Dialogue Quality and Nugget Detection for Short Text Conversation (STC-3) based on Hierarchical Multi-Stack Model with Memory Enhance Structure

Hsiang-En Cherng and Chia-Hui Chang

National Central University, Taoyuan, Taiwan
seancherng.tw@gmail.com, chiahui@g.ncu.edu.tw

Abstract. In this paper, we consider the Nugget Detection (ND) and Dialogue Quality (DQ) subtasks for Short Text Conversation 3 (STC-3) using deep learning method. The goal of NQ and DQ subtasks is to extend the one-round STC to multi-round conversation such as customer-helpdesk dialogues. The DQ subtask aims to judge the quality of the whole dialogue using three measures: Task Accomplishment (A-score), Dialogue Effectiveness (E-score) and Customer Satisfaction of the dialogue (S-score). The ND subtask, on the other hand, is to classify if an utterance in a dialogue contains a nugget, which is similar to dialogue act (DA) labeling problem. We applied a general model with utterance layer, context layer and memory layer to learn dialogue representation for both DQ and ND subtasks and use gating and attention mechanism at multiple layers including: utterance layer and context layer. The result shows that BERT produced a better utterance representation than multi-stack CNN for both DQ and ND subtasks and outperform the baseline models proposed by NTCIR on Ubuntu customer helpdesk dialogues corpus.

Keywords: Dialogue act, nugget detection, dialogue quality, short text conversation

Team name. WIDM

Subtasks. STC-3 Nugget Detection & Dialogue Quality Subtasks

1. Introduction

Automatic question-answering and dialog systems are important applications in enterprise customer services. With such systems, customer service departments are able to save a plenty of time and human resources, and provide a 24-hour chatbot to answer customers' questions. Short text conversation (STC) task is proposed for such goals in NTCIR-12. Various techniques from retrieval-based approaches to generation-based approaches have been studied in STC-1 & STC-2. However, evaluation of STC tasks relied greatly on human annotation. Thus, STC-3 in NTCIR-14

has initiated a new subtask called Nugget Detection (ND) and Dialogue Quality (DQ). The former aims to recognize the purpose or motivation (a total of 7 types) of each utterance in a dialogue, while the latter aims to evaluate the quality of a dialogue by three measures: Task Accomplishment (A-score), Dialogue Effectiveness (E-score) and Customer Satisfaction of the dialogue (S-score).

ND can be considered as a kind of Dialog Act (DA) labeling problem. Most researches consider DA labeling problem as sequence labeling problem and use traditional machine learning methods [12,19,23]. Recently, many deep learning models [2,7,10,11,14,15] are proposed to tackle the problem. However, the golden answer of ND in STC-3 is the utterance's nugget probability distribution instead of a certain nugget tag, thus the evaluation is based on JSD and RNSS scores which measure the probability distribution between outputs and golden answers as defined in [22].

In this paper, we compared several DNN models based on a general model with utterance layer, contextual layer, memory layer and output layer. Since the DQ and ND subtasks use label probability distribution as training data, we apply softmax instead of CRF layer to predict label distribution. In this paper, we report the performance of not just the uploaded model during STC-3 but also the better model with pre-trained BERT word embedding. The former used multi-stack CNN with word2vec input for utterance representation, while the latter use pure BERT for utterance representation. Overall, in both DQ and ND subtasks, the new model results in the best performance than with NTCIR baseline models.

2. Related Work

Short text conversation (STC). Short text conversation (STC) task is proposed in NTCIR-12 as the first step toward natural language conversation for chatbots. For either retrieval-based (STC-1) or generation-based (STC-2) methods, the evaluation usually requires a lot of annotation efforts. Thus, automatic evaluation of dialog quality (DQ) and nugget detection (ND) is an important step to move from one-round conversation to multi-round conversation. The DQ subtask aims to analyze the quality of a given dialogue. The ND subtask is similar to dialogue act (DA) labeling problem, which could be solved using sequence labeling technique or classification problem. Previous researches on STC have investigated different techniques including: Hidden Markov Model [19], Naïve Bayes [12], Conditional Random Fields (CRF) [12,19,23], and deep learning methods [2,7,10,11,14,15]. Early deep learning models rely on CNN and BI-LSTM modules [11]. The CNN-based model outperforms the BILSTM-based model in both SWDA [8] and MRDA [18] datasets. Hierarchical CNN and BI-LSTM models are latter proposed to better represent sentences [2]. For example, [14] applied hierarchical CNN and hierarchical BI-LSTM for sentence and dialogue representation for DA labeling. More recently, CRF-based DNN model such as LSTM+CRF models are proposed in [7,15]. Furthermore, combining hierarchical BI-LSTM structure with CRF layer to represent utterance and dialogue is also studied in [10]. The major difference of the ND task to traditional DA labeling is the output: for each utterance,

the ground truth is not a single label but label distribution. Thus, the performance evaluation is based on JSD and RNSS [22].

Word Embedding. Word embedding is one of the most important techniques in natural language processing. The goal is to map the one-hot encoding of words to a lower dimensional space such that the vector of words represents their meanings and serves as an input to the neuron networks. There are several word embedding algorithms, including Word2Vec [15], GloVe, FastText, ELMo, OpenAI, BERT, etc. Word2Vec is an unsupervised learning algorithm, which trains the vector of words from given corpus by skip-gram and CBOW methods. The former predicts the word by its context and the latter predicts the context by a given word. BERT [5] is built on top of several clever ideas including semi-supervised sequence learning, multi-task training, bi-directional transformer [18], and masked language model. One can use pre-trained BERT for word representation or fine-tune on unlabeled data and train on labelled data for desired task.

3. Dialogue Quality (DQ) Subtask

The goal of DQ subtask is to evaluate the quality of a dialogue by three measures: Task Accomplishment (A-score), Dialogue Effectiveness (E-score) and Customer Satisfaction of the dialogue (S-score). We proposed two models for DQ subtask, the major difference of these two models is sentence representation, embedding layer and utterance layer, one is based on skip-gram with multi-stack CNN and the other is based on BERT structure.

3.1. Memory enhanced multi-stack CNN with gating mechanism for DQ

In this section, we followed the idea of hierarchical CNN in [14] to construct our model but use multi-stack CNN, we apply 2-stack CNN to utterance representation and 1-stack CNN to context representation. There are 5 layers in our proposed model including input embedding layer, utterance layer, dialog context layer, memory layer and output layer (Fig 1). The goal of such hierarchical structure is to decode the input dialogue hierarchically from word, sentence to context, to capture the dependency of words and utterances.

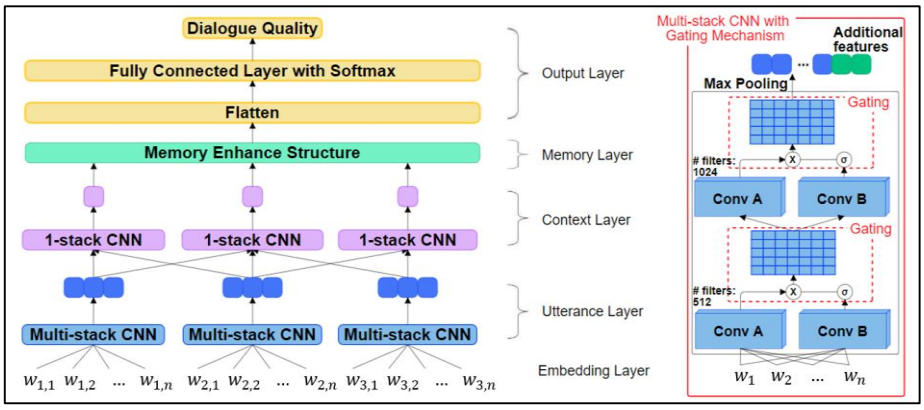

Fig 1. Memory enhance hierarchical gated CNN (MeHGCNN) for DQ subtask

Utterance Layer (UL). CNN with small size of filters is effective in learning sentence representation. We apply 2-stack CNN structure to learn the representation of an utterance. With multi-stack structure, long-range context information could be learned in a small size of filter. For example, filter with size 2 could only learn bi-gram context features. But $n(k-1)+1$ gram features could be learning by applying n-stack structure with the filter size k.

Let u_i denote the ith utterance of a dialogue. Each utterance X_i contains n word tokens $w_{(i,1),}w_{(i,2)}, \dots, w_{(i,n)}$, where $w_{(i,n)}$ denotes the nth token in X_i:

$$X_\mathrm{i} = \left[\mathrm{w}_{(\mathrm{i},1),}\mathrm{w}_{(\mathrm{i},2)}, \dots, \mathrm{w}_{(\mathrm{i},\mathrm{n})}\right] \tag{1}$$

we added gating mechanism and attention mechanism for dialogue quality decision. Gating mechanism is widely used in recurrent neuron network such as LSTM and GRU to control the gates of memory states. The idea of gated CNN is to learn whether to keep or drop a feature generated by CNN. Gating mechanism is implemented by element-wise multiplication and sigmoid function [4] between the output of two convolution operations. Using gating mechanism in multi-stack CNN could generalize features more effectively. Multi-stack CNN with gating mechanism is computed as follows:

$$ulA_i^l = ConvA\left(X_i^l\right) \tag{2}$$
$$ulB_i^l = ConvB\left(X_{i\,\mathrm{i}}^l\right) \tag{3}$$
$$ulC_i^l = ulA_i^l \odot \sigma\left(ulB_i^l\right) \quad if\ \ l \le 2 \tag{4}$$
$$X_i^{l \leftarrow l+1} = ulC_i^l \quad if\ \ l > 2 \tag{5}$$

where X_i^l denotes the utterance vector from lth layer and initializes $l = 1, X_i^l = X_\mathrm{i}$. ulA_i^l denotes the features generated by convolution A, ulB_i^l denotes the gates for all features generated by ulA_i^l. ulC_i^l denotes the utterance representation after applying gating mechanism between ulA_i^l and ulB_i^l where $\odot$ denotes element-wise multiplication and σ denotes sigmoid function. For 2-stack CNN, we execute equation

(2) to (5) for twice. The dimension of ulA_i^l, ulB_i^l and ulC_i^l are equivalent and depend on filter of convolution layer, which is [seqlen,512] for the 1st convolution layer output and [seqlen,1024] for the 2nd convolution layer output, where seqlen denotes the sequence length of an utterance.

In convolution operation, CNN is often followed by max-pooling layer. In this paper, we only apply a max-pooling operation to the output of last convolution stack ulC_i^l then add speaker and nugget information as additional features, which:

$$ul_i = \left[maxpool\left(ulC_i^l\right), speaker_i, nugget_i\right] \quad (6)$$

where ul_i denotes the output of utterance layer, $speaker_i$ and $nugget_i$ denote the speaker and nugget distribution of X_i respectively. The dimension of $maxpool\left(ulC_i^l\right)$ is [1, 1024] and ul_i is [1, 1032] where $|speaker_i| = 1$ and $|nugget_i| = 7$.

Context Layer (CL). Next, we learn the utterance vector with its adjacent utterance in context layer. Taking the utterance vector concatenated with its previous utterance and next utterance with dimension [1, 1032*3] as input. We then apply 1-stack CNN structure to capture the context information between these utterances. The output of context layer contains vectors of each utterance containing context information. Similar with the utterance layer, the operation of 1-stack CNN is computed as equation (2) to (5), the notation of context layer output is $\boldsymbol{cl_i}$ and without any additional features.

Memory Layer (ML). Utterance layer and context layer well capture the context information between adjacent words and utterances but hard to get the long range context features. Therefore, memory network structure [21] is applied to our hierarchical models as Fig 2. Memory network structure is to capture context information by self-attention and feed-forward neuron networks and directly compute the similarity weight between any two utterances, which is able to better represent the context information than Bi-LSTM or multi-stack CNN. For memory layer, first we prepare Input Memory and Output Memory by BI-GRU from context layer $\boldsymbol{cl_i}$:

$$\overrightarrow{I_i} = \overrightarrow{GRU}(cl_i, h_{i-1}) \quad (7)$$
$$\overleftarrow{I_i} = \overleftarrow{GRU}(cl_i, h_{i+1}) \quad (8)$$
$$I_i = tanh\left(\overrightarrow{I_i} + \overleftarrow{I_i}\right) \quad (9)$$
$$\overrightarrow{O_i} = \overrightarrow{GRU}(cl_i, h_{i-1}) \quad (10)$$
$$\overleftarrow{O_i} = \overleftarrow{GRU}(cl_i, h_{i+1}) \quad (11)$$
$$O_i = tanh\left(\overrightarrow{O_i} + \overleftarrow{O_i}\right) \quad (12)$$

where $\overrightarrow{I_i}$, $\overleftarrow{I_i}$ are encoded by BI-GRU to generate input memory I_i, output memory O_i is the combination between $\overrightarrow{O_i}$ and $\overleftarrow{O_i}$ which are also encoded by BI-GRU. The

hidden unit of the BI-GRU for input memory and output memory is 1024, which means the dimension of $I_i = O_i = [1,1024]$.

Second, attention weight is calculated by inner product between current utterance cl_i and input memory I_i, followed by a softmax operation to get attention weight w_i as follows, where k denotes number of utterances in a dialogue.

$$w_i = \frac{exp(cl_i \cdot I_i)}{\sum_{i'=1}^{k} exp(cl_{i'} \cdot I_{i'})} \tag{13}$$

Third, weighted sum between attention and output memory is calculated. The memory layer output of ith utterance ml_i is the addition between the weighted sum of the output memory and the original utterance vector cl_i. The final output of memory layer ml is the concatenation of all k utterances as follows:

$$ml_i = \sum_{i'=1}^{k} w_{i'} \cdot O_{i'} + cl_i \tag{14}$$

$$ml = [ml_1, ml_2, \dots, ml_k] \tag{15}$$

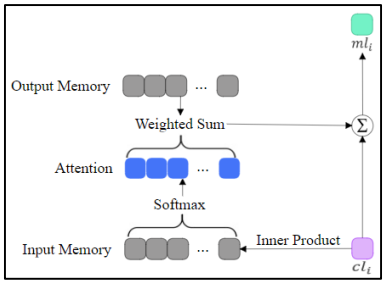

Fig 2. Overview of memory layer

Output Layer. Finally, we apply a fully-connected layer with softmax function to calculate the score distribution for the given dialogue as:

$$fc = mlW_{fc} + b_{fc} \tag{16}$$

$$P(score|dialogue) = \frac{exp(fc_i)}{\sum_{i'=1}^{5} exp(fc_{i'})} \tag{17}$$

where W_{fc} is the transition matrix with dimension $[|ml|, 5]$ where 5 is the numer of score distribution (-2,-1,0,1,2). $P(score|\mathrm{u_i})$ denotes the output score distribution for utterance i in a dialogue.

3.2. Experiments

The STC-3 DQ and ND subtask use Ubuntu customer helpdesk dialogues as the corpus. The training data contains 1,672 dialogues with a total of 8,672 utterances, validation data are randomly selected from training data. Testing data contains 390 dialogues with a total of 1,755 utterances. The label of both DQ and ND subtasks is annotated by 19 students from the department of Computer Science, Waseda University.

For data preprocessing, we remove all full-shape characters and half-shape characters except (A-Za-z!"#$%&()*+,-./:;<=>?@[\]^_`{|}~ ') then apply NLTK tool [1] to convert utterances to sequence of words. For each sentence, we only reserve the first 150 words and drop the remaining words for all utterances. Word2vec [16] is used to train the word embedding model using STC-3 and wiki text8 corpus with word dimension sized 100, window size 5 by skip-gram model.

Result of Dialogue Quality (DQ) Subtask. This section shows the performance of DQ subtask [22] in Normalized Match Distance (NMD) and Root Symmetric Normalized Order-Aware Divergence (RSNOD) as defined in [22]. For hyper-parameter tuning, batch size is 40 and epoch is 50 with early stopping 3. The number of filters for 2-stack CNN of utterance layer is 512 for the 1st stack and 1024 for the 2nd stack. The number of neuron for 1-stack CNN of context layer is 1024. Adam optimizer with 1e-5 learning rate is applied to optimize cross-entropy loss function. The performance of A-score, E-score and S-score are shown in Table 1. MeHGCNN is the model we proposed in section 3.1 and MeGCBERT is the model which replace the embedding layer and utterance layer with BERT. MeGCBERT outperforms MeHGCNN and all NTCIR baseline models. Furthermore, since BERT is a complex model with several bidirectional transformers, we doubt that simple BERT without any complex context and memory layer is able to perform well in DQ subtask. The result shows that even BERT performs well in utterance representation, the context layer and memory information is still necessary for DQ subtask.

Table 1. Performance of DQ subtask

Model	(A-score)		(E-score)		(S-score)	
	NMD	RSNOD	NMD	RSNOD	NMD	RSNOD
BL-uniform	0.1677	0.2478	0.1580	0.2162	0.1987	0.2681
BL-popularity	0.1855	0.2532	0.1950	0.2774	0.1499	0.2326
BL-lstm	0.0896	0.1320	0.0824	0.1220	0.0838	0.1310
BL-BERT	0.0934	0.1379	0.0881	0.1344	0.0842	0.1337
MeHGCNN	0.0862	0.1307	0.0814	0.1225	0.0787	**0.1241**
MeGCBERT	**0.0823**	**0.1255**	**0.0791**	**0.1202**	**0.0758**	0.1245

Table 2 shows the ablation of MeGCBERT model. Both gating mechanism and memory enhance well improve the performance in three types of score. The improvement of A-score and S-score is more significant than E-score. Adding nugget

feature also works well in A-Score and S-Score but only a little improvement in E-score. In summary, all the mechanisms we proposed improve A-score E-score and S-score.

Table 2. Ablation of MeGCBERT

Model	(A-score)		(E-score)		(S-score)	
	NMD	RSNOD	NMD	RSNOD	NMD	RSNOD
MeGCBERT	**0.0823**	**0.1255**	**0.0791**	**0.1202**	**0.0758**	**0.1245**
W/o gating mechanism	0.0885	0.1322	0.0813	0.1214	0.0815	0.1289
W/o memory layer	0.0913	0.1364	0.0808	0.1235	0.0799	0.1273
W/o nugget features	0.0963	0.1388	0.0802	0.1204	0.0774	0.1247

3.3. Learning curve with different training data size for DQ

In this section, we discuss the performance of validation data with MeGCBERT model trained by different size of training data. Fig 3, Fig 4 and Fig 5 show the learning curve of MeGCBERT for each score type. The horizontal axis denotes the proportion of training data we used to train the model and the vertical axis denotes the performance in validation data. The performance of NMD and RSNOD do not significantly improve when using 100% of training data comparing to only 80% training data, which means the number of training data might be enough in DQ subtask. On the other hand, adding more training data might not help in improving the performance in all A-score, E-score and S-score.

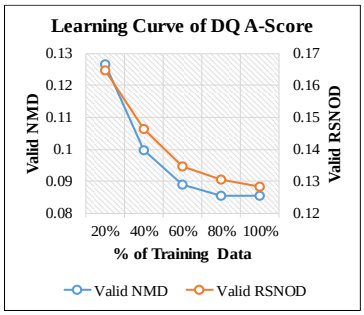

Fig 3. Learning Curve of A-Score

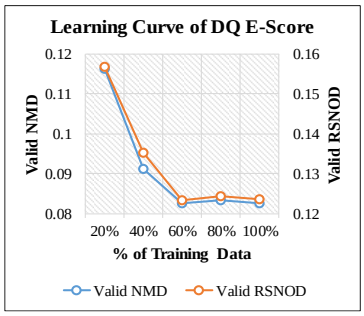

Fig 4. Learning Curve of E-Score

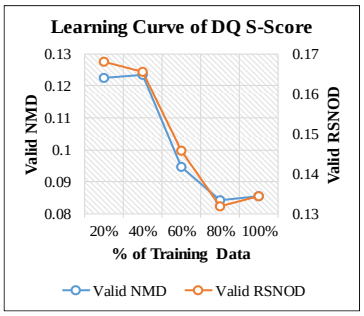

Fig 5. Learning Curve of S-Score

4. Nugget Detection (ND) Subtask

The goal of ND subtask is to classify the nugget type for all utterances in a given customer-service dialogue. There are seven types of nugget for each utterance as shown in Table 3. We proposed two hierarchical models for ND subtask, the major difference is sentence representation, one is based on skip-gram + multi-stack CNN and the other is based on BERT structure.

Table 3. Seven types of nugget for ND subtask

Nugget	Description
CNUG0	Customer trigger: Problem stated
CNUG*	Customer goal: Solution confirmed
CNUG	Customer regular: Utterances contain information to solution
CNaN	Customer Not-a-nugget: Utterances do not contain information to solution
HNUG*	Helpdesk goal: Solution stated
HNUG	Helpdesk regular: Utterances contain information to solution
HNaN	Helpdesk Not-a-nugget: Utterances do not contain information to solution

4.1. Multi-stack CNN with LSTM for ND

In this section, we followed the idea of hierarchical CNN in [14] to construct our model but use 3-stack CNN to represent an utterance. For context representation, we apply 2-stack Bi-directional LSTM (Fig 6). There are 4 layers in our proposed model including input embedding layer, utterance layer, dialog context layer, and output layer.

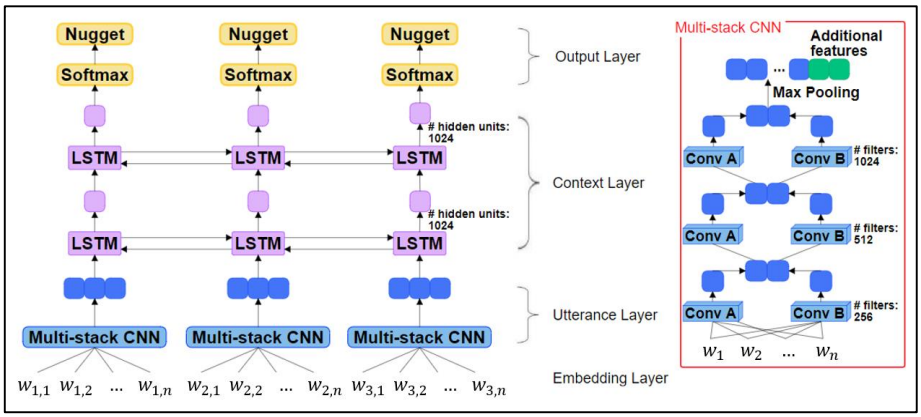

Fig 6. Hierarchical CNN + BI-LSTM (HCNN-LSTM) for ND subtask

Utterance Layer (UL). The multi-stack CNN structure of HCNN-LSTM utterance layer is similar with the one in MeHGCNN we proposed for DQ subtask. Instead, features are prepared by the concatenation between output of convolution A and convolution B. The kernel size of both convolution operations are 2 and 3 which could capture different size of n-gram features. That is,

$$X_{\mathrm{i}} = \left[\mathrm{w}_{(\mathrm{i},1)}, \mathrm{w}_{(\mathrm{i},2)}, \ldots, \mathrm{w}_{(\mathrm{i},\mathrm{n})}\right] \quad (18)$$

$$ulA_i^l = ConvA\left(X_i^l\right) \quad (19)$$

$$ulB_i^l = ConvB\left(X_i^l\right) \quad (20)$$

$$ulC_i^l = \left[ulA_i^l, ulB_i^l\right] \quad (21)$$

$$X_i^{l \leftarrow l+1} = ulC_i^l \quad if \;\; l \leq 3 \quad (22)$$

$$ul_i = [maxpool(ulC_i), speaker_i] \quad if \;\; l > 3 \quad (23)$$

where the dimension of X_{i} is $[\mathrm{emb}, \mathrm{seqlen}]$, emb is the word embedding size, which is fix to 100, and seqlen denotes the number of words in a utterance, which is 150. For 3-stack CNN, equation (19) to (22) is executed for 3 times. $speaker_i$ denotes the speaker of X_{i} with value 1 for customer and 0 for helpdesk. The dimension of ulC_i is [seqlen, 1024] where 1024 is the number of filters of the 3rd convolution layer. Finally, after applying a max pooling operation and concatenating with speaker features, the dimension of utterance layer output ul_i become [1,1025].

Context Layer (CL). Context layer takes the output of utterance layer as input, we then apply 2-stack BI-LSTM structure to capture the context information of adjacent utterances in a same dialogue. With multi-stack BI-LSTM, context information could be learned from long-range utterances. The output of context layer is the utterance vector with context information. Formally, BI-LSTM is computed as follows.

$$\overrightarrow{cl_i^l} = \overrightarrow{LSTM}\left(ul_i^l, h_{i-1}\right) \quad (24)$$

$$\overleftarrow{cl_i^l} = \overleftarrow{LSTM}\left(ul_i^l, h_{i+1}\right) \quad (25)$$

$$cl_i^{l\leftarrow l+1} = tanh\left(\overrightarrow{cl_i^l} + \overleftarrow{cl_i^l}\right) \quad (26)$$

$$ul_i^l = cl_i^l \qquad if \;\; l \leq 2 \quad (27)$$

$$cl_i = cl_i^l \qquad if \;\; l > 2 \quad (28)$$

For 2-stack BI-LSTM, the operations of equation (24) to (26) is executed for 2 times. Let l denotes the current stack which is initialized to 1. h_i denote the BI-LSTM hidden state for the ith utterance. $\overrightarrow{cl_i}$ and $\overleftarrow{cl_i}$ are the context vector for ul_i decoded by forward and backward LSTM, respectively. Finally, cl_i is the combination of $\overrightarrow{cl_i}$ and $\overleftarrow{cl_i}$. Since the number of BI-LSTM hidden units is 1024, the dimension of cl_i is [1,1024].

Output Layer. Finally, the model outputs nugget probability distribution for utterance $\boldsymbol{cl_i}$ by a softmax function as follows:

$$P(nugget|\mathrm{u_i}) = \frac{exp(W cl_i)}{\sum_{i'=1}^{k} exp(cl_{i'})} \quad (29)$$

where k denotes the number of utterances in a dialogue and $P(nugget|\mathrm{u_i})$ denotes the probability distribution of nugget for X_i. Dimension of the output is [1,7] since the number of nugget type is seven.

4.2. Experiments

Result of Nugget Detection (ND) Subtask. For hyper-parameters tuning of HCNN-LSTM, batch size is 30 and iterate 50 epochs for training process and set up early stropping to 3. We apply Adam optimizer with 1e-5 learning rate and the loss function is cross entropy. For HCNN-LSTM with skip-gram sentence representation, we apply 3-stack CNN in utterance layer with number of filters 256 for 1st stack, 512 for 2nd stack and 1024 for 3rd stack. For context layer we apply 2-stack BI-LSTM and the number of hidden units are both 1024 for 1st and 2nd BI-LSTM stack.

Two measures are used to evaluate ND tasks: Jensen-Shannon divergence (JSD) and Root Normalized Sum of Squared Errors (RNSS) as defined in [22]. As shown in Table 4, the inclusion of multi-stack CNN improves a little JSD with respect to baseline LSTM. However, RNSS is higher than BL-LSTM. For different sentence representation, BERT-LSTM outperform HCNN-LSTM. With the comparison between BERT-LSTM and BL-BERT, it shows that context layer is also important for ND subtask. Table 5 shows the ablation of BERT-LSTM for ND subtask. Multi-stack context layer well improves the performance since multi-stack structure can capture long-range context information, which is necessary for ND subtask. However, as Table 6, adding gating mechanism, or memory layer doesn't help the model to

improve the performance. It might be result from the insufficient training data that cause overfitting in complex models. The analysis of performance between different training data size is further discussed in section 4.3.

Table 4. Performance of ND subtask

Model	JSD	RNSS
BL-uniform	0.2304	0.3708
BL-popularity	0.1665	0.2653
BL-lstm	0.0248	0.0952
BL-BERT	0.0341	0.1171
HCNN-LSTM	0.0246	0.0962
BERT-LSTM	**0.0228**	**0.0933**

Table 5. Ablation of BERT-LSTM

Model	JSD	RNSS
BERT-LSTM	**0.0228**	**0.0933**
W/o CL multi-stack	0.0246	0.0951

Table 6. Experiments of gating and memory enhance

Model	JSD	RNSS
BERT-LSTM	**0.0228**	**0.0933**
W/ gating mechanism	0.0244	0.0960
W/ memory layer	0.0234	0.0941

4.3. Learning curve of different data size for ND subtask

Fig 7 shows the learning curve of BERT-LSTM model. Both JSD and RNSS reduce when adding number of training data until 100% training data are used. This tendency shows our model could perform better if there is more training data for ND subtask. On the other hand, we cannot expect a complex model to perform well without sufficient training data, this is the major reason that we do not apply gating mechanism and memory layer in HCNN-LSTM models for ND subtask.

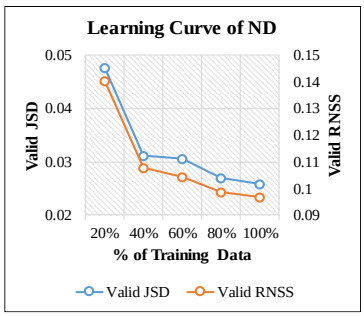

Fig 7. Learning Curve of ND

5. Conclusion

In this paper, we propose two hierarchical multi-stack models for both DQ and ND subtasks. The experiments show that multi-stack mechanism is effective in capturing long-range context information between words and utterances and improve the performance. In addition, gating mechanism and memory enhance structure is applied to MeHGCNN for DQ subtask, which improves the performance of all three types of score. Due to the insufficient training data of ND subtask, adding complex structure such as gating mechanism and memory enhance structure might cause overfitting and drop the performance. Moreover, besides word2vec algorithm with multi-stack CNN, we also try BERT as sentence representation which well improves the performance of all measures in both DQ and ND subtasks. Finally, our models outperform comparing with other baselines proposed by NTCIR.

6. References

1. Bird, S., Loper, E.: NLTK: The Natural Language Toolkit. Association for Computational Linguistics (2004)
2. Blunsom, P., Kalchbrenner, N.: Recurrent convolutional neural networks for discourse compositionality. Proceedings of the 2013 Workshop on Continuous Vector Space Models and their Compositionality (2013)
3. Chung, J., Gulcehre, C., Cho, K., Bengio, Y.: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. arXiv preprint arXiv:1412.3555 (2014)
4. Dauphin, Y, N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. arXiv preprint arXiv: 1612.08083 (2016)
5. Devlin, J., Chang, M, W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv: 1810.04805 (2018)
6. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation (1997)
7. Huang, Z., Xu, W., Yu, K.: Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
8. Jurafsky, D., Shriberg, L., Biasca, D.: Switchboard SWBD-DAMSL Shallow-Discourse-Function Annotation Coders Manual, Draft 13 (1997)
9. Kim, Y.: Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 conference on Empirical Methods in Natural Language Processing (2014)
10. Kumar, H., Agarwal, A., Dasgupta, R., Joshi, S., Kumar, A.: Dialogue Act Sequence Labeling using Hierarchical encoder with CRF. The Thirty-Second AAAI Conference on Artificial Intelligence (2018)
11. Lee, J, Y., Dernoncourt, F.: Sequential short-text classification with recurrent and convolutional neural networks. Proceedings of NAACL-HLT (2016)
12. Lendvai, P., Geertzen, J.: Token-based chunking of turn-internal dialogue act sequences. SIGDIAL Workshop on Discourse and Dialogue (2007)
13. Liu, F., Baldwin, T., Cohn, T.: Capturing Long-range Contextual Dependencies with Memory-enhanced Conditional Random Fields. Proceedings of the Eighth International Joint Conference on Natural Language Processing (2017)
14. Liu, Y., Han, K., Tan, Z., Lei, Y.: Using Context Information for Dialog Act Classification in DNN Framework. Conference on Empirical Methods in Natural Language Processing (2017)

15. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional lstm-cnns-crf. Association for Computational Linguistics (2016)
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (2013)
17. Sakai, T.: Comparing Two Binned Probability Distributions for Information Access Evaluation. The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (2018)
18. Shriberg, E., Dhillon, R., Bhagat, S., Ang, J., Carvey, H.: The ICSI meeting recorder dialog act (MRDA) corpus (2004)
19. Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Van, Ess-Dykema, C., Meteer, M.: Dialogue act modeling for automatic tagging and recognition of conversational speech. Association for Computational Linguistics (2000)
20. Vaswani, A., Shazeer, M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A, N., Kaiser, K., Polosukhin, I.: Attention Is All You Need. arXiv preprint arXiv: 1706.03762 (2017)
21. Weston, J., Chopra, S., Bordes, A.: Memory networks. Proceedings of the 3rd International Conference on Learning Representations (2015)
22. Zeng, Z., Kato, S., Sakai, T.: Overview of the NTCIR-14 Short Text Conversation Task: Dialogue Quality and Nugget Detection Subtasks. Proceedings of the 14th NTCIR Conference on Evaluation of Information Access Technologies (2019)
23. Zimmermann, M.: Joint segmentation and classification of dialog acts using conditional random fields. 10th Annual Conference of the International Speech Communication Association (2009)