influence of detaching operation mentioned in Section 3.4. As shown in Table 5, if we do not detach $\boldsymbol{E}$ from the original computation graph when performing the speaker prediction task, the prediction accuracy reaches 96.8% in the test set of FriendsQA, indicating obvious label leakage. In the meantime, the EM and F1 scores drop to 54.5% and 70.8%, respectively. On the contrary, our model reaches a speaker prediction accuracy of 80.8%, which demonstrates that the detaching operation can effectively prevent label leakage.

5.4 Influence of Speaker and Utterance Numbers

Figure 4 illustrates the model performance with regard to the number of speakers and utterances on FriendsQA. At the beginning, the baseline model has similar performance to our model. However, with the number of speakers and utterances increasing, there is a growing performance gap between the baseline model and our model. This observation demonstrates that our SIDB and KIDB have strong abilities to deal with more complex dialogue contexts with a larger number of speakers and utterances.

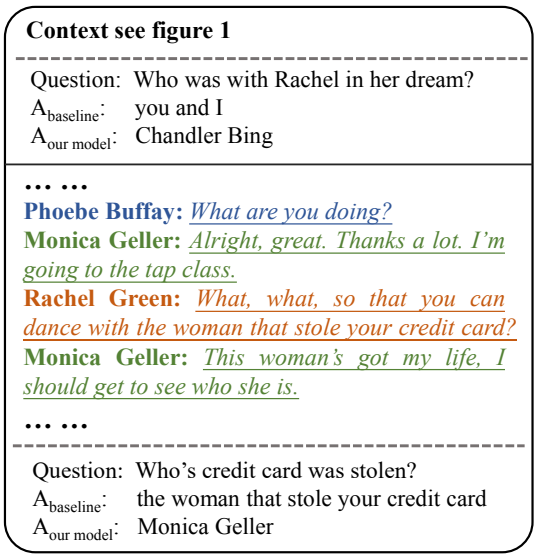


Figure 5: Two cases from FriendsQA

5.5 Case Study

To get more intuitive explanations of our model, we select two cases from FriendsQA in which the baseline model fails to answer (F1 = 0, or "exactly not match") but our model is able to answer (exactly match). Figure 5 illustrates two cases where the context of the first one is shown in Figure 1.

In the first case, the baseline model simply predicts that *"you and I"* were in *Rachel's dream* while fails to notice that *"you"* here refers to *Chandler*. On the contrary, our model is able to capture this information since it helps the speaker prediction task. In fact, if we mask *Rachel* in U_9, our model could tell the masked speaker is *Rachel*, indicating that it knows it should be *Rachel* who had a dream and U_9 is in response to U_8.

Similar observations can be seen in the second case. The baseline model simply matches the semantic meaning of the question and the context then makes a wrong prediction. Compared with the baseline model, our model has the ability to catch the information flow from *Rachel* to *Monica* thus predicts the answer correctly.

6 Conclusion

In this paper, for multi-party dialogue MRC, we propose two novel self- and pseudo-self-supervised prediction tasks on speaker and key-utterance to capture salient clues in a long and noisy dialogue. Experimental results on two multi-party dialogue MRC benchmarks, FriendsQA and Molweni, have justified the effectiveness of our model.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization.

Henry Y Chen, Ethan Zhou, and Jinho D Choi. 2017. Robust coreference resolution and entity linking on dialogues: Character identification on tv show transcripts. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 216–225.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Leyang Cui, Yu Wu, Shujie Liu, Yue Zhang, and Ming Zhou. 2020. Mutual: A dataset for multi-turn dialogue reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1406–1416.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Deepanway Ghosal, Navonil Majumder, Soujanya Poria, Niyati Chhaya, and Alexander Gelbukh. 2019. Dialoguegcn: A graph convolutional neural network for emotion recognition in conversation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 154–164.

Jia-Chen Gu, Tianda Li, Quan Liu, Zhen-Hua Ling, Zhiming Su, Si Wei, and Xiaodan Zhu. 2020. Speaker-aware bert for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2041–2044.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Karl Moritz Hermann, Tomáš Kočiskỳ, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Proceedings of the 28th International Conference on Neural Information Processing Systems-Volume 1*, pages 1693–1701.

Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. Gsn: A graph-structured network for multi-party dialogues. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5010–5016. International Joint Conferences on Artificial Intelligence Organization.

Qi Jia, Yizhu Liu, Siyu Ren, Kenny Zhu, and Haifeng Tang. 2020. Multi-turn response selection using dialogue dependency relations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1911–1920.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. Race: Large-scale reading comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Changmao Li and Jinho D Choi. 2020. Transformers to learn hierarchical contexts in multiparty dialogue for span-based question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5709–5714.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020. Molweni: A challenge multiparty dialogues-based machine reading comprehension dataset with discourse structure. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2642–2652.

Jiaqi Li, Ming Liu, Zihao Zheng, Heng Zhang, Bing Qin, Min-Yen Kan, and Ting Liu. 2021. Dadgraph: A discourse-aware dialogue graph neural network for multiparty dialogue machine reading comprehension. *arXiv preprint arXiv:2104.12377*.

Jian Liu, Dianbo Sui, Kang Liu, and Jun Zhao. 2020. Graph-based knowledge integration for question answering over dialogue. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2425–2435.

Longxiang Liu, Zhuosheng Zhang, , Hai Zhao, Xi Zhou, and Xiang Zhou. 2021. Filling the gap of utterance-aware and speaker-aware representation for multi-turn dialogue. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ryan Lowe, Nissan Pow, Iulian Vlad Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 285–294.

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 130–136.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics*, 7:249–266.

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.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. Dream: A challenge data set and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in Neural Information Processing Systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Weishi Wang, Steven CH Hoi, and Shafiq Joty. 2020. Response selection for multi-party conversations with dynamic topic tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6581–6591.

Thomas Wolf, Julien Chaumond, Lysandre Debut, Victor Sanh, Clement Delangue, Anthony Moi, Pierric Cistac, Morgan Funtowicz, Joe Davison, Sam Shleifer, et al. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505.

Zhengzhe Yang and Jinho D Choi. 2019. Friendsqa: Open-domain question answering on tv show transcripts. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 188–197.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in Neural Information Processing Systems*, 32:5753–5763.

Dian Yu, Kai Sun, Claire Cardie, and Dong Yu. 2020. Dialogue-based relation extraction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4927–4940.

Zhuosheng Zhang, Jiangtong Li, Pengfei Zhu, Hai Zhao, and Gongshen Liu. 2018. Modeling multi-turn conversation with deep utterance aggregation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3740–3752.

Zhuosheng Zhang, Junlong Li, and Hai Zhao. 2021. Multi-turn dialogue reading comprehension with pivot turns and knowledge. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:1161–1173.