Enhancing Extractive Question Answering in Multiparty Dialogues with Logical Inference Memory Network

Shu Zhou[1,2]*, Rui Zhao[3], Zhengda Zhou[1,2]*,
Haohan Yi[1,2]*, Xuhui Zheng[1,2], Hao Wang[1,2]†
[1]Nanjing University, China
[2]Key Laboratory of Data Engineering and Knowledge Services
in Jiangsu Provincial Universities (Nanjing University), China
[3]University of Technology Sydney, Australia
shuzhou@smail.nju.edu.cn; rui.zhao-2@student.uts.edu.au; zzd20010619@163.com;
yi_haohan@163.com; 287961528@qq.com; ywhaowang@nju.edu.cn

Abstract

Multiparty dialogue question answering (QA) within machine reading comprehension (MRC) presents significant challenges due to the complex interplay of information across multiple speakers and the need for advanced logical reasoning. While existing models often focus on separating dialogue information based on speakers and utterances, they rarely address the crucial aspect of logical inference, leading to suboptimal performance in understanding and answering questions. To bridge this gap, we introduce the Logical Inference Memory Network (LIMN), a novel architecture designed for extractive QA in multiparty dialogues. LIMN incorporates a unique inference module pretrained on plain text QA datasets (like SQuAD 2.0), enabling it to transfer robust logical reasoning abilities to the dialogue domain. This module generates representations that are specifically attuned to logical inference, which are then integrated into the dialogue context. Furthermore, we propose a key-utterance-based interaction mechanism that dynamically focuses on the most relevant utterances within the dialogue, enhancing the model's ability to pinpoint answers. To ensure robust performance, LIMN employs a multitask learning strategy that jointly optimizes for answer extraction, answerability prediction, key-utterance identification, and masked speaker prediction. Extensive experiments on the Molweni and FriendsQA benchmarks, encompassing 25,000 and 10,000 questions respectively, demonstrate that LIMN achieves state-of-the-art results, affirming the effectiveness of incorporating logical inference in multiparty dialogue QA.

1 Introduction

Multiparty dialogue-based machine reading comprehension (MRC) is a crucial area of research in natural language processing (NLP), aiming to enable machines to understand and reason about conversations involving multiple participants. The ability to effectively process such dialogues has significant implications for various downstream applications, including extractive question answering (QA) (Li and Zhao, 2021b; Liu et al., 2021b; Ma et al., 2023a). Multiparty dialogue QA, in particular, focuses on identifying and extracting the most relevant answers from a sequence of utterances contributed by different speakers (Li et al., 2020a; Liu et al., 2021b).

Compared to simpler forms of MRC, such as plain article comprehension (e.g., reading comprehension on Wikipedia articles, as addressed by the SQuAD dataset (Rajpurkar et al., 2018a)) and two-party dialogue QA (e.g., conversations between two individuals), multiparty dialogue QA presents unique challenges. These challenges stem from two primary factors: 1) The complex information flow inherent in conversations with multiple speakers, necessitating models to capture intricate discourse relations and speaker-specific information (Li et al., 2021). 2) The need for sophisticated logical reasoning to identify answers that might not be explicitly stated but rather inferred from the dialogue context.

To tackle these challenges, various deep learning approaches have been proposed for multiparty dialogue QA. Many of these methods focus on structuring dialogues as relational graphs to model the interactions between utterances (Liu et al., 2020). For example, (Li et al., 2021) and (Ma et al., 2023a) introduce discourse dependency links to capture information flow by constructing heterogeneous graph networks. Other studies explore the use of multi-head attention (MHA) mechanisms to model fine-grained relations between speakers and utterances or across different time points in the dialogue (Liu et al., 2021a; Li et al., 2022a). Additionally, research has shown that incorporating auxiliary sub-tasks, such as speaker and key-utterance pre-

* These authors contributed equally to this work.
†Corresponding author.

Proceedings of the 31st International Conference on Computational Linguistics, pages 8725–8738
January 19–24, 2025.

diction, can enhance performance by enabling the model to learn more nuanced representations (Li and Zhao, 2021a). External knowledge, including discourse, syntax, commonsense, and knowledge graphs, has also been leveraged to improve dialogue understanding (Shuster et al., 2021; Zhang et al., 2021).

Despite these advancements, existing methods, which predominantly rely on fine-tuning transformer-based pre-trained language models (PLMs) (Vaswani et al., 2017), often fall short in capturing the complex logical inference relations crucial for accurate question answering in multiparty dialogues. While PLMs excel at transferring general language knowledge from vast text corpora, they struggle to adapt to the specific demands of dialogue-based QA, where the distribution of data differs significantly from that of plain article QA. This discrepancy often results in weaker logical inference capabilities in dialogue QA compared to plain article QA. Although the conversational nature of dialogues introduces complexities not found in plain articles, the underlying narrative logic still follows a sequential progression similar to that found in natural language. Therefore, a critical challenge in multiparty dialogue QA is to enhance the model's ability to perform sequential and QA-based logical inference.

To address this critical need, we propose a novel Logical Inference Memory Network (LIMN) specifically designed for extractive QA in multiparty dialogues. Unlike previous approaches, LIMN incorporates an inference memory encoder that is pretrained on a plain article QA dataset (SQuAD 2.0) and then frozen. This allows LIMN to leverage the strong logical reasoning capabilities developed during pretraining and transfer them to the dialogue domain. The inference memory encoder generates representations that capture the sequential and logical relationships necessary for accurate QA, effectively acting as an external memory of logical inference knowledge. Furthermore, we introduce a key-utterance-based interaction mechanism that dynamically focuses on crucial utterances, facilitating deeper information exchange within the dialogue.

The main contributions of this paper are summarized as follows:

(1) We introduce LIMN, a novel architecture for multiparty dialogue QA that explicitly models logical inference relations by incorporating an inference memory encoder pretrained on plain article QA corpora. This encoder captures and retains sequential and logical inference patterns, providing a significant advantage over previous models.

(2) We propose a key-utterance-based interaction mechanism that enhances information flow by strategically focusing on and integrating information from the most relevant utterances identified within the dialogue.

(3) We demonstrate through extensive experiments on the Molweni and FriendsQA datasets that LIMN outperforms existing state-of-the-art models, highlighting the effectiveness of our approach in improving logical inference for multiparty dialogue QA.

2 Methodology

Figure 1 provides an overview of the proposed LIMN model, which is specifically designed to enhance extractive question answering in multiparty dialogues. LIMN consists of three primary components: a PLM-based encoder, a dialogue decouple network, and a multitask learning strategy. 1) Initially, the model takes questions and dialogue utterances, along with their associated speaker characteristics, and combines them into a sequential format. These are then encoded using a PLM-based encoder, such as BERT or ELECTRA. 2) To capture fine-grained information crucial for QA, a dialogue decouple network is employed. This network consists of an inference memory encoder and a speaker and utterance decouple encoder. The inference memory encoder is first pretrained on plain article QA datasets from SQuAD 2.0 (Rajpurkar et al., 2018b) to learn general patterns of logical inference. After pretraining, it is frozen to preserve this acquired knowledge and generate logical inference-aware representations for downstream multiparty dialogue QA tasks. The speaker and utterance decouple encoder, on the other hand, focuses on disentangling speaker-specific information and identifying key utterances within the dialogue. 3) Finally, a multitask learning strategy is implemented to optimize the model for various aspects of multiparty dialogue MRC and QA, including the primary task of span-level answer extraction and auxiliary sub-tasks like answerability classification, key-utterance prediction, and masked speaker prediction. The algorithm can be found in Appendix E.

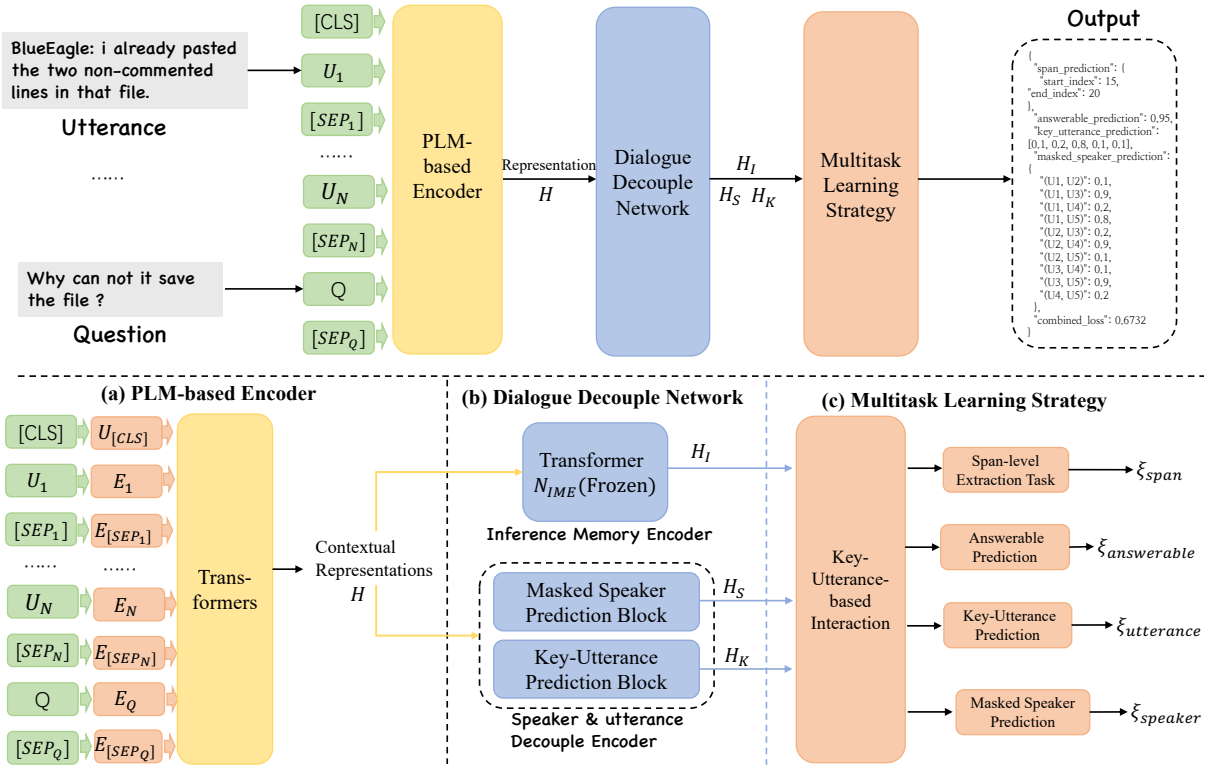

Figure 1: Overview of the LIMN model for multiparty dialogue QA. The model comprises (a) PLM-based encoder, (b) dialogue decouple network, and (c) multitask learning strategy. The PLM encodes dialogue utterances along with speaker information. The dialogue decouple network then processes these encodings through its inference memory and speaker-utterance decouple encoders. Finally, multitask strategies are applied to facilitate effective QA inference.

2.1 Task formulation

The multiparty dialogue QA task can be formally defined as follows: give a dialogue D with N utterances and M related questions Q, the objective is to identify the precise answer span A for each question. If a question is unanswerable given the dialogue context, the answer is designated as empty. Each utterance U_n within the dialogue is composed of the speaker's name S_n and a sequence of D_n words. Similarly, each question-answer pair (Q_m, A_m) consists of a sequence of words.

2.2 Pretrained Language Model-based Encoder

Leveraging the success of fine-tuning PLMs for various NLP tasks, we adopt a PLM (such as BERT or ELECTRA) as the core context encoder for modeling both the dialogue and the questions. Given a dialogue context D and a corresponding question Q_m, the input sequence x with L tokens is constructed as follows:

$$x = \{[CLS]U_1[SEP_1]\cdots U_N[SEP_N]Q_i[SEP_Q]\} \quad (1)$$

where $[CLS]$ and $[SEP_N]$, $n \in \{1:N;Q\}$ are special tokens for classification and separating utterances and questions. To convert discrete tokens (or words) into dense vectors for deep learning model optimization, embedding technology is utilized for generating word embeddings, denoted as $E \in R$ with dimensionality of d. Furthermore, to capture contextual dialogue representations H, PLM encodes E with stacked N_{PLM} transformer layers associating well pretrained checkpoints, formulated as: $H = PLM(E)$. Note that the input and output of the transformer layer are equipped with the same dimensionality of d for consistency.

2.3 Information Decouple

2.3.1 Inference memory encoder

To enhance the model's ability to understand and analyze dialogues, particularly for logical inference, we introduce an inference memory encoder. This encoder is designed to transform the contextual representations H from the PLM into inference-aware representations H_I. The key idea

is to pretrain this encoder on a large plain text QA dataset (SQuAD 2.0) and then freeze its weights. This allows the encoder to learn general patterns of logical inference from a simpler domain and retain this knowledge when applied to more complex multiparty dialogues. The structure of the inference memory encoder mirrors that of a transformer layer with N_{IME} layers.

During the pretraining phase (continuous learning), plain text QA articles and questions are first processed by the PLM encoder to obtain H. The inference memory encoder then transforms H into inference-aware representations: $H_I = IME(H)$. The encoder is trained to ensure that it captures inference-relevant information for both articles and dialogues. Crucially, during the fine-tuning phase on multiparty dialogue QA tasks, the weights of the inference memory encoder are frozen. This ensures that the QA inference capabilities learned from SQuAD are preserved and effectively transferred to the dialogue domain.

2.3.2 Speaker and Utterance Decouple Encoder

Multiparty dialogues often involve complex interactions between speakers with varying characteristics and contributions. To address this, we introduce a speaker and utterance decouple encoder. This component consists of a masked speaker prediction block and a key-utterance prediction block, both designed to enhance the model's understanding of speaker-specific nuances and the identification of crucial utterances.

The masked speaker prediction block aims to refine the utterance representations H_{U_n} within the contextual representations H by incorporating speaker-specific information. During training, a candidate utterance U_{mask} is selected, and its speaker S_{mask} is treated as unknown (masked). A self-supervised task is then performed to determine whether U_{mask} originates from the same speaker as other utterances in the context. Speaker-aware representations H_S are generated using a multi-headed attention (MHA) module with N_{sMHA} layers:

$$H_S = sMHA(H_{detached}, H_{detached}, H_{detached}) \quad (2)$$

where $H_{detached}$ represents a detached version of H (gradients are not propagated back through this path), preventing the speaker prediction task from directly influencing the initial contextual representations.

The key-utterance prediction block focuses on transforming the dialogue context representations H into key-utterance-aware representations H_K. Importantly, unlike other specialized representations that are often generated from separate modules, H_K is derived directly from the original H through backpropagation, guided by the key-utterance prediction task. This approach facilitates the creation of representations that are simultaneously aware of inference and speaker information.

2.4 Multitask Learning Strategy

Recognizing the complexity of information flow in multiparty dialogues, we employ a multitask learning strategy to leverage robust representations and improve overall performance. This strategy involves a primary QA task (span-level answer extraction) and three auxiliary sub-tasks: answerable prediction, key-utterance prediction, and masked speaker prediction. Each sub-task contributes to refining specific aspects of the dialogue representations.

2.4.1 Span-level Answer Extraction Task

The core of our model is the span-level answer extraction task, which is essential for identifying the correct answer to a given question. This task is performed during both the pretraining (continuous learning) and fine-tuning phases. In the pretraining phase, the model learns to identify answer spans within the inference-aware representations H_I. Two trainable vectors, v_{start} and v_{end}, are used to compute the probabilities of each token being the start or end of the answer span:

$$\begin{aligned} P_{start} &= softmax(v_{start}^T H_I) \\ P_{end} &= softmax(v_{end}^T H_I) \end{aligned} \quad (3)$$

where P_{start} and P_{end} denote probabilities of tokens being the start and end of the answer, respectively. The loss for this task is calculated using cross-entropy:

$$\xi_{span}^{spuad} = -(logP_{start}[y_{start}] + logP_{end}[y_{end}]) \quad (4)$$

where y_{start} and y_{end} are the ground truth start and end positions. Optimizing ξ_{span}^{squad} during pretraining ensures that the model learns to capture logical inference relationships, providing valuable checkpoints for downstream dialogue QA tasks.

During fine-tuning on multiparty dialogues, the model is initialized with the weights obtained from

the pretraining phase on SQuAD. To further enhance the model's ability to handle complex interactions, we introduce a key-utterance-based interaction mechanism. This mechanism models the interplay between the inference-aware (H_I), key-utterance-aware (H_K), and speaker-aware (H_S) representations to generate more robust dialogue representations $H_{enhanced}$. The interaction process involves a heuristic matching strategy:

$$\begin{aligned} H_D &= Fusion(H_K, H_S) \\ H_d &= [H_K; H_S; H_K - H_S; H_K \cdot H_S] \end{aligned} \tag{5}$$

Here, $Fusion$ represents a concatenation operation followed by a linear transformation. H_d combines H_K and H_S through concatenation, element-wise difference, and element-wise product to capture different aspects of their interaction.

To integrate the inference-aware representations H_I into the dialogue representations H_D, another fusion operation is performed: $H_F = Fusion(H_D, H_I)$. H_F is then decomposed into representations corresponding to the dialogue context (H_F^C), the question (H_F^Q), and answer-candidate tokens (H_F^A). To model the interactions between these components, we employ a Dynamic Multi-head Attention (DUMA) module:

$$\begin{aligned} H_A &= DUMA(H_F^C, H_F^A) \\ &= Fusion(mean(H_F^{'C}), mean(H_F^{'A})) \\ H_F^{'C} &= MHA(H_F^C, H_F^A, H_F^A) \\ H_F^{'A} &= MHA(H_F^A, H_F^C, H_F^C) \end{aligned} \tag{6}$$

The DUMA module uses multi-head attention to model the interactions between the context and answer-candidate representations and vice-versa. Finally, the enhanced dialogue representation $H_{enhanced}$ is obtained by: $H_{enhanced} = Fusion(H_F, repeat(H_A))$.

2.4.2 Answerable Prediction

Determining whether a question is answerable based on the provided dialogue is crucial for robust QA. We incorporate an answerable prediction module that utilizes the key-utterance-aware representations H_K. Specifically, H_K is divided into utterance (H_K^C) and question (H_K^Q) components, with total lengths $L_Q + L_C = L$. An interaction between these components is modeled using the DUMA module to derive answer-aware representations H_K^A:

$$H_K^A = DUMA(H_K^C, H_K^Q) \tag{7}$$

Answerability is then predicted using a multilayer perceptron (MLP) classifier with a sigmoid activation function, which assesses the relevance of the question to the dialogue context. The binary cross-entropy loss is used to measure the accuracy of these predictions, denoted as $\xi_{answerable}$.

2.4.3 Key-utterance Prediction

Identifying the key utterance(s) containing the exact answer is essential in extractive QA. The model updates the dialogue context representations H to key-utterance-aware representations H_K using a pseudo-self-supervised approach. Each utterance representation $H_{[SEP_n]}$ is concatenated with the answer-aware representation H_K^A. A binary classifier then determines whether the corresponding utterance U_n is a key utterance. The binary cross-entropy loss is used to calculate the key-utterance prediction loss $\xi_{utterance}$ for all N utterances.

2.4.4 Masked Speaker Prediction

This sub-task focuses on enhancing the speaker-aware representations. It involves pairing a masked utterance U_{mask} with other unmasked utterances U_{unmask}. The model then determines whether both utterances are from the same speaker. This is achieved using a binary classifier that operates on the concatenated representations $H_S[SEP_{unmask}]$ and $H_S[SEP_{mask}]$. The cross-entropy loss $\xi_{speaker}$ is computed for all $N-1$ utterance pairs.

2.4.5 Multitask Objective

To optimize the model for all aspects of multiparty dialogue QA, we combine the objectives of the individual tasks into a single multitask objective:

$$\xi = \xi_{span} + \xi_{answerable} + \xi_{utterance} + \xi_{speaker} \tag{8}$$

This combined objective allows the model to learn jointly from all tasks, leading to improved performance and more robust representations.

3 Experiments

3.1 Datasets

SQuAD 2.0 (Rajpurkar et al., 2018b), based on Wikipedia and combining answerable and unanswerable questions, is used for learning . It features 130k samples from 442 articles, including 43k negative samples, and a development set with 11k samples (Rajpurkar et al., 2018a). Comparative studies involve the Molweni and FriendsQA datasets (Yang and Choi, 2019). The Molweni

dataset is based on Ubuntu forum conversations and contains discourse relations in multi-party discussions (Li et al., 2020b). FriendsQA, based on the TV show Friends, has 1,222 dialogues with 10,610 questions (Yang and Choi, 2019).

3.2 Baselines and Metrics

We use two PLMs as baselines, including $\text{BERT}_{\text{large}}$ (Devlin et al., 2019) and $\text{ELECTRA}_{\text{large}}$ (Clark et al., 2020). We compare LIMN with SUP (Li and Zhao, 2021b), SUP+MI (Zhu et al., 2022), SUP+BiDeN (Li et al., 2022a), MaskAttn+Graph (Ma et al., 2023b) and SOTA model (CADA (Li et al., 2023)). The baseline introductions are in Appendix A.

Consistent with SQuAD 2.0 (Rajpurkar et al., 2018b), both the exact match (EM) and macro-averaged F1-score were applied as the evaluation metrics.

3.3 Hyperparameter Fine-tuning and Implementation Details

In the experiments, key hyperparameters impacting results include the learning rate, N_{IME} (number of MHA layers in the speaker prediction module), and N_{IME} (number of transformer block layers in the inference memory encoder). Initial tests determined the optimal learning rate and N_{sMHA} through grid search, then fixed them to evaluate N_{IME}'s influence, setting N_{IME} at 3 for consistency.

We utilized BERT and ELECTRA large versions as backbones, setting initial learning rates to 0.0001 with a warm-up ratio of 0.1. For Molweni, batch sizes and learning rates were adjusted based on the backbone: BERT used a batch size of 8 and a learning rate of 4e-5, while ELECTRA used 12 and 2e-5. In FriendsQA, batch sizes were set to 4, with learning rates of 6e-6 for ELECTRA and 4e-6 for BERT. The maximum number of utterances was capped at 10 for Molweni and 20 for FriendsQA. The training cost discussion could be found in Appendix C.

3.4 Comparative Results

To evaluate the effectiveness of the proposed Logical Inference Memory Network (LIMN), we conducted comparative experiments using two robust Pretrained Language Models (PLMs): BERT-large and ELECTRA-large. These experiments were performed on the Molweni and FriendsQA datasets, with the results during the fine-tuning phase presented in Table 1.

Model	Molweni		FriendsQA	
	EM	F1	EM	F1
$\text{BERT}_{\text{large}}$ (baseline)	50.5	65.1	46.0	63.1
SUP (2021)	49.2	64.0	46.9	63.9
SUP+MI (2022)	51.1	64.7	-	-
MaskAttn+Graph (2023)	49.7	64.4	47.0	63.0
CADA (2023, SOTA)	52.9	67.6	47.4	65.6
LIMN† (ours)	**53.1**	**68.1**	**50.1**	**67.2**
$\text{ELECTRA}_{\text{large}}$ (baseline)	56.8	70.6	52.8	70.1
SUP (2021)	58.0	72.9	55.8	72.3
SUP+MI (2022)	58.7	73.1	57.1	73.0
CADA (2023, SOTA)	59.8	73.6	59.2	76.7
LIMN† (ours)	**60.2**	**74.5**	**60.5**	**77.6**

Table 1: Comparative results on both Molweni and FriendsQA. Numbers marked with † denoted that the improvements were statistically significant (t-test with p-value < 0.05) comparing with the corresponding models. Numbers in bold denoted that the results achieved the best performance.

Our findings reveal that question-answering (QA) tasks in multiparty dialogues are significantly more challenging than those in plain text scenarios. Traditional methods that model dialogues at the speaker and utterance levels using Speaker and Utterance Prediction (SUP) demonstrate some success. More advanced techniques, such as speaker mask attention and heterogeneous graph networks based on discourse and speaker relations, further improve performance. However, these methods often rely on additional data annotation, increasing the complexity of implementation.

In contrast, the proposed LIMN model achieved state-of-the-art results on both benchmark datasets, irrespective of the underlying PLM (BERT-large or the more advanced ELECTRA-large). This consistent performance underscores the generalizability and robustness of our approach. The superior performance of LIMN can be attributed to three key factors: 1) Effective decoupling of speaker and key-utterance information within the dialogue through SUP sub-tasks. 2) Introduction of plain article QA datasets as external knowledge, which enables the model to learn and retain QA-related knowledge, thereby bolstering its logical inference capabilities for dialogue QA tasks. 3) A key-utterance-based information interaction strategy that effectively integrates the representations decoupled by multiple auxiliary sub-tasks, focusing on deep information interactions among dialogue contexts, questions, and key utterances.

3.5 Ablation Study

To verify the influence of introducing external QA knowledge, we remove both the inference memory encoder and external knowledge injection. To verify the influence of the inference memory encoder, we removed it but still used the external plain text QA dataset for continuous learning. In addition, we also verified the impact of the key-utterance-based interaction strategy and the freeze operation of the inference memory encoder.

As illustrated in Table 2, the experimental results show that each part has a significant impact on the performance of the proposed model. The introduction of plain text QA knowledge to enhance the inference ability of the model is effective. In addition, the use of an inference memory encoder and freeze operation allows the model to learn and memorize this inference ability better. The interaction strategy based on key-utterance also improves the performance of the model.

Model	Molweni		FriendsQA	
	EM	F1	EM	F1
LIMN(ELECTRA)	**60.2**	**74.5**	**60.5**	**77.6**
w/o External Knowledge Injection	58.4	73.0	56.8	72.9
w/o Inference Memory Encoder	59.3	73.4	60.1	75.2
w/o Key-utterance-based Interaction	59.2	73.7	59.5	74.6
w/o Freeze Operation	58.5	74.0	59.7	75.4

Table 2: Ablation experiment results.

3.6 Influence of Speaker and Utterance Numbers

	LIMN		CACD		baseline	
	EM	F1	EM	F1	EM	F1
Number of speakers						
$2 \sim 3$	65.1	78.3	57.1	73.2	58.3	72.9
$4 \sim 5$	58.1	76.8	57.2	74.3	55.1	73.9
$6 \sim 7$	57.6	73.2	54.4	72.2	48.2	67.8
≥ 8	58.6	70.1	53.2	63.2	45.2	63.2
Number of utterances						
$1 \sim 10$	68.3	83.1	67.1	82.3	62.3	80.1
$11 \sim 20$	67.8	79.8	58.3	74.5	58.1	74.1
$21 \sim 30$	53.1	68.7	52.8	66.1	50.2	64.6
$31 \sim 40$	48.7	64.7	48.2	63.2	43.2	57.6
≥ 41	59.7	69.2	52.1	61.2	44.1	56.2

Table 3: Performance of LIMN (ELECTRA) on FriendsQA under different numbers of utterances and speakers.

We further investigated the influence of dialogue complexity, as measured by the number of speakers and utterances, on model performance using the Molweni and FriendsQA datasets. The performance variations of the LIMN model across different dialogue sizes are detailed in Table 3.

We specifically selected the FriendsQA dataset for these experiments due to its greater complexity compared to Molweni, characterized by a higher number of interlocutors and longer discourse. The FriendsQA dataset also exhibits a wider range of dialogue sizes. Table 3 reveals that the performance of all models, including LIMN, was affected to varying degrees by the increasing number of speakers and utterances. However, LIMN and the CACD model, both of which decouple dialogues at the speaker and utterance levels, were less susceptible to these variations than the baseline model. Notably, LIMN, which leverages external knowledge for learning inference relations, exhibited superior performance across most dialogue size ranges. This advantage was particularly pronounced in more complex dialogues, attributable to its enhanced inference capabilities and the key-utterance-based information interaction mechanism.

3.7 Case Study: Model Performance Analysis

Example 1 (from Molweni Datasets) ... **BlueEagle**: i already pasted the two non-commented lines in that file. **Techsupport**: permission denied lol cant save the file. **BlueEagle**: that's because it's owned by root **Techsupport**: with the user that i'm looged in as **BlueEagle**: but you already know that , don't you ? ... **Question**: Why can not it save the file ? **Answer**: because it's owned by root ? ELECTRA(baseline): permission denied SUP: permission denied lol LIMN(ELECTRA): because it's owned by root	*Example 2 (from Molweni Datasets)* ... **slerder**: thanks. is dpkg for installing things. **ziroday**: yep , apt is a frontend to dpkg **ziroday**: can you please do sudo apt-get install install pastebinitand then do cat FILEPATH pastebinit **ziroday**: you installed the nvidia driver. **BlueEagle**: but you already know that , don't you ? ... **Question**: What is the first step of ziroday's advice ? **Answer**: do sudo apt-get install install pastebinit ? ELECTRA(baseline): installed the nvidia driver SUP: do sudo apt-get install install pastebinit and then do cat FILEPATH pastebinit LIMN(ELECTRA): do sudo apt-get install install pastebinit
Example 3 (from FriendsQA Datasets) ... **Monica Geller**: And well , we probably should n't see each other anymore . I 'm sorry . **Peter Becker**: Okay , yeah . I mean ... If that 's , if that 's really what you want , okay . **Monica Geller**: Okay , bye . **#Note#**: (She kisses him on the cheek, and he kisses her back on the mouth .) **Peter Becker**: I 'm sorry things did n't work out ... **Question**: How does Monica kiss Peter ? **Answer**: on the cheek ? ELECTRA(baseline): on the cheek , and he kisses her back on the mouth SUP: on the mouth LIMN(ELECTRA): on the cheek	*Example 4 (from FriendsQA Datasets)* ... **Monica Geller**: You were the next caller five hours ago . You must be going crazy . **Peter Becker**: Nah . I kept myself busy . **Monica Geller**: Okay , bye . **#Note#**: (Both Rachel and Monica walk into their bedrooms , stop , and come back into the living room with confused looks on their faces .) **Peter Becker**: Oh , okay , yeah . I put your stuff in her room, and her stuff in your room . ... **Question**: Where did Phoebe not put Rachel and Monica 's things ? **Answer**: their bedrooms ELECTRA(baseline): i put your stuff in her room , and her stuff in your room SUP: in her room , and her stuff in your room LIMN(ELECTRA): their bedrooms

Table 4: Examples from Molweni and FriendsQA, where the red font indicates the answer and the blue font indicates the wrong prediction.

We analyzed four examples from Molweni and FriendsQA datasets to demonstrate the LIMN model's effectiveness from Table 4.

Example 1 (Why-type question from Molweni): The baseline and SUP models identified a superficial reason for a file save failure. In contrast,

LIMN accurately pinpointed the root ownership as the underlying cause, showcasing its superior logical inference.

Example 2 (What-type question from Molweni): The baseline model's prediction deviated significantly. While the SUP model correctly found the key utterance, it failed to precisely address the question's specifics. LIMN, however, provided an exact answer match.

Example 3 (How-type question from FriendsQA): All models identified the key utterance, but the baseline model struggled with the referential relationship between Monica and Peter. The SUP model misunderstood this relationship, whereas LIMN accurately matched "Monica" with "she" and "Peter" with "he", leading to the correct answer.

Example 4 (Where-type question from FriendsQA): Both baseline and SUP models misinterpreted the question's negation and produced opposite results. LIMN correctly comprehended the question and provided the exact answer.

3.8 Comparison of Performance for Different Types of Problems

In order to explore the performance of the model on different types of questions, we can divide the questions into the following categories for experimentation: Simple Fact-based Questions, Inference-based Questions, Cross-speaker Reference Inference-based Questions, Cross-speaker Reference Questions, and Negative Questions.

The purpose of the experiment is to compare the performance of the full model with that of the model with IME removed on different types of questions to verify the advantages of the Logical Reasoning perceptual representation on complex reasoning problems.

Question Type	LIMN (full Model)		w/o IME	
	EM	F1	EM	F1
Fact-based Questions	**68.4**	**80.1**	65.2	77.3
Inference-based Questions	**58.3**	**72.9**	52.8	68.2
Cross-speaker Questions	**60.5**	**75.4**	56.1	70.3
Negative Questions	**55.2**	**69.7**	50.4	66.1

Table 5: Performance Comparison of Different Types of Problems: Complete vs. Removed IME Models on Different Problem Types.

In Simple Factual Problems, the full model showed a small improvement on both EM and F1, indicating that IME is still helpful for simple problems, but the advantage is not significant. In Reasoning Problems, the complete model shows significant improvement in the performance on reasoning problems, especially the F1 score is improved by 4.7 percentage points, indicating that IME has a significant enhancement effect on reasoning ability. In Multi-party referencing problems, the full model performs better on complex problems involving multi-party conversations, suggesting that IME can help capture the flow of information between different speakers. In Negative problems, the full model also has a strong advantage in negation problems, with an improvement of nearly 5 percentage points in EM, suggesting that the Logical Reasoning Perceptual Representation helps to deal with complex linguistic structures such as negation.

The experimental results from Table 5 show that IME provides significant enhancement for complex problems such as inference problems, multiple reference problems and negation problems. For simple factual problems, although IME provides some enhancement, its contribution is more evident in problems that require complex reasoning and cross-speaker information association. This validates the effectiveness of introducing IME and external knowledge injection for complex reasoning problems.

3.9 Impact of External Knowledge Infusion

In order to analyze the impact of external knowledge injection (SQuAD 2.0) on QA inference for multi-party conversations, we designed to pre-train the model using different types of external knowledge and compare their performances in multi-party conversation tasks. The experiments can be categorized into three groups: no external knowledge (No External Knowledge), flat text knowledge that is not relevant to multi-party dialogues (SQuAD 2.0), and knowledge that is relevant to the dialogues (social media dialogue data).

External Knowledge Type	Molweni		FriendsQA	
	EM	F1	EM	F1
No External Knowledge	58.4	73.0	56.8	72.9
SQuAD 2.0 (Plain Text QA)	**60.2**	**74.5**	**60.5**	**77.6**
Dialogue-related Data	59.8	74.1	59.7	76.4

Table 6: Performance impact of external knowledge injection: comparing the impact of different external knowledge types on multi-party dialogue QA.

The model without external knowledge performs significantly worse, especially with lower F1 scores on the FriendsQA dataset. This suggests that with-

out the injection of external knowledge, the model performs poorly on complex reasoning tasks. The SQuAD 2.0 dataset provides a significant performance improvement, especially on the FriendsQA dataset, which is complex in its reasoning, with an increase in F1 scores of almost 5 percentage points. This shows that even flat text QA data, which is not dialogue related, can significantly enhance the model's inference. Conversation-related data also provides a performance gain, but slightly less so than SQuAD 2.0. While conversation-related data may be better at capturing the structure of conversations, SQuAD 2.0 provides richer training in logical reasoning, and is therefore slightly better at reasoning.

The experimental results from Table 6 show that external knowledge is crucial to the improvement of the model's reasoning ability. Even flat text QA data unrelated to conversations (SQuAD 2.0) significantly improves the performance of multi-party dialogue QA, especially on tasks involving complex reasoning. Conversation-related external data is helpful in improving the understanding of conversation structure, but is slightly weaker than SQuAD 2.0 in logical reasoning; therefore, a judicious choice of external knowledge sources can effectively enhance the model's reasoning ability.

3.10 Comparison of Different Reasoning Mechanisms

In order to verify the effectiveness of Inference Memory Encoder (IME) relative to other common inference mechanisms (e.g., Graph Neural Networks, Attention-based mechanisms), we designed comparative experiments to compare the performance of different inference mechanisms on multi-party dialogue QA tasks. Common comparative reasoning mechanisms include Graph Neural Network (GNN), Attention-based Reasoning and Inference Memory Encoder (IME). The experiment aims to compare the performance of these mechanisms on the Molweni and FriendsQA datasets in order to validate the advantages of IME when dealing with complex reasoning tasks.

Attention-based reasoning mechanism, which performed relatively well on both datasets but slightly underperformed the F1 score on the more complex FriendsQA dataset, suggesting that this approach is slightly less effective at handling complex reasoning in multi-party conversations. The GNN, on the other hand, performs better in dealing with relationships between different speakers in a conversation, and in particular slightly outperforms the attention-based inference mechanism on FriendsQA, showing its ability to capture speaker relationships in graph structures. IME outperforms the other two inference mechanisms on both datasets, especially on the more complex FriendsQA dataset, where the improvement in F1 score is more significant. This suggests that IME is more capable of capturing complex reasoning relationships and the logical structure of multi-party conversations.

Reasoning Mechanism	Molweni		FriendsQA	
	EM	F1	EM	F1
Attention-based	58.5	73.2	57.4	74.0
GNN	59.1	73.8	58.2	74.6
IME (ours)	**60.2**	**74.5**	**60.5**	**77.6**

Table 7: A Comparison of Different Reasoning Mechanisms: the Performance of IME, GNN, and Attention-Based Reasoning Mechanisms in Multi-Party Dialogue QA Tasks.

Experimental results from Table 7 show that the Inference Memory Encoder (IME) has significant advantages over graph neural networks and attention-based inference mechanisms in the multi-party dialogue QA task. the IME is better able to capture logical reasoning cues in multi-party dialogues and can utilize external knowledge injection to enhance the reasoning ability. As a result, IMEs show greater versatility and effectiveness in scenarios that require complex reasoning.

4 Conclusions

This paper introduces LIMN, a model enhancing QA tasks in multiparty dialogues by using QA-related latent memory and key-utterance-based interactions. It decouples dialogue at speaker and utterance levels, achieving state-of-the-art results on Molweni and FriendsQA datasets. Future work will focus on refining external knowledge for improved QA inference.

Limitations

Computational Resource Requirements: We have acknowledged the high computational demands, which may impact accessibility for smaller research teams.

Dependency on External Knowledge Quality: The model's inference performance is influenced by the quality of external knowledge injection, and reliance on datasets such as SQuAD may limit its

robustness to diverse dialogue contexts.

Acknowledgments

This article is the research result of the National Natural Science Foundation of China (No. 72074108) and the Special Fund for Basic Scientific Research Business of Central Universities project at Nanjing University, and is supported by the Jiangsu Young Talents in Social Sciences and Tang Scholar of Nanjing University.

References

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*.

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, Hong Kong, China. Association for Computational Linguistics.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020a. 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Jiaqi Li, Ming Liu, Min-Yen Kan, Zihao Zheng, Zekun Wang, Wenqiang Lei, Ting Liu, and Bing Qin. 2020b. 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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

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. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Yanling Li, Bowei Zou, Yifan Fan, Mengxing Dong, and Yu Hong. 2023. Coreference-aware Double-channel Attention Network for Multi-party Dialogue Reading Comprehension. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Gold Coast, Australia. IEEE.

Yiyang Li and Hai Zhao. 2021a. Self- and Pseudo-self-supervised Prediction of Speaker and Key-utterance for Multi-party Dialogue Reading Comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2053–2063, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yiyang Li and Hai Zhao. 2021b. Self-and pseudo-self-supervised prediction of speaker and key-utterance for multi-party dialogue reading comprehension. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2053–2063.

Yiyang Li, Hai Zhao, and Zhuosheng Zhang. 2022a. Back to the future: Bidirectional information decoupling network for multi-turn dialogue modeling. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2761–2774, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yiyang Li, Hai Zhao, and Zhuosheng Zhang. 2022b. Back to the future: Bidirectional information decoupling network for multi-turn dialogue modeling. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2761–2774.

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, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Longxiang Liu, Zhuosheng Zhang, Hai Zhao, Xi Zhou, and Xiang Zhou. 2021a. Filling the gap of utterance-aware and speaker-aware representation for multi-turn dialogue. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13406–13414.

Yongkang Liu, Shi Feng, Daling Wang, Kaisong Song, Feiliang Ren, and Yifei Zhang. 2021b. A graph reasoning network for multi-turn response selection via customized pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13433–13442.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2023a. Enhanced speaker-aware multi-party multi-turn dialogue comprehension. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2410–2423.

Xinbei Ma, Zhuosheng Zhang, and Hai Zhao. 2023b. Enhanced Speaker-Aware Multi-Party Multi-Turn Dialogue Comprehension. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2410–2423.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018a. Know What You Don't Know: Unanswerable Questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018b. Know what you don't know: Unanswerable questions for squad. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kai Sun, Dian Yu, Dong Yu, and Claire Cardie. 2019. Improving machine reading comprehension with general reading strategies. 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 2633–2643.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

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, Stockholm, Sweden. Association for Computational Linguistics.

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.

Shu Zhou, Xin Wang, Zhengda Zhou, Haohan Yi, Xuhui Zheng, and Hao Wan. 2024. The master-slave encoder model for improving patent text summarization: A new approach to combining specifications and claims. *arXiv preprint arXiv:2411.14072*.

Xingyu Zhu, Jin Wang, and Xuejie Zhang. 2022. An Enhanced Key-utterance Interactive Model with Decoupled Auxiliary Tasks for Multi-party Dialogue Reading Comprehension. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Padua, Italy. IEEE.

A Baseline Introduction

SUP (Self- and Pseudo-Self-Supervised Prediction) (Li and Zhao, 2021b): The model introduces a self-supervised task to predict the speaker and the key words, respectively. In this way, the model can learn the identity of the speaker and the key words containing the answers without human labeling.

SUP+MI (SUP with Mutual Information) (Zhu et al., 2022): Based on the SUP model, SUP+MI enhances the representation of different information flows in the modeled dialogue by introducing Mutual Information. Mutual Information helps to model the relationship between speakers and discourse in a conversation and improves the effect of information decoupling.

SUP+BiDeN (SUP with Bidirectional Information Decoupling Network) (Li et al., 2022a): This model combines SUP with Bidirectional Information Decoupling Network (BiDeN), which enhances logical reasoning in dialogue by interpreting the dialogue context in a forward and backward direction to capture the past, current and future information flow in the dialogue.

MaskAttn+Graph (Masked Attention with Graph Networks) (Ma et al., 2023b): The model uses a masked attention mechanism to decouple speaker and discourse information and combines it with Graph Networks to model discourse relations in multi-party dialogues. Through Graph Networks, the model can capture the complex dependencies between speakers and discourse in a conversation.

CADA (Coreference-Aware Dialogue Attention) (Li et al., 2023): The CADA model focuses on co-referential relations in dialogues, modeling the referential relations between different speakers and discourses in a dialogue through the mechanism of co-reference-aware attention. This approach helps to better understand the semantic cues and inferential relationships in a dialogue.

B Related Work

Existing research on multiparty dialogue understanding can be broadly categorized into two primary directions: graph-based modeling and specialized multi-head attention (MHA) mechanisms.

The first direction focuses on representing dialogue contexts as heterogeneous graphs, where nodes typically represent utterances and edges encode relationships between them. For instance, Ghosal et al. (Ghosal et al., 2019) introduced DialogueGCN, a dialogue graph convolutional net-

work that models both self- and inter-speaker dependencies to capture the nuances of dialogue context. Li et al. (Li et al., 2021) proposed a discourse-aware dialogue graph (DADgraph) that incorporates dependency links and discourse relations to model inter-utterance relationships. Ma et al. (Ma et al., 2023a) and Zhou et al. (Zhou et al., 2024) developed an enhanced speaker-aware model that leverages two graph networks to capture both annotated and unannotated discourse relations, further incorporating a decoupling module based on masking-based multi-head attention to isolate speaker-specific characteristics. Distinct from these approaches, Liu et al. (Liu et al., 2021b) introduced a graph reasoning network (GRN) that utilizes next utterance prediction (NUP) and utterance order prediction (UOP) as pretraining tasks to improve performance on dialogue response selection. This model then employs an utterance dependency graph (UDG) to capture dependencies and facilitate reasoning by propagating information along utterance paths.

The second direction extends the standard MHA mechanism to focus on specific aspects of dialogue crucial for comprehension. Liu et al. (Liu et al., 2021a) proposed a mask-based decoupling fusing network (MDFN) that separates dialogue context based on speaker and utterance levels by incorporating inter-speaker and intra-speaker masks into the MHA. Li et al. (Li et al., 2022b) introduced the Bidirectional Information Decoupling Network (BiDeN), which adapts the back-and-forth reading strategy (Sun et al., 2019) to model temporal characteristics in dialogues. BiDeN analyzes the dialogue context from three perspectives: future-to-current, current-to-current, and past-to-current, using a masking-based MHA mechanism.

Given the inherent complexity and potential for noisy information in multiparty dialogues due to multiple speakers and utterances, Li et al. (Li and Zhao, 2021b) introduced self- and pseudo-self-supervised sub-tasks for speaker and key-utterance prediction (SUP). These auxiliary tasks aim to model the complex information flow by recognizing that not all utterances contribute equally to answering a specific question.

C Training Cost Discussion

Below, we include specific results regarding training times, GPU utilization, and memory requirements associated with our model's key components: We trained on two NVIDIA A100 GPUs (40GB memory). The model was pre-trained and fine-tuned using Mixed Precision to reduce memory usage and increase speed. Our pre-training phase took a total of 24 hours to complete, including 10 epochs, each of which took about 2.4 hours, and used about 35GB of memory. The fine-tuning phase takes 8 hours and 7 hours to complete on each dataset with 5 epochs respectively, and the memory usage is controlled to be around 25GB due to the module freezing operation, which reduces the memory burden and computation overhead. By freezing the pre-trained Inference Memory Encoder, we reduce the computation of parameter update and optimize the use of GPU resources. In the multitasking setting, the time per batch is about 0.5 seconds, and the total time spent is about 7 hours with 50,000 batches. By reducing the number of update modules, we reduced the computational overhead by about 20% in the multitasking learning phase. As a result, our model took a total of 46 hours to complete in the full training process. With specific module freezing and the use of mixed precision, the model can be reproduced in an efficient environment, ensuring reasonable training time and resource requirements.

D Generalization Ability Test

In order to test the generalization ability of the model on different datasets, we designed experiments using two major datasets, Molweni and FriendsQA, for training, and then tested the performance of the model on other datasets such as Ubuntu Dialogue Corpus and DailyDialogue. This helps to verify that the IME has good generalization performance. The experiments will compare the performance of the full model (using IME) with the model without IME on different datasets to see its generalization ability.

Training Dataset	Ubuntu Dialogue Corpus		DailyDialogue	
	EM	F1	EM	F1
Molweni + IME	**55.2**	**68.9**	**62.5**	**74.3**
Molweni (w/o IME)	52.1	66.3	59.4	70.8
FriendsQA + IME	**57.8**	**71.2**	**64.7**	**76.1**
FriendsQA (w/o IME)	54.5	68.4	61.8	72.9

Table 8: Generalizability test: performance of models trained on Molweni and FriendsQA datasets on Ubuntu Dialogue Corpus and DailyDialogue datasets.

Performance on the Molweni training dataset is dominated by the fact that the model significantly outperforms the model without IME on Ubuntu

Dialogue Corpus and DailyDialogue when IME is used, especially on the DailyDialogue dataset, where the F1 score improves by close to 4 percentage points. The performance on the FriendsQA training dataset is better for the model using IME on both test datasets, especially on the Ubuntu Dialogue Corpus dataset with complex dialogue contexts, where the F1 score improves by nearly 3 percentage points, indicating that IME is better adapted to complex reasoning across datasets. The model without IME is slightly less capable of generalization, and performs significantly worse than the model with IME, especially on tasks that require stronger reasoning.

The experimental results from Table 8 show that the generalization ability of the model using IME on different datasets is significantly better than that of the model without IME. This suggests that IME enhances the model's reasoning ability when dealing with unseen datasets, especially in tasks with complex dialogue and reasoning requirements. Overall, IME provides stronger generalization capabilities, allowing the model to maintain high performance in different scenarios.

E Algorithm

Algorithm 1 Logical Inference Memory Network (LIMN) for Multiparty Dialogue QA

Require: Dialogue $D = \{U_1, U_2, ..., U_N\}$, where each utterance U_n has speaker S_n and words D_n.
Require: Question Q.
$H \leftarrow \text{PLMEncoder}(D, Q)$ {See Algorithm 2}
$H_I, H_S, H_K \leftarrow \text{DialogueDecoupleNetwork}(H)$ {See Algorithm 3}
span, answerable, key_utterance, speaker, $\xi \leftarrow$ MultitaskLearningStrategy(H, H_I, H_S, H_K, D, Q) {See Algorithm 4}
return span, answerable, key_utterance, speaker, ξ

Algorithm 2 PLM-based Encoder

Require: Dialogue $D = \{U_1, U_2, ..., U_N\}$, where each utterance U_n has speaker S_n and words D_n.
Require: Question Q.
$x \leftarrow \{[CLS], U_1, [SEP_1], ..., [SEP_N], Q, [SEP_Q]\}$ {Concatenate dialogue and question with special tokens}
$E \leftarrow \text{TokenEmbeddings}(x)$ {Convert tokens to embeddings using a predefined vocabulary}
$H \leftarrow \text{PLM}(E)$ {Encode with Pretrained Language Model (e.g., BERT, ELECTRA). H has dimensions (sequence_length, hidden_size)}
return H {Contextual representations}

Algorithm 3 Dialogue Decouple Network

Require: Contextual representations H from the PLM-based Encoder.
{**Inference Memory Encoder (Pretrained and Frozen)**}
$H_I \leftarrow \text{IME}(H)$ {Encode with Inference Memory Encoder. H_I has dimensions (sequence_length, hidden_size)}
{**Masked Speaker Prediction Block**}
$U_{mask} \leftarrow \text{SelectMaskedUtterance}(D)$ {Select a candidate utterance and mask its speaker}
Perform masked speaker prediction task using U_{mask} and H (Implementation details omitted for brevity but involve comparing U_{mask} with other utterances to determine if they have the same speaker)
$H_S \leftarrow \text{sMHA}(H_{\text{detached}}, H_{\text{detached}}, H_{\text{detached}})$ {Generate speaker-aware representations using multi-head attention. H_S has dimensions (sequence_length, hidden_size)}
{**Key-Utterance Prediction Block**}
Perform key-utterance prediction task using D, Q and H (Implementation details omitted for brevity but involve identifying utterances most relevant to answering the question)
$H_K \leftarrow \text{GenerateKeyUtteranceRep}(H)$ {Generate key-utterance-aware representations from H via backpropagation. H_K has dimensions (sequence_length, hidden_size)}
return H_I, H_S, H_K

Algorithm 4 Multitask Learning Strategy

Require: Contextual representations H, inference-aware representations H_I, speaker-aware representations H_S, key-utterance-aware representations H_K, Dialogue D, Question Q

{**Key-Utterance-based Interaction Mechanism**}
$H_D \leftarrow$ Fusion(H_K, H_S) {Concatenate and linearly transform. H_D has dimensions (sequence_length, fusion_dim)}
$H_d \leftarrow [H_K; H_S; H_K - H_S; H_K \cdot H_S]$ {Concatenate, element-wise subtract, and element-wise multiply. H_d has dimensions (sequence_length, 4*hidden_size)}
$H_F \leftarrow$ Fusion(H_D, H_I) {Concatenate and linearly transform. H_F has dimensions (sequence_length, fusion_dim)}
Decompose H_F into H_F^C (context), H_F^Q (question), H_F^A (answer candidates) {Each of H_F^C, H_F^Q, H_F^A has dimensions corresponding to their respective parts of the sequence}
$H_A \leftarrow$ DUMA(H_F^C, H_F^A) {Dynamic Multi-head Attention. See detailed pseudocode below (Algorithm 5). H_A has dimensions (1, fusion_dim)}
$H_{\text{enhanced}} \leftarrow$ Fusion$(H_F, \text{repeat}(H_A))$ {Concatenate H_F with repeated H_A. H_enhanced has dimensions (sequence_length, 2*fusion_dim)}
{**Span-level Answer Extraction**}
$P_{\text{start}} \leftarrow \text{softmax}(v_{\text{start}}^T H_{\text{enhanced}})$ {v_start is a trainable vector. P_start has dimension (sequence_length)}
$P_{\text{end}} \leftarrow \text{softmax}(v_{\text{end}}^T H_{\text{enhanced}})$ {v_end is a trainable vector. P_end has dimension (sequence_length)}
span $\leftarrow (\text{argmax}(P_{\text{start}}), \text{argmax}(P_{\text{end}}))$ {Predicted start and end indices}
$\xi_{\text{span}} \leftarrow -(\log P_{\text{start}}[y_{\text{start}}] + \log P_{\text{end}}[y_{\text{end}}])$ {Cross-entropy loss, where y_start and y_end are the ground truth indices}
{**Answerable Prediction**}
Split H_K into H_K^C (utterance) and H_K^Q (question)
$H_K^A \leftarrow$ DUMA(H_K^C, H_K^Q) {H_K^A has dimension (1, hidden_size)}
answerable $\leftarrow$ sigmoid(MLP(H_K^A)) {MLP is a multi-layer perceptron. answerable is a probability between 0 and 1}
$\xi_{\text{answerable}} \leftarrow$ BinaryCrossEntropy(answerable, ground truth)
{**Key-Utterance Prediction**}
for $n = 1$ **to** N **do**
    concat $\leftarrow [H_{[SEP_n]}; H_K^A]$ {Concatenate the representation of the [SEP] token for the $n-th$ utterance with H_K^A}
    key_utterance$_n \leftarrow$ sigmoid(MLP(concat)) {Predict if utterance U_n is a key utterance}
end for
key_utterance $=$ [key_utterance$_1$, ..., key_utterance$_N$]
$\xi_{\text{utterance}} \leftarrow$ BinaryCrossEntropy(key_utterance, ground truths) {for all utterances}
{**Masked Speaker Prediction**}
for each pair of utterances (U_i, U_j) **do**
    concat $\leftarrow [H_S[\text{SEP}_i]; H_S[\text{SEP}_j]]$ {Concatenate the representations of the [SEP] tokens for the i-th and j-th utterances from H_S}
    speaker$_{ij} \leftarrow$ sigmoid(MLP(concat)) {Predict if utterances U_i and U_j are from the same speaker}
end for
speaker $=$ {speaker$_{ij}$ for all utterance pairs (U_i, U_j)}
$\xi_{\text{speaker}} \leftarrow$ CrossEntropy(speaker, ground truths) {for all utterance pairs}
{**Combined Loss**}
$\xi \leftarrow \xi_{\text{span}} + \xi_{\text{answerable}} + \xi_{\text{utterance}} + \xi_{\text{speaker}}$
return span, answerable, key_utterance, speaker, ξ

Algorithm 5 Dynamic Multi-head Attention (DUMA)

Require: Context representations H_F^C, Answer candidate representations H_F^A.

$H_F^{'C} = \text{MHA}(H_F^C, H_F^A, H_F^A)$ {Multi-head attention of context with answer candidates as key and value. $H\_F^{'C}$ has the same dimension as H_F^C}
$H_F^{'A} = \text{MHA}(H_F^A, H_F^C, H_F^C)$ {Multi-head attention of answer candidates with context as key and value. $H\_F^{'A}$ has the same dimension as H_F^A}
$H_A =$ Fusion$(\text{mean}(H_F^{'C}), \text{mean}(H_F^{'A}))$ {Concatenate and linearly transform the mean of $H\_F^{'C}$ and $H\_F^{'A}$. H_A has dimension (1, fusion_dim)}
return H_A