The Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)

Multi-Agent Discussion Mechanism for Natural Language Generation

Xu Li, Mingming Sun, Ping Li
Cognitive Computing Lab (CCL), Baidu Research
{lixu13,sunmingming01,liping11}@baidu.com

Abstract

We introduce the discussion mechanism into the multi-agent communicating encoder-decoder architecture for Natural Language Generation (NLG) tasks and prove that by applying the discussion mechanism, the communication between agents becomes more effective. Generally speaking, an encoder-decoder architecture predicts target-sequence word by word in several time steps. At each time step of prediction, agents with the discussion mechanism predict the target word after several discussion steps. In the first step of discussion, agents make their choice independently and express their decision to other agents. In the next discussion step, agents collect other agents' decision to update their own decisions, then express the updated decisions to others again. After several iterations, the agents make their final decision based on a well-communicated situation. The benefit of the discussion mechanism is that multiple encoders can be designed as different structures to fit the specified input or to fetch different representations of inputs.We train and evaluate the discussion mechanism on Table to Text Generation, Text Summarization and Image Caption tasks, respectively. Our empirical results demonstrate that the proposed multi-agent discussion mechanism is helpful for maximizing the utility of the communication between agents.

1 Introduction

Natural Language Generation (NLG) (Kukich 1983; Reiter and Dale 1997) is a challenging task of generating texts on the condition of specified information such as knowledge, images or texts. The challenge lies in understanding the input information and organizing the texts to be generated, known as *what to say* and *how to say it* (Reiter and Dale 1997; Wiseman, Shieber, and Rush 2017). In recent research, the sequence to sequence structure (Sutskever, Vinyals, and Le 2014) has been widely applied in NLG tasks. More concretely, an encoder is adopted to generate a representation of the input information, and a decoder is used to predict the natural language with respect to the encoded inputs. Attention mechanism (Bahdanau, Cho, and Bengio 2014) is often adopted at each time step of the decoder to select a better representation of input information.

As NLG tasks become increasingly complex, it is common that the input information is complex (Celikyilmaz et al. 2018) or has multiple views (Wiseman, Shieber, and Rush 2017). For example, in the NBA game summarization task (Wiseman, Shieber, and Rush 2017), texts are generated from the given information of an NBA match, the input knowledge has multiple views such as background information about teams, score records through timeline or the performances of each player. Another example is text summarization task in which text inputs are too long to be handled by a single encoder. In these cases, a good understanding and representation of the input information is critical in order to generate correct texts. Given the complexity of real world environments and limited capability or visibility of a single agent, the multi-agent system worths an in-depth study.

In recent research, many multi-agent communicating systems (Sukhbaatar, Szlam, and Fergus 2016) have been proposed and adopted to a variety of collaborative tasks such as logic puzzles (Foerster et al. 2016), visual dialog (Das et al. 2017), reference games (Lazaridou, Pham, and Baroni 2016) and information extractions (Sun, Li, and Li 2018). All those applications achieve better results when the input information becomes more complex. Among those systems, communication plays a vital role in the coordinative behavior of agents. Agents communicate with each other by sending messages which designed manually or learned from data. For example, (Lazaridou, Peysakhovich, and Baroni 2016) learns to generate human language as messages sharing between agents. In a recently proposed deep communicate model (Sukhbaatar, Szlam, and Fergus 2016), the agents learn to represent their observed information or decision as a continuous vector and share it with other agents.

In the field of NLG, (Celikyilmaz et al. 2018) first introduced deep communicating multi-agent system to text summarization tasks to obtain a better representation of long text inputs. In their work, multiple encoders share their previous layer's encoded vectors to the next layer of other encoders. For each time step of encoders, agents make their decision based on how other agents encoded input information in previous layers. However, how the other encoders encode input information in the previous layer may not be valuable information that can help the agent making their decision of the current layer at the current time step. Besides, communication in encoder request the multiple encoders has the same structure which limited the application and capacity of such multi-agent system. We hypothesize that agents could make

a better decision based on a well-informed situation, that is they can adjust their decision on the condition of other agent's decisions. To achieve that, we introduce discussion mechanism to multi-agent system, which is helpful in effective communication and loosen the constrain on encoders' structure.

In this paper, the discussion mechanism is introduced to the multiple agents system with the goal of maximizing the communicative utility among agents. Instead of implementing communication between layers of multiple encoders, the discussion mechanism is introduced to each step during the prediction processing. Specifically, at each time step of decoder, agents discuss with each other in several rounds, then an action is chosen based on a well-communicated environment. In the first round of discussion, agents make their decision independently, and share their decision to other agents. In the next round, an agent is informed with the knowledge of other agents' independent choice and adjust its decision to achieve a better result. After several iterations, agents make their final decision based on a fully informed situation. Then the decoder update it's hidden state of current time step based on the output of discussion mechanism.

We train and evaluate the proposed discussion mechanism on *three* types of input information. The first one is the NBA game *description generation* task, different views of an NBA match is organized as several tables. The experimental result shows that the discussion mechanism can help to utilized input information of different views of an entity (NBA game in this case). Next, we adopt the multi-agent model with discussion mechanism to the *text summarization* task, in this experiment, text to be summarized are too long to be encoded by a single encoder agent, so we divide the long input text into several pieces and feed them to multiple agents. The results in this test demonstrate that multi-agent system with discussion mechanism can achieve better performance in the tasks with partially visible inputs. Lastly, we evaluate discussion mechanism on the *image caption* task. We use pre-trained image classification models as the image feature extractors. The experiments indicate that the proposed discussion mechanism can read and reorganize the encoded features from the different structure of encoders to obtain a more valuable representation for further generation.

The contribution of this paper is three-fold:

- We propose discussion mechanism of multi-agent communicate system and apply it to text generation tasks.
- With discussion mechanism, the communication between agents becomes more effective, all the action making by the agent are based on the well communicated situation.
- The proposed method does not constrain the multiple encoders to have the same structure. The encoders can be designed compatible with the multiple inputs.

In the rest of this paper, we first introduce our model framework details in Section 2 and describe experiments setups in Section3. The experiment results and analysis are presented in Section 4. We provide the conclusion and future work at Section 5.

2 Models

2.1 Problem Definition

Our model is driven from a traditional attention encoder-decoder architecture in sequence to sequence learning. We adopt multiple encoders to model the multiple inputs and add discussion mechanism in attention decoder to receive encoded representations and discuss how to utilize them to obtain a better representation of the complex inputs. The set of multiple inputs of our model is defined as $\mathcal{X}$:

$$\mathcal{X} = \{X_i | i \in (0, N_E)\} \tag{1}$$

where N_E is the number of encoders, X_i is the input information of the i-th agent. In practice, X could be images, texts, tables or speech signal, etc. If the input can be represented as sequence, we can define

$$X = \{x^t | t \in (0, L)\} \tag{2}$$

where L is the input sequence length. The output of our model is defined as Y:

$$Y = \{y^t | t \in (0, T)\} \tag{3}$$

where T is the length of ground truth sequence. To generate the target sequence, we adopt Recurrent Neural Network (RNN) (Medsker and Jain 1999) as the decoder. The decoder decodes the encoded features step by step on the condition of previous states and context vectors, then feeds the decoded features to the prediction layer. The prediction layer finally predicts the probability distribution over the vocabulary. The model framework taking three inputs as example is illustrated in Figure 1.

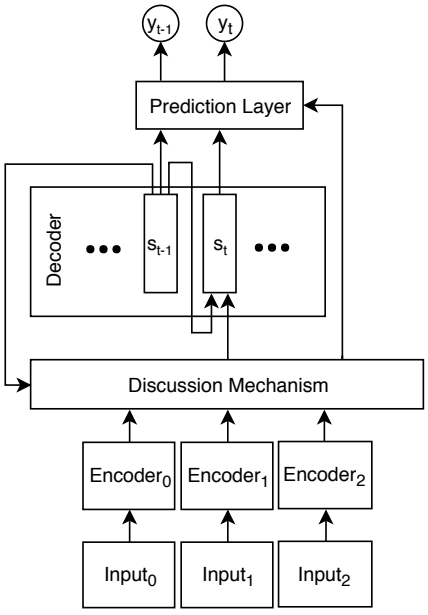

Figure 1: The multi-agent model with discussion mechanism. Multiple inputs are encoded by multiple encoders. The encoded features are then fed into the discussion mechanism. Decoder updates its state step by step with respect to the output of discussion mechanism. Tokens are emitted by the prediction layer on the condition of decoder states.

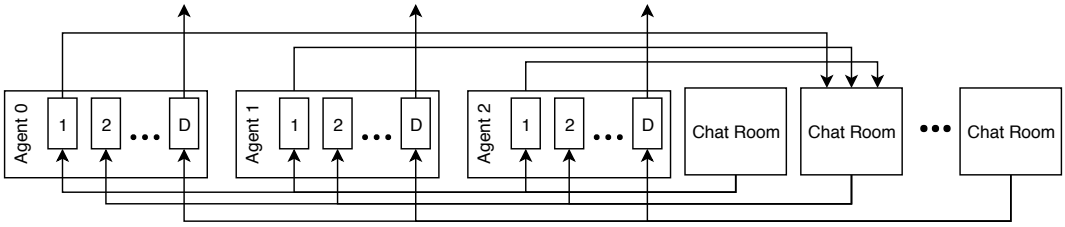

Figure 2: Discussion Mechanism Rolling Out on Discuss Rounds. The multiple agents are sharing messages through chat room. Chat room is initialized as zero matrix and maintains the messages from all agents. In each discussion round, Agents fetching messages from the discussion channel and update its action with respect to other agents' messages, then export its updated message to the chat room. After several iterations, agents reach a consensus at discussion round D. The consensus on a well communicate situation are collected and output to successors.

2.2 Multiple Encoder Agents

In this paper, we consider the multiple encoder agents $\{\mathcal{A}_i | i \in (0, N_E)\}$, in which each agent $\mathcal{A}^i$ generates a representation for the corresponding inputs X_i. For the i-th encoder, we define the encoded representations from X_i as H_i.

$$H_i = \mathcal{A}_i(X_i) \tag{4}$$

The encoder function $\mathcal{A}_i(*)$ can be a gated Recurrent Neural Network such as Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber 1997) or Gated Recurrent Unit (GRU) (Cho et al. 2014) to model sequence inputs or Convolutional Neural Networks (CNN) (Krizhevsky, Sutskever, and Hinton 2012) to encode image inputs or other feature extraction models. We will provide implement details in the section for experiments. We represent H as a sequence of features. If the input is a sequence and encoder is an RNN model, then h_i^t is corresponding to x_t in input X_i. If the input is not a sequence (such as an image), then we reorganize the encoder output to a sequence structure.

2.3 Discussion mechanism

The encoded features H_i represent a different part or view of input information. While deciding the current time step's prediction, the decoder should know which feature is more important to current prediction. For a single encoder-decoder architecture, attention mechanism is adopted to learn a distribution over encoder sequences. The final encoder feature is the weighted sum of H_i over that distribution. However, the problem becomes more complex in multi-agent decision system. To learn a reasonable weight for features, the agent needs to be aware of other agent's choice and the information it obtains from the corresponding encoder. Hence, the agent has to share learnable messages to others and collect other agents' messages to support its decision.

As shown in Figure 2, the proposed discussion mechanism allows agents to communicate in several rounds. Agents export their messages to a chat room and in the same time collect other agent's messages from it. The chat room is initialized with all zero vectors, and maintains agents' messages during the discussion procedure. In the first discussion round, agents make their first decision independently. In the following rounds, agents collect messages produced by other agents, which contain the information of agents' actions, and make their next decision based on these messages. After several iterations, agents export their final decision on a well-communicated basis. Algorithm 1 describes the discussion mechanism of a single decoder time step.

Algorithm 1: The discussion mechanism for a single time step t in the decoder

```
Initialize chat room messages M as zero matrix;
for each discussion round do
    for each agent A_i do
        fetching messages from chat room;
                A_i ← M
        update context vector c_i^t by Eq. (5) to (7);
    end
    for each agent A_i do
        export its context vector to chat room;
                c_i^t → M
    end
    update chat room by Eq. (8);
end
Output collection {c_i^t}.
```

Formally, to predict an attention weight on the j-th encoded features of the i-th input at time step t, agent i should be considering the decoder state s^{t-1} at previous time step, encoded feature h_i^j, and most importantly, the information in chat room $M^{t(d-1)}$ at the last discussion round all together. The attention with discussion mechanism is formulated as in Eq (5) to (7) (For simplicity, the bias item of linear transformation is omitted in this paper).

$$e_i^{tdj} = V_i^T \tanh(U_i h_i^j + W_i s^{t-1} + D_i M^{t(d-1)}) \tag{5}$$

$$\alpha_i^{tdj} = \frac{\exp(e_i^{tdj})}{\sum_{k=1}^{L_i} \exp(e_i^{tdk})} \tag{6}$$

$$c_i^{td} = \sum_{j=1}^{L_i} \alpha_i^{tdj} h_i^j \tag{7}$$

where e represents the attention energy vector, α is the attention distribution, c specify the context vector. In subscript, i

specify the agent id. On the superscript, t is the time step of decoder, d is the round of discussion, j specify the position of encoded feature. s represent the state in an RNN decoder, M^{td} is the information maintained in meeting room, L_i is the sequence length of the i-th input. V_i, U_i, W_i, D_i and b_i are trainable variables.

The context vector c_i^{td}, which can be seen as the fixed representation of what has been read from the i-th encoder's features at decoder time step t and discuss step d, is used as communicate messages export to chat room. M^{td} is initialized with zeros and updated with Eq. (8) by concatenating those attention context vectors into a matrix:

$$M^{td} = \begin{cases} O, & d = 0 \\ \text{concat}(\{c_i^{td}\}_{i=0}^{N_E}), & d \neq 0 \end{cases} \tag{8}$$

where O represents zero matrix, $concat$ means concatenating vectors into matrix.

After D discussion rounds, the agents are fully informed with other agents' messages. We use the i-th agent's context vector c_i^{td} at the last discuss round $d = D$ as the final context vector at decoder time step t. That is, the discussion mechanism finally outputs a set of context vectors $\{c_i^t = c_i^{tD} | i \in (0, N_E)\}$, with the attention probability of agent $\mathcal{A}_i$ on the encoded feature j:

$$a_i^{jt} = \alpha_i^{tDj}. \tag{9}$$

2.4 Decoder

To utilize multiple context vectors, a learnable selective gate is adopted to assign a weight to each context vector:

$$g_i^t = \text{softmax}(V_0 \tanh(W_{gs} s^t + W_{gc} c_i^t)) \tag{10}$$

where V_0, W_{gs}, W_{gc} are trainable variables, g_i^t is the selective distribution on the multiple encoders. The final context vector c_*^t is computed as:

$$c_*^t = \sum_i g_i^t c_i^t \tag{11}$$

Text generation is a kind of sequence generation task and hence we adopt a gated RNN structure such as Long Short term memory (LSTM) or Gated Recurrent Unit (GRU) as decoder to model sequence context information and update decoder state step by step:

$$s^t = \varphi(y^{t-1}, [s^{t-1}, c_*^t]) \tag{12}$$

where φ is a gated function such as LSTM or GRU, y^{t-1} is the representation of the previous word, and the $[a, b]$ is the concatenation of the respective linear transformations of a and b.

As the state of current time step is calculated, we use two linear layers to predict the probability of tokens at time t:

$$P_{vocab}^t = \text{softmax}(V_2 \cdot \text{relu}(V_1 \cdot [s^t, c_*^t])) \tag{13}$$

where V_2, V_1 are trainable variables. P_{vocab}^t is the emitted probabilistic distribution over the vocabulary.

When the input of some agent is text sequence, it would be good to consider the possibility that the generated word is copied from the input text. It is extremely useful for OOV words in the input text, but also helps for general words. Pointer Network (Vinyals, Fortunato, and Jaitly 2015) is proposed to implement this consideration. It computes a generation probability p_g which determines whether to generate a word from the vocabulary by sampling from P_{vocab}^t, or copying a word from the corresponding agent's inputs sequences.

$$p_g = \text{sigmoid}(W_{ps} s^t + W_{pc} c_*^t) \tag{14}$$

where W_{ps}, W_{pc} are trainable variables. It also computes the probability of the word w coping from source of input sequence i by:

$$C_i^t(w) = \sum_{x_j = w} a_i^{jt} \tag{15}$$

Note that if w is an out-of-vocabulary (OOV) word, then $P_{vocab}^t(w)$ is zero; similarly if w does not appear in the source document, then $C_i^t(w)$ is zero.

The final probability distribution is computed over an extended vocabulary which includes words in the input text sequences that are considered out-of-vocabulary (OOV):

$$P^t(w) = p_g P_{vocab}^t(w) + (1 - p_g) \sum_i g_i^t C_i^t(w) \tag{16}$$

P^t is the final distribution used to select predicted tokens and calculate losses.

2.5 Training

Given an input $\mathcal{X}$ and a ground truth $y^* = \{y_t^* | t \in (0, T)\}$, we use maximum likelihood training for the multi-agent sequence generation with the negative log-likelihood loss:

$$L = -\sum_{t=0}^{T_t} \log p(y_t^* | y_1^* ... y_{t-1}^*, \mathcal{X}) \tag{17}$$

During the training procedure, we use teacher forcing strategy, that is, the y_{t-1} in Eq (12) is set to the previous word in ground truth sequence.

3 Experiment Setup

We design three experiments to test the proposed multi-agent model with discussion mechanism on its natural language generation ability with different types of inputs (text, table, image) and encoders. In Table to Text Generation, we test the proposed model on its ability of reading and reorganizing the information from different views of a single entity. In Text Summarization task, the input is text, and each agent can only see a part of the whole input sequence, the performance is highly relying on the effect of communication. Image Caption test is designed to test the discussion mechanism's capacity of handling encoded features from distinguishing structures of encoders.

3.1 Table to Text Generation

We test the discussion mechanism on the table to text generation task using the ROTOWIRE dataset (Wiseman, Shieber, and Rush 2017). ROTOWIRE uses professionally written, medium length game summaries targeted at fantasy basketball fans. The writing is colloquial, but relatively well

structured, and targets an audience primarily interested in game statistics. The dataset consists of articles summarizing NBA basketball games, paired with their corresponding information tables. The NBA match's information is given by four tables. The first table records the summary information of the match including the match time, host team name, visit team name. The second table and third table records the match information of host team and visit team such as scores, fouls, field goals, free throws, and so on. The fourth table is player performance table. It records the performance of each player, for example, the position, assists, points, blocks, and so on.

Key	Value
home name	Hawks
home city	Atlanta
visit name	Magic
visit city	Orlando
date	02_27_15

Table 1: Key-Value Tables in ROTOWIRE Dataset

We first transpose the knowledge in the tables to text sequence. For the first three table, each table has two columns and several lines, the first column is the key, while the second column is the corresponding value. An example of such key-value table is illustrated in Table 1. We transpose the knowledge contained in the table as a key-value sequence: $key_0, value_0; key_1, value_1; ....key_n, value_n$. The player performance table has 26 columns and 24 lines, each column represent a player in the game, each line represents a specified performance.

Attribute	Player 1	Player 2
AST	16	5
BLK	2	8
PLAYER NAME	Join Jenkins	Mike Muscala

Table 2: Player Performance Table in ROTOWIRE Dataset.

For example, in Table 2, the entry in column 1 and line 1 represents the number 1 player's performance on assists. If we flat the table as triplets (a single value in the table can be translated as $player, attribute, value$) to sequence directly, it would be 1,872 elements in an input sequence. To reduce the input sequence length, we view the table as two dimensions, one is the player dimension, while the other is attribute dimension. Input sequences are reorganized on the two dimensions. On player dimension, the sequence has 26 elements, and each element contains the information in the corresponding column, we concatenate and normalize those integer values to a vector, and obtain the embedding vector of the text values, then concatenate them as one element of the input sequence. On the attribute dimension, the input sequence has 24 elements, and each element is the embedding vector of the attribute. To sum up, we have five sequences as the input to the multiple encoders, each describing one aspect of an NBA game.

The player performance table contains both player-wise and performance-wise information. We apply Pointer-Generation Network slightly differently from the formulas proposed in Section 2. In this experiment, the attention distribution a of input sequences 4 and 5 coming from the player performance table in the dataset have dimensions of 26 and 24, respectively. To obtain a copy element from the origin table, we multiply the attention distributions a_4 and a_5 to obtain a probability matrix, in which values d_{mn} specify the current probability of coping from the table's corresponding position (m, n):

$$d^t_{mn} = a^{mt}_4 a^{nt}_5 \tag{18}$$

where m, n specify the row and column numbers in the table. We then calculate copy probability of word w in the extended vocabulary:

$$C^t_4(w) = \sum_{PPT_{mn}=w} d^t_{mn} \tag{19}$$

where PPT_{mn} represents the word of player performance table at position (m, n), w is the word in extended vocabulary. Besides, we need to recalculate the selective gate probability of C_4 to the sum of selective gate value of inputs 4 and 5 as $g^t_4 = g^t_4 + g^t_5$. The final probability distribution over the extended vocabulary is calculated as follows:

$$P^t(w) = p_g P^t_{vocab}(w) + (1 - p_g) \sum_{i=1}^{4} g^t_i C^t_i(w) \tag{20}$$

In this experiment, we split the dataset to 3,398 for training, 750 for validation and 750 for testing. We build word dictionary up to 6,000 most frequent words among 11.3K vocabulary contained in the dataset. We adopted 2 layer bi-directional GRU as the encoder agents and 1 layer GRU as decoder function in Eq. (12). We initialized five agents in the experiment, each for one input sequence. Hidden state dimension is set to 600, and the embedding dimension is set to 128. We train the model by Adam optimizer with initialized learning rate 0.001. Clipped gradient by 1 and L2 regularization has been adopted to avoid gradient explosion. Batch normalization has been adopted in each time step of decoder GRU to accumulate training process.

Baselines of the dataset, including Joint Copy with Reconstruction loss (JC), Joint Copy with Reconstruction loss and Total Variation Distance extensions (JCT) and Conditional Copy (CC), have been proposed by (Wiseman, Shieber, and Rush 2017). To evaluate the proposed model, we compare the BLEU (Papineni et al. 2002) score and perplexity between baseline and multi-agent models with different discussion round number. We set discussion rounds to 1-5 to test the performance of the discussion mechanism.

3.2 Text Summarization

We evaluated the proposed model on CNN/DailyMail (Nallapati, Zhai, and Zhou 2017; Hermann et al. 2015) dataset. Preprocessing steps of (Celikyilmaz et al. 2018) have been replicated to obtain the same data splits. Similarly, we do not anonymize named entities during preprocessing. The preprocessed data has 287,226 training pairs, 13,368 validation pairs, and 11,490 test pairs.

During training and testing, we truncate the article to 800 tokens and limit the length of the summary to 100 tokens for training and 110 tokens at test time. We distribute the truncated articles among agents for multi-agent models, preserving the paragraph and sentences as possible. The input and output vocabulary size is set to 50,000 most frequent tokens in the training set. We adopted a 2-layer bi-directional LSTM as the encoder agents and 1 layer LSTM as decoder function in Eq. (12) to model the input text's context information. We adopted 3 agents with 3 discussion round in the experiment. The hidden dimension is set to 512, and the embedding dimension is 128. Ada-grad (Duchi, Hazan, and Singer 2011) with a learning rate of 0.15 and an initial accumulator value of 0.1 has been adopted as the optimizer during training. We use gradient clipping with a maximum gradient norm of 2.

We evaluate our system using ROUGE-1 (unigram recall), ROUGE-2 (bigram recall) and ROUGE-L (longest common sequence) (Lin 2004). We select the models with the highest ROUGE-L scores on the validation data to evaluate on the test set. At test time, we use beam search with size 4 to generate final predictions.

We compare the proposed models against previously published models: SummaRuNNer (Nallapati, Zhai, and Zhou 2017), a graph-based attentional neural model (Tan, Wan, and Xiao 2017), Pointer-networks with and without coverage (See, Liu, and Manning 2017), Controllable Abstractive Summarization (Fan, Grangier, and Auli 2018) which allows users to define attributes of generated summaries and also uses a copy mechanism for source entities and decoder attention to reduce repetition, and Deep communicate agents which communication occurs between different encoders layer (Celikyilmaz et al. 2018).

3.3 Image Caption

We perform experiments on the MS COCO (Lin et al. 2014) dataset. The MS COCO dataset contains 82,783 images for training and 40,504 for validation, with each image associated with 5 captions. Note that we do not evaluate it on the test set used for MS COCO Image Captioning challenge; instead we use the publicly available splits used in previous work (Karpathy and Fei-Fei 2015), that is, all 82,783 images from the training set for training, 5,000 images from the validation set for validation and the remaining 5,000 from the validation set for testing. Instead of training the encoder-decoder framework end to end, we adopted two well trained image feature extractors InceptionV3 (Szegedy et al. 2016) and Resnet50 (He et al. 2016) as the encoder functions and fix the parameters in them during training.

We build word dictionary up to 20,000 most frequent words. We adopt a 2-layer GRU as decoder function in formula (12). The number of agents is set to two, one agent for one encoded feature. The hidden state dimension is set to 512, and the embedding dimension is set to 256. We train the model by Adam optimizer with initialized learning rate 0.001. We clip the gradient by 1 and use L2 regularization on the prediction layer.

We test the performance of discussion mechanism with the single encoder of InceptionV3 and Resnet50 respectively. Besides, we set the discussion round to 1, 3 and 5 to test the performance influence by discussion round. We evaluate our system using BLEU score from unigram to 4-gram, ROUGE-L (longest common sequence), and CIDEr (Vedantam, Lawrence Zitnick, and Parikh 2015). We select the models with the lowest cross entropy loss on the validation data to evaluate on the test set. At test time, we use beam search with size 5 to generate final predictions.

4 Experiment Results

4.1 Table to Text Generation

Table 3 shows the experiment result on Table to Text Generation experiment. The first three rows are baselines proposed in (Wiseman, Shieber, and Rush 2017), and the last five lines are Multi-agent discussion mechanism result with discussion round from 1 to 5.

Model	PPL	BLEU
JC (Wiseman, Shieber, and Rush 2017)	7.47	10.88
JCT (Wiseman, Shieber, and Rush 2017)	7.42	12.96
CC (Wiseman, Shieber, and Rush 2017)	7.67	14.49
Multi-agent Discussion Round 1	**7.41**	15.32
Multi-agent Discussion Round 2	7.47	15.28
Multi-agent Discussion Round 3	7.45	**15.88**
Multi-agent Discussion Round 4	7.50	15.21
Multi-agent Discussion Round 5	7.59	14.50

Table 3: Performance evaluation of the discussion mechanism in the task of Table to Text Generation.

According to the results, all multi-agent models show improvements over the single agent baselines. The multi-agent model with 3 discussion rounds achieves the best BLEU score. If we set the discussion round to 1, the multi-agent model achieves the best perplexity performance.

The result indicates that, with discussion mechanism, multi-agent model can organize the complex inputs from different aspects of an entity to a better representation than a single agent. The model with discussion round 1 achieves better perplexity score. This is probably because, with increasing discussion round, the agent is confused by the information in the chat room. The decrease of BLEU score after discussion round 3 is consistent with the conjecture.

4.2 Text Summarization

The results for Text Summarization are shown in Table 4. The first five rows in the table are for single agent models, and the second last row is the recently proposed multi-agent model, the last row is multi-agent with discussion mechanism. Overall, compared with baselines, our method achieves better performance on the ROUGE-1 and ROUGE-L scores, the ROUGE-2 score is slightly lower than Deep communication model proposed by (Celikyilmaz et al. 2018). Note that we do not use reinforcement learning (RL) methods such as policy gradient to train our model, thus we do not compare the result with models trained by RL.

Several conclusions can be drawn from the results. Firstly, multi-agent system is better than single agent in capturing

Model	ROUGE-1	ROUGE-2	ROUGE-L
SummaRuNNer (Nallapati, Zhai, and Zhou 2017)	39.60	16.20	35.30
graph-based attention (Tan, Wan, and Xiao 2017)	38.01	13.90	34.00
pointer generator (See, Liu, and Manning 2017)	36.44	15.66	33.42
pointer generator + coverage (See, Liu, and Manning 2017)	39.53	17.28	36.38
controlled summarization with fixed values (Fan, Grangier, and Auli 2018)	39.75	17.29	36.54
Multi-agent encoder communicate (Celikyilmaz et al. 2018)	41.11	**18.21**	36.03
Multi-agent discussion mechanism	**44.26**	18.16	**36.63**

Table 4: Performance evaluation of the discussion mechanism in the task of Text Summarization.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	CIDEr
Single Agent InceptionV3	62.8	44.3	30.4	20.8	41.0	60.9
Single Agent Resnet50	63.3	45.4	31.6	21.9	41.4	61.4
Multi-agent Discussion Round 1	62.9	44.2	30.1	20.3	40.9	58.4
Multi-agent Discussion Round 3	**65.3**	**46.7**	**32.6**	**22.6**	**42.7**	**66.6**
Multi-agent Discussion Round 5	61.9	43.3	29.2	19.6	40.2	56.0

Table 5: Performance evaluation of the discussion mechanism in the task of Image Caption.

the information contained in complex inputs. As the input sequence becomes too long to be modeled by a single encoder, all multi-agent systems outperform single models. Secondly, the communication between agents in discussion mechanism model is more effective than in deep communication model where the communication is conducted between different encoder layers. Our model improves, respectively, ROUGE-1 and ROUGE-L scores by 3.15 and 0.60 points over the deep communication model trained without RL and has similar score on ROUGE-2. Thirdly, the significant improvement in the ROUGE-1 score might be due to the discussion conducted in the decoder time step. The discussion mechanism may be capable in modeling the fine-grained sequence information. Communication in several discussion rounds tends to concentrate on details of the environment. Fourthly, the proposed discussion mechanism can well model the partially visible input situation. Given that only a part of the sequence can be read by single encoder, the discussion mechanism plays a vital role for achieving the better performance.

4.3 Image Caption

The test result of image caption on the MS COCO dataset is presented in Table 5, with the first row being the result of InceptionV3 encoder, and the second row representing the performance of Resnet50 encoder. The results of Multi-agent discussion models are shown in the last three rows. We test the discussion mechanism on different discussion round which has been noted after "Discussion Round" in each row. In this experiment, multi-agent with discussion mechanism achieves better performance over all other models. Interestingly, the performances of multi-agent model with discussion round 1 and 5 are worse than the single agent models.

The result demonstrates that our proposed model can be adopted on the multiple encoders with different structures. With differently structured encoders, agents can get different representations from the same input, or the encoders can be designed to be more suitable to the input's characteristics.

The worse performance on discussion round 1 indicates that the discussion between agents plays a crucial role in multi-agent system. If the discussion round is set to 1, there are no valuable messages shared among agents, which means they have to make their decision without informed of any other agents' messages. Hence, the pressure of reorganizing the multiple encoded representations is shouldered on the decoder. If the discussion round is set to 5, agents make their decision after a relatively long discussion, which may cause mixtures between agents. As the discussion round increases, more assumptions of other agents are introduced to chat room, the messages sharing between agents may be polluted by some undesirable noises.

5 Conclusion

In this paper, a framework of multi-agent discussion mechanism is proposed to promote the effectiveness of communication between agents. We adopt such mechanism into an encoder-decoder architecture for the natural language generation tasks. The proposed mechanism can be adopted on distinguishing structures of encoders and allows the model emitted predictions based on a well-communicated situation. During discussion, agents collect and send messages to keep aware of other agents' state and update their actions dynamically. Our experimental results demonstrate that the proposed discussing mechanism is helpful for maximizing the utility of the communication between agents.

Several further studies on the discussion mechanism can be conducted in the future. Firstly, asynchronous communication can be tried, since not every agent needs to communicate at every time step. Besides, the explanation of agents' messages shared with each other deserves in-depth research. Lastly, as the number of agents growth, more efficient communication mechanism are also worth a study.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate.

CoRR abs/1409.0473.

Celikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep communicating agents for abstractive summarization. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1662–1675.

Cho, K.; van Merrienboer, B.; Gülçehre, Ç.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 1724–1734.

Das, A.; Kottur, S.; Moura, J. M. F.; Lee, S.; and Batra, D. 2017. Learning cooperative visual dialog agents with deep reinforcement learning. In *IEEE International Conference on Computer Vision*, 2970–2979.

Duchi, J.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12(Jul):2121–2159.

Fan, A.; Grangier, D.; and Auli, M. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL*, 45–54.

Foerster, J.; Assael, I. A.; de Freitas, N.; and Whiteson, S. 2016. Learning to communicate with deep multi-agent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2137–2145.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hermann, K. M.; Kocisky, T.; Grefenstette, E.; Espeholt, L.; Kay, W.; Suleyman, M.; and Blunsom, P. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, 1693–1701.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Karpathy, A., and Fei-Fei, L. 2015. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3128–3137.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 1097–1105.

Kukich, K. 1983. Design of a knowledge-based report generator. In *Proceedings of the 21st annual meeting on Association for Computational Linguistics*, 145–150.

Lazaridou, A.; Peysakhovich, A.; and Baroni, M. 2016. Multi-agent cooperation and the emergence of (natural) language. *CoRR* abs/1612.07182.

Lazaridou, A.; Pham, N. T.; and Baroni, M. 2016. Towards multi-agent communication-based language learning. *CoRR* abs/1605.07133.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, 740–755. Springer.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*.

Medsker, L., and Jain, L. C. 1999. *Recurrent neural networks: design and applications*. CRC press.

Nallapati, R.; Zhai, F.; and Zhou, B. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, 3075–3081.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, 311–318.

Reiter, E., and Dale, R. 1997. Building applied natural language generation systems. *Natural Language Engineering* 3(1):57–87.

See, A.; Liu, P. J.; and Manning, C. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 1073–1083.

Sukhbaatar, S.; Szlam, A.; and Fergus, R. 2016. Learning multiagent communication with backpropagation. In *Advances in Neural Information Processing Systems*, 2244–2252.

Sun, M.; Li, X.; and Li, P. 2018. Logician and orator: Learning from the duality between language and knowledge in open domain. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2119–2130.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112.

Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; and Wojna, Z. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2818–2826.

Tan, J.; Wan, X.; and Xiao, J. 2017. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, 1171–1181.

Vedantam, R.; Lawrence Zitnick, C.; and Parikh, D. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4566–4575.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, 2692–2700.

Wiseman, S.; Shieber, S. M.; and Rush, A. M. 2017. Challenges in data-to-document generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2253–2263.