WIDM @ NTCIR-14 STC-3 Task: Dialogue Quality and Nugget Detection for Short Text Conversation (STC-3) based on Hierarchical Multi-Stack Model with Memory Enhance Structure

NATIONAL CENTRAL UNIVERSITY, TAOYUAN, TAIWAN

AUTHORS: HSIANG-EN CHERNG AND CHIA-HUI CHANG

PRESENTER: HSIANG-EN CHERNG (SEAN)

Outline

1. Introduction
2. Dialogue Quality (DQ) Subtask
3. Nugget Detection (ND) Subtask
4. Conclusion

Introduction

Task Overview – DQ Subtask

Task Overview – ND Subtask

Contribution

Task Overview – DQ Subtask

Goal of DQ

- DQ aims to evaluate the quality of a dialogue by three measures (scale: -2, -1, 0, 1, 2)
  1) A-score: Task Accomplishment
  2) E-score: Dialogue Effectiveness
  3) S-score: Customer Satisfaction of the dialogue

Why DQ

- To build good task-oriented dialogue systems, we need good ways to evaluate them
- You cannot improve dialogue systems if you cannot measure, DQ provides 3 measures

Task Overview – ND Subtask

Goal of ND

- ND subtask aims to classify the nugget of utterances in a dialogue
- ND is similar to dialogue act (DA) labeling problem

Nugget: purpose or motivation

Why ND

- Nuggets may serve as useful features for automatically estimating dialogue quality
- ND may help us diagnose a dialogue closely (why it failed, where it failed)
- Experiences from ND may help us design effectively and efficiently helpdesk systems

Contribution

1. We proposed and compared several DNN models based on
   - Hierarchical multi-stack CNN for sentence and dialog representation
   - BERT for sentence representation
2. We compared the models with or without memory enhance
3. We compared simple BERT model with BERT + complex structures model
4. In both DQ and ND, our models result in the best performance comparing with organizer baseline models

BERT: An pre-train model based on multiple bi-directional transformer blocks (Devlin, J., Chang, M, W., Lee, K., Toutanova, K. 2018)

Dialogue Quality (DQ) Subtask

Model

Experiments

Memory enhanced multi-stack gated CNN (MeHGCNN)

Embedding layer

- 100 dimensions Word2Vec

Utterance layer

- 2-stack gated CNN learning sentence representation

Context layer

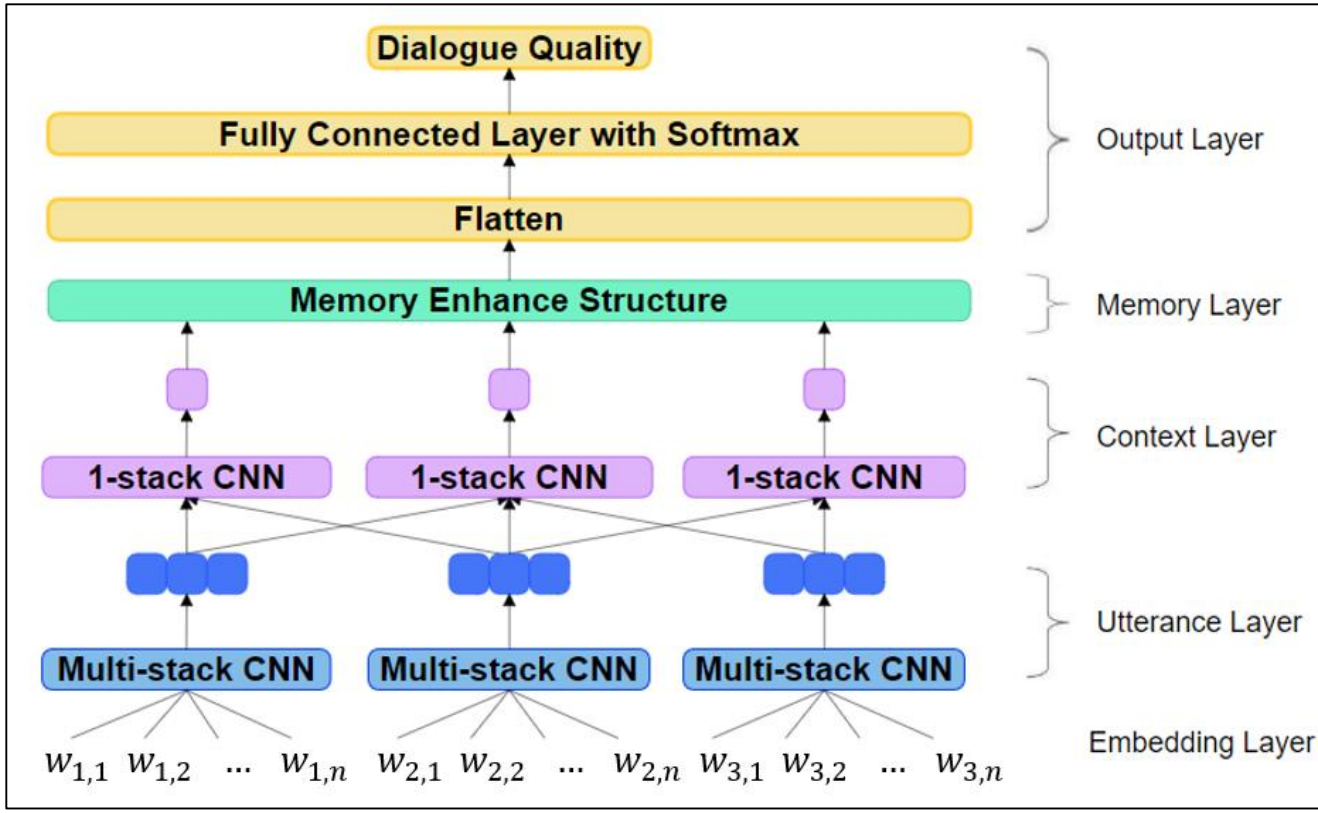
- 1-stack gated CNN learning context information

Memory layer (Memory Network)

- Further capture long-range context features

Output layer

- Output DQ distribution by softmax



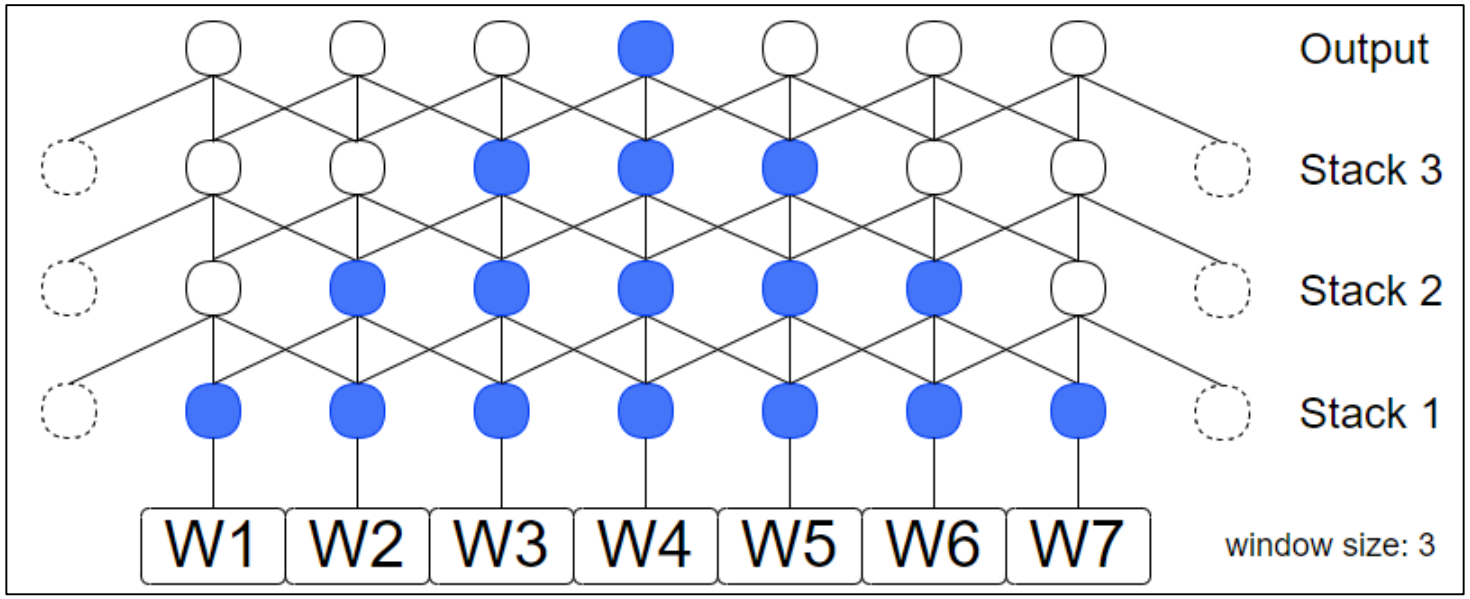
3 techniques we used in our models

1. Multi-stack structure
2. Gating mechanism
3. Memory enhance (memory network)

Multi-stack

Multi-stack structure

- Hierarchically capture rich n-gram information
- Window size k and # stacks m can capture m(k-1)+1 words features



Gating mechanism & Memory Enhance Structure

Gating mechanism

- Widely used in 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
- *Language modeling with gated convolutional networks (Dauphin, Y, N., Fan, A., Auli, M. 2016)*

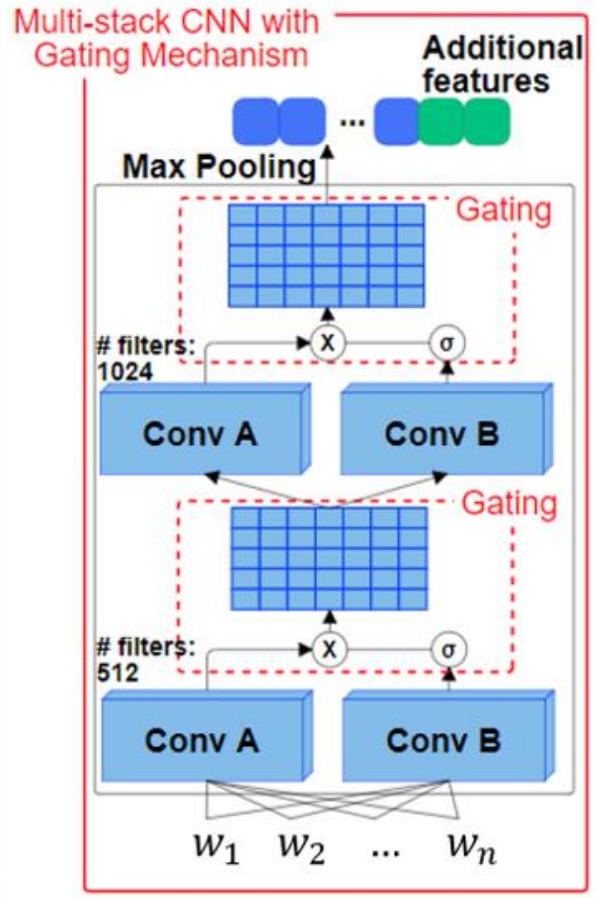
Memory enhance structure

- LSTM are not good at capturing very long-range context features
- Memory network is applied to our models to get detail context features by self-attention
- *Memory networks (Weston, J., Chopra, S., Bordes, A. 2015)*

Utterance Layer: 2-stack Gated CNN

- Utterance layer (UL)
  - $l = 1$
  - $\mathrm{X}_i^l = [w_{(i,1)}, w_{(i,2)}, \dots, w_{(i,n)}]$
  - $if\ l \le 2$
    - $ulA_i^l = ConvA(\mathrm{X}_i^l)$
    - $ulB_i^l = ConvB(\mathrm{X}_i^l)$
    - $ulC_i^l = ulA_i^l \odot \sigma(ulB_i^l)$
    - $\mathrm{X}_i^{l \leftarrow l+1} = ulC_i^l$
  - $ul_i = [maxpool(ulC_i^l), \overset{1x1}{speaker_i}, \overset{1x7}{nugget_i}]$

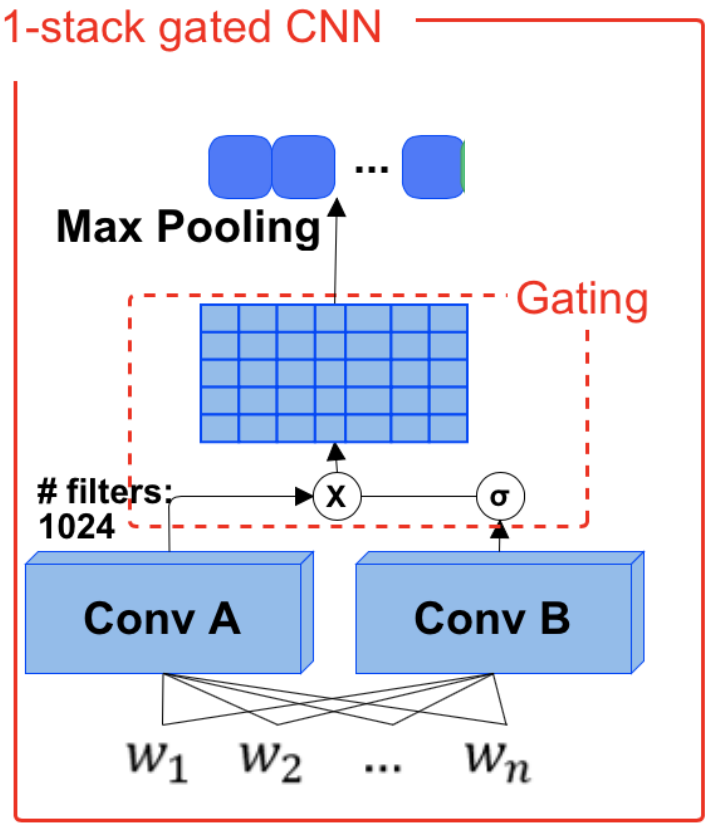
Apply max-pooling to the output of the last stack



Context Layer: 1-stack Gated CNN

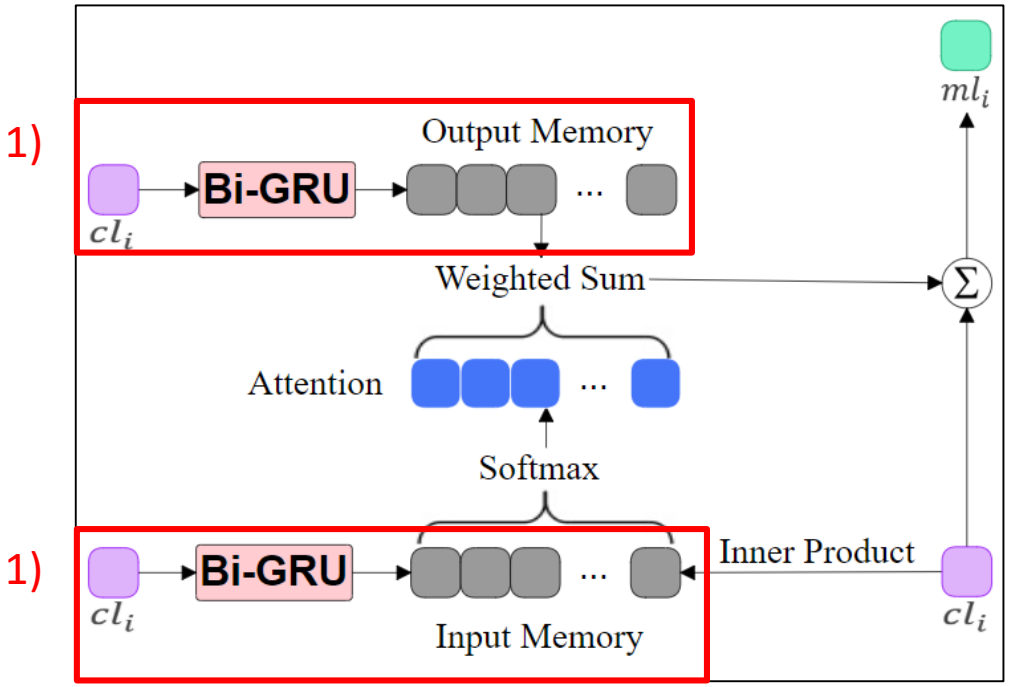
- Context layer (CL)
  - Conduct the same operations as UL but no additional features
  - $clA_i = ConvA([ul_{i-1}, ul_i, ul_{i+1}])$
  - $clB_i = ConvB([ul_{i-1}, ul_i, ul_{i+1}])$
  - $clC_i = clA_i \odot \sigma(clB_i)$
  - $cl_i = maxpool(clC_i)$

The output of context layer for utterance i is $\boldsymbol{cl_i}$



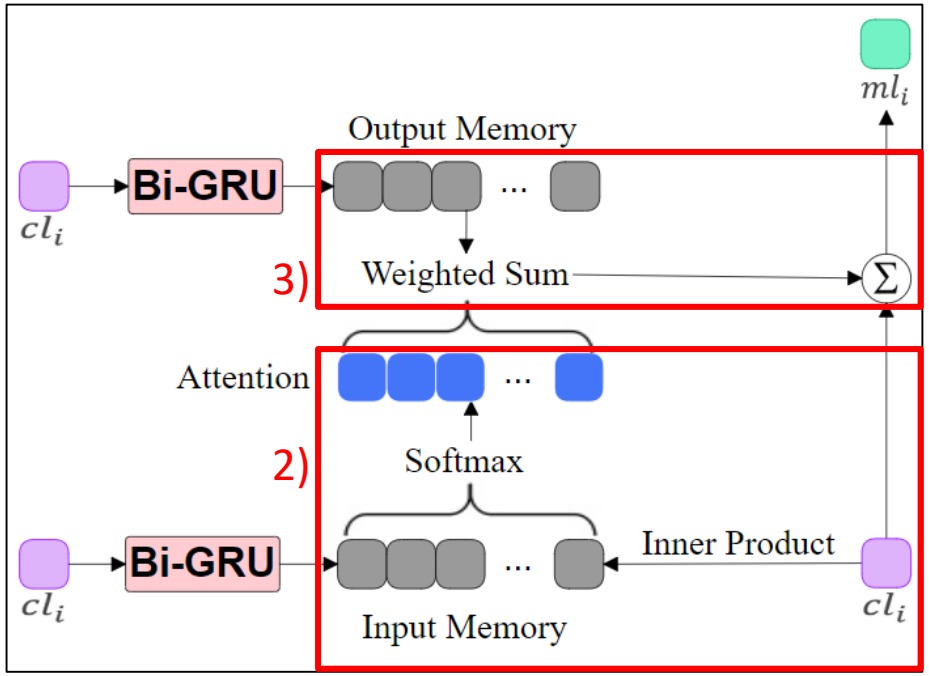
Memory Layer

- Memory layer (ML)
  1) Both input memory (I_i) and output memory (O_i) are generated by BI-GRU from cl_i
     - **Input Memory**
       - $\overrightarrow{I_i} = \overrightarrow{GRU}(cl_i, h_{i-1})$
       - $\overleftarrow{I_i} = \overleftarrow{GRU}(cl_i, h_{i+1})$
       - $I_i = tanh(\overrightarrow{I_i} + \overleftarrow{I_i})$
     - **Output Memory**
       - $\overrightarrow{O_i} = \overrightarrow{GRU}(cl_i, h_{i-1})$
       - $\overleftarrow{O_i} = \overleftarrow{GRU}(cl_i, h_{i+1})$
       - $O_i = tanh(\overrightarrow{O_i} + \overleftarrow{O_i})$



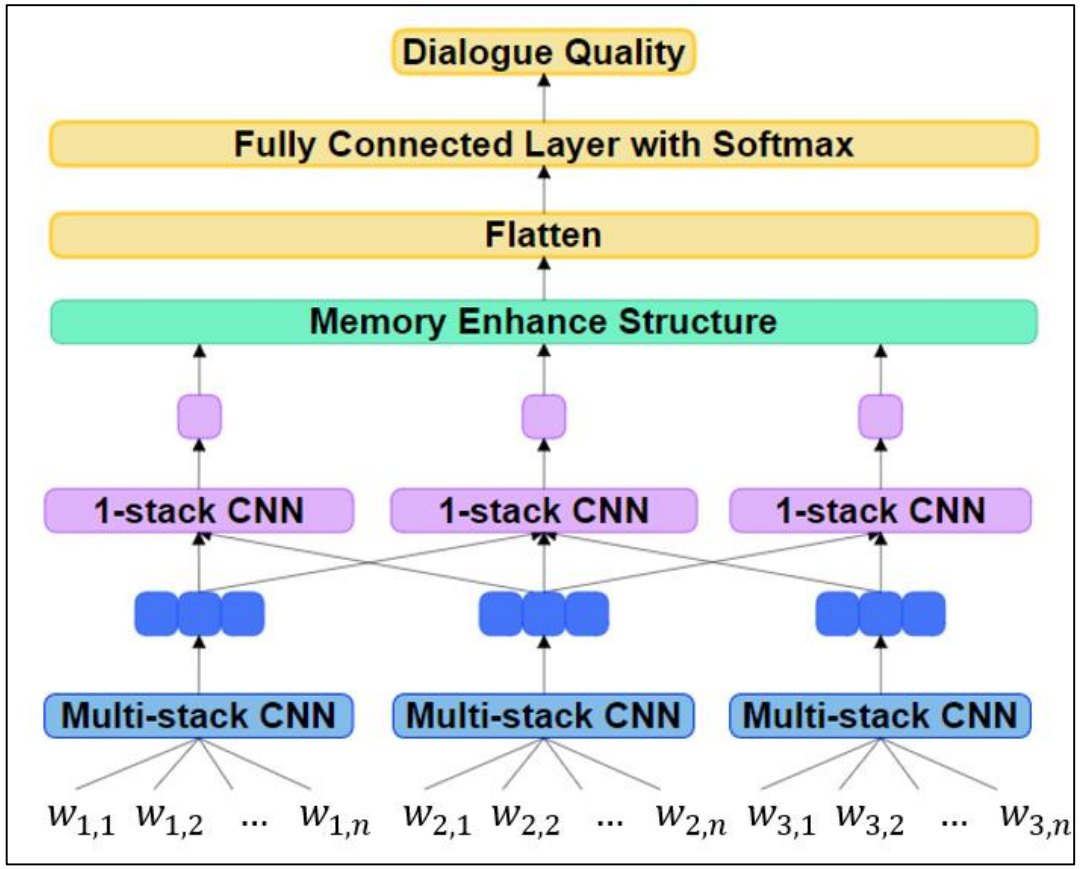
Memory Layer (cont.)

- Memory layer (ML)
  - 2) **Attention weight** is the inner product between cl_i and I_i followed by softmax
    - $w_i = \frac{exp(cl_i \cdot I_i)}{\sum_{i'=1}^{k} exp(cl_{i'} \cdot I_{i'})}$
  - 3) The output of memory layer for cl_i is the addition between **weighted sum of O_i and cl_i**
    - $ml_i = \sum_{i'=1}^{k} w_{i'} \cdot O_{i'} + cl_i$



Output Layer

- Output layer
  - Flatten all utterances vectors
    - $ml = [ml_1, ml_2, \dots, ml_k]$
  - Apply a fully-connected layer with softmax to output the score distribution as
    - $fc = mlW_{fc} + b_{fc}$
    - $P(score|dialogue) = \frac{exp(fc_i)}{\sum_{i'=1}^{5} exp(fc_{i'})}$
  - Dimension of $P(score|u_i)$ is 1x5 since the scale of scores are -2, -1, 0, 1, 2



Dialogue Quality (DQ) Subtask

Model

Experiments

Data

Customer helpdesk dialogues

- Annotators: 19 students from Waseda university
- Validation data is randomly selected 20% from training data

Data	Training	Testing
# Dialogues	1,672	390
# Utterances	8,672	1,755

Preprocessing

- Remove all full-shape characters
- Remove all half-shape characters except A-Za-z!"#$%&()*+,-./:;<=>?@[\]^_`{|}~ ‘
- Tokenize by NLTK toolkit (Edward Loper and Steven Bird. 2002)

Word Embedding

Embedding parameter

- Dimension: 100
- Tool: genism
- Method: skip-gram
- Window size: 5

STC-3 DQ&ND data

- Customer helpdesk dialogues
- Including train data and test data

Data source	# words
text8(wiki)	17,005,208
STC-3 DQ&ND	339,410
Total	17,344,618

Hyper parameters of DQ

Hyper parameters	Value
Batch size	40
Epochs	50
Early stopping	3
Optimizer	Adam optimizer
Learning rate	0.0005
Multi-stack CNN of UL	• # convolutional layers: 2 • # Filter: [512, 1024] • Kernel size: 2 & 2
Multi-stack CNN of CL	• # convolutional layers: 1 • # Filter: [1024]

Result of DQ Subtask

- **MeHGCNN**: Our proposed model
- **MeGCBERT**: Replace embedding and utterance layer of MeHGCNN with BERT
- **BL-BERT**: Simple BERT model with only BERT and output layer

	Model	(A-score)		(E-score)		(S-score)	
		NMD	RSNOD	NMD	RSNOD	NMD	RSNOD
Organizer baselines	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
Ours	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

Ablation of MeGCBERT for DQ

Gating mechanism & Memory enhance

- Well improve A-score & S-score
- A little improvement in E-score

Adding Nugget features

- Well improve A-score
- A little improvement in E-score

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 enhance	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

Nugget Detection (ND) Subtask

Model

Experiments

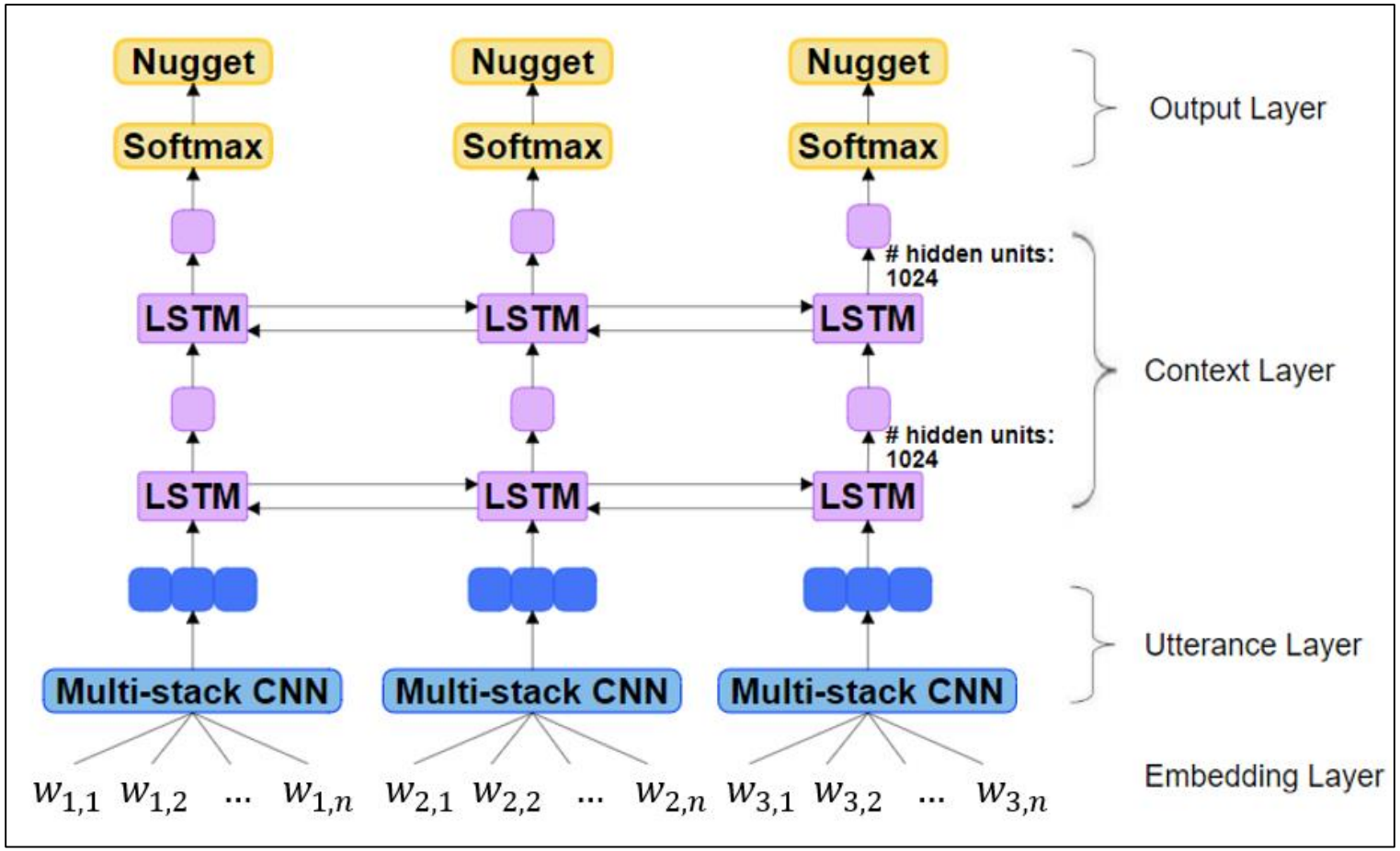
Hierarchical multi-stack CNN with LSTM (HCNN-LSTM)

Embedding layer
- 100 dimensions Word2Vec

Utterance layer
- Apply 3-stack CNN to learn sentence representation

Context layer
- Apply 2-stack BI-LSTM to learn context information between utterances

Output layer
- Output the nugget distribution by softmax



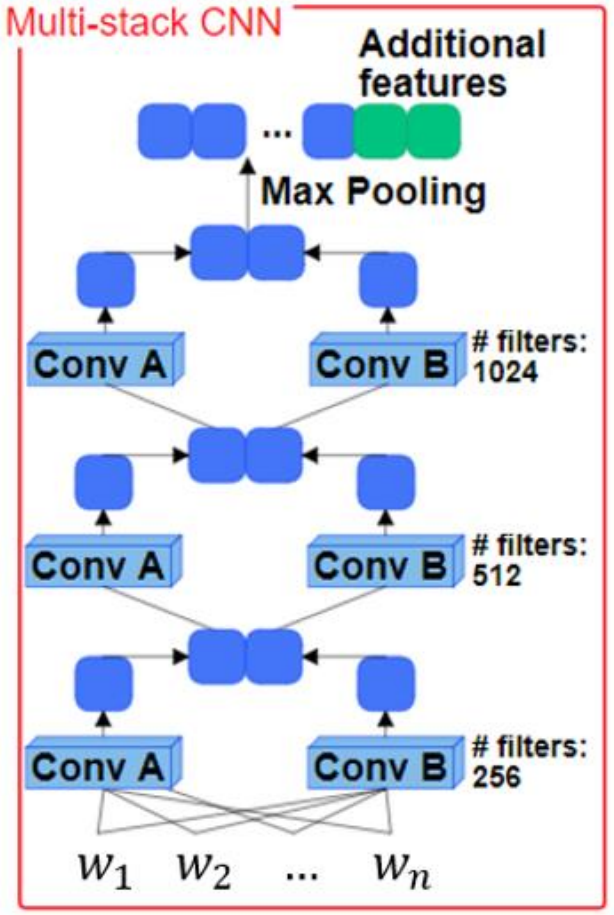
Utterance Layer: 3-stack CNN

- Utterance layer (UL)
  - $l = 1$
  - $\mathrm{X}_i^l = [w_{(i,1)}, w_{(i,2)}, \ldots, w_{(i,n)}]$
  - $ul\mathrm{A}_i^l = ConvA(\mathrm{X}_i^l)$
  - $ul\mathrm{B}_i^l = ConvB(\mathrm{X}_i^l)$
  - $ul\mathrm{C}_i^l = [ul\mathrm{A}_i^l, ul\mathrm{B}_i^l]$
  - $\mathrm{X}_i^{l \leftarrow l+1} = ul\mathrm{C}_i^l$
  - $ul_i = [maxpool(ul\mathrm{C}_i^l), speaker_i]$

$if\ l \leq 3$

1x1

Filter size: 2&3 for convA&convB

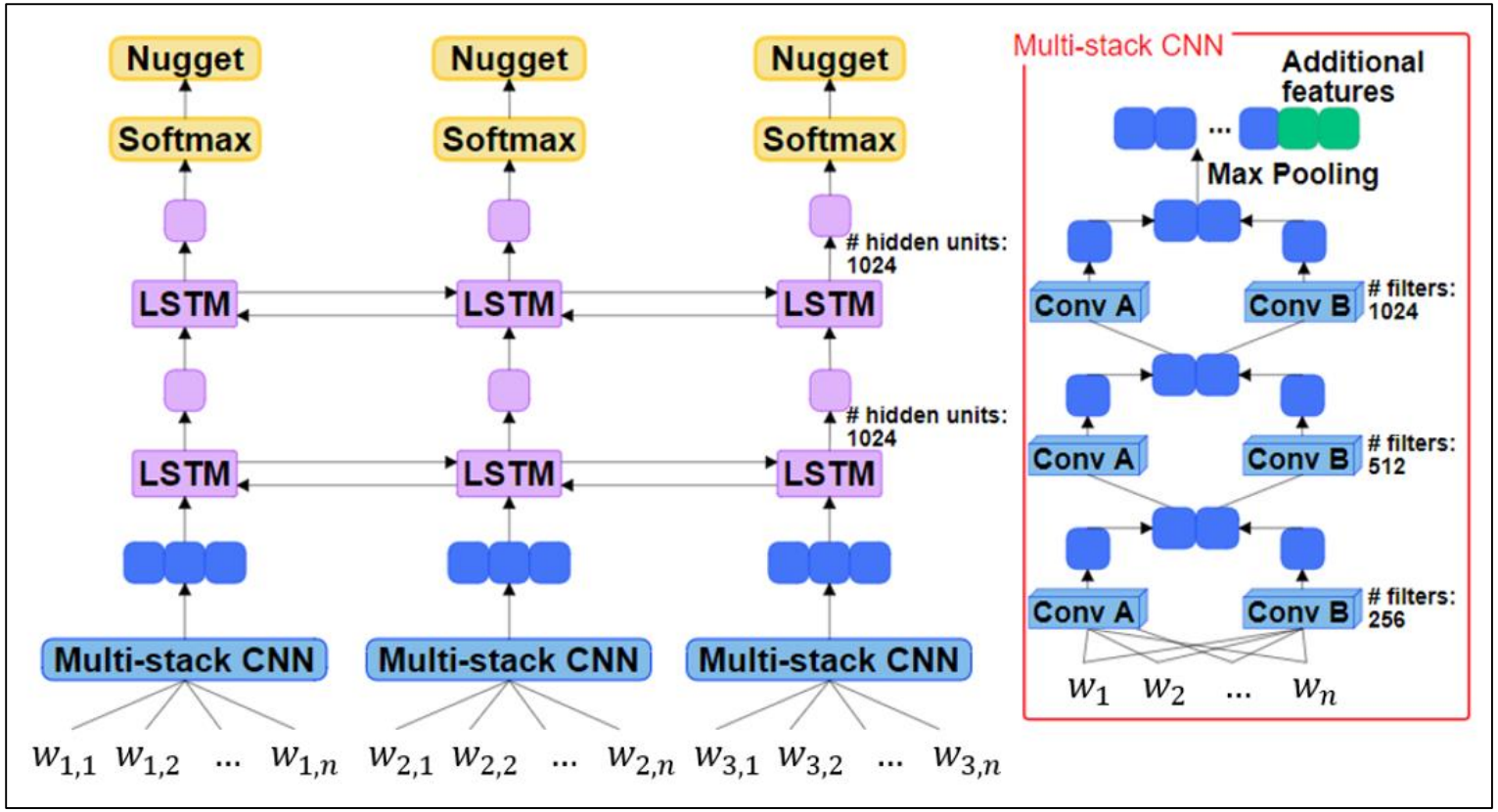


Context Layer: 2-stack BI-LSTM & Output Layer

- Context Layer (CL)
  - $\overrightarrow{cl_i^l} = \overrightarrow{LSTM}(ul_i, h_{i-1})$
  - $\overleftarrow{cl_i^l} = \overleftarrow{LSTM}(ul_i, h_{i+1})$
  - $cl_i^l = tanh\left(\overrightarrow{cl_i^l} + \overleftarrow{cl_i^l}\right)$
  - $ul_i = cl_i^l$
  - $cl_i = cl_i^l$

$if\ l \leq 2$

- Output layer
  - $P(nugget|u_i) = \frac{exp(Wcl_i)}{\sum_{i'=1}^{k} exp(cl_{i'})}$



Nugget Detection (ND) Subtask

Model

Experiments

Hyper parameters of ND

Hyper parameters	Value
Batch size	30
Epochs	50
Early stopping	3
Optimizer	Adam optimizer
Learning rate	0.0005
Multi-stack CNN	• # convolutional layers: 3 • # Filter: [256, 512, 1024] • Kernel size: 2 & 3
Multi-stack BI-LSTM	• # BI-LSTM layers: 2 • # hidden units: [1024, 1024] • Activation function of concatenation: tanh

Result of ND Subtask

- **HCNN-LSTM**: Our proposed model
- **BERT-LSTM**: Replace the embedding layer and utterance layer of HCNN-LSTM with BERT
- **BL-BERT**: Simple BERT + Output layer model

- BERT-LSTM outperforms all other models
- HCNN-LSTM outperforms NTCIR baselines in JSD
- Context layer is important for BERT
  - JSD drop 0.012 and RNSS drop 0.024 without context layer

Organizer baselines

Ours

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**

Ablation & Complex Structure Experiments

Left table shows that both UL and CL are important for ND subtask

Right table shows that both gating mechanism and memory enhance structure doesn't improve the performance

- Since the less training data, complex structure might cause overfitting

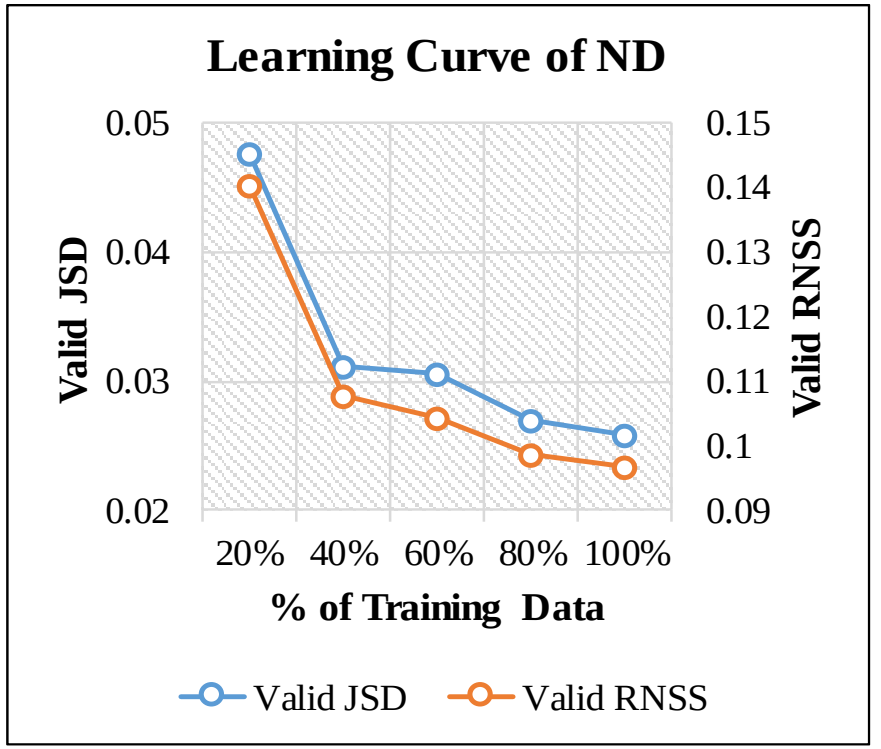
Model	JSD	RNSS
BERT-LSTM	**0.0228**	**0.0933**
W/o CL multi-stack	0.0246	0.0951

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

Learning Curve of Different Training Data Size for ND

For ND subtask

- Both JSD and RNSS reduce when adding % of training data until 100%
- The tendency shows our model could perform better if there is more training data
- We do not apply a complex model for ND since the lack of training data



Conclusion

Conclusion

1. We propose two hierarchical models for DQ and ND subtasks
2. We compare the models w/ & w/o gating mechanism & memory enhance
   - Both improve the performance of DQ subtask
   - But drop the performance of ND subtask
3. Data for ND might be insufficient which cause overfitting in complex models
4. We compare sentence representation between BERT and word2vec
5. Our models outperform other organizer baseline models in ND & DQ subtasks

Q&A

Nugget Types for ND

CNUG0: Customer trigger

- Problem stated

CNUG*: Customer goal

- Solution confirmed

CNUG: Customer regular

- Contains info that leads to solution

CNaN: Customer Not-a-Nugget

- Does not contain info that leads to solution

HNUG*: Helpdesk goal

- Solution stated

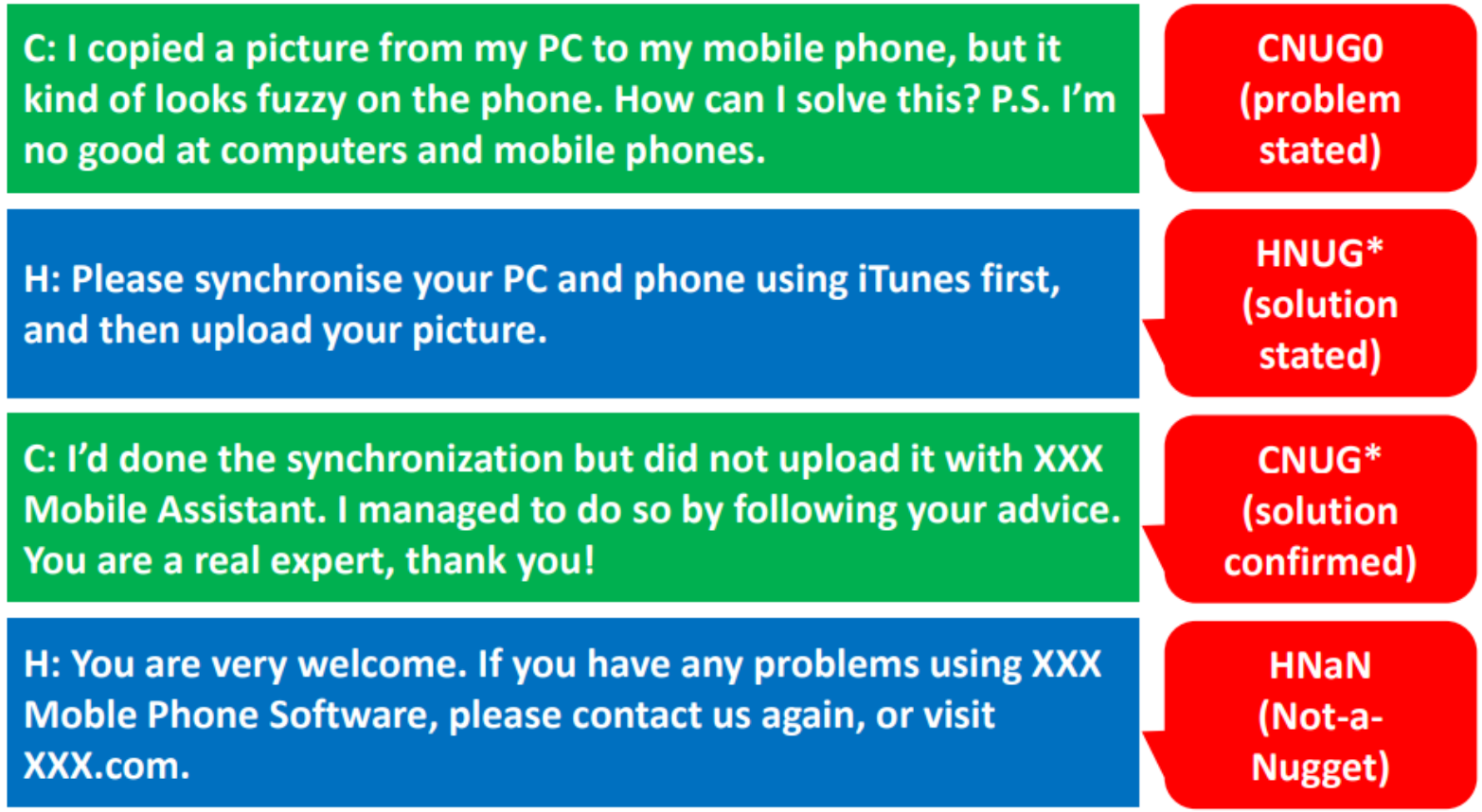
HNUG: Helpdesk regular

- contains info that leads to solution

HNaN: Helpdesk Not-a-Nugget

- Does not contain info that leads to solution

Example of ND



Measures of DQ

A-score: Task Accomplishment

- Has the problem been solved? To what extent?

E-score: Dialogue Effectiveness

- Do the utterers interact effectively to solve the problem efficiently?

S-score: Customer Satisfaction of the dialogue

- Not of the product/service or the company

Scale: -2, -1, 0, 1, 2

Related Work

Short Text Conversation (STC)

Word Embedding to BERT

Short Text Conversation (STC)

Traditional mechine learning methods

- Hidden Markov Model (Stolcke et al. 2006)
- Naïve Bayes (Lendvai and Geertzen 2007)

Deep learning methods

- CNN based & RNN based models (Lee, J, Y., Dernoncourt, F. 2016)
- Recurrent convolutional neural networks (Blunsom, P., Kalchbrenner, N. 2013)
- LSTM + CRF model (Huang, Z., Xu, W., Yu, K. 2015; Ma, X., Hovy, E. 2016)
- Hierarchical CNN + CNN / Bi-LSTM (Liu, Y., Han, K., Tan, Z., Lei, Y. 2017)
- Hierarchical encoder with CRF (Kumar, H., Agarwal, A., Dasgupta, R., Joshi, S., Kumar, A. 2018)

Word Embedding to BERT

Word embedding

- Word2Vec (Mikolov, T., Chen, K., Corrado, G., Dean, J. 2013)
- Our proposed models apply word2vec with skip-gram algorithm

BERT (**B**idirectional **E**ncoder **R**epresentations from **T**ransformers)

- An pre-train model based on multiple bi-directional transformer blocks
- Redefines the state of the art for 11 natural language processing tasks
- BERT (Devlin, J., Chang, M, W., Lee, K., Toutanova, K. 2018)

Transformer

- Constructed by **self-attention** and **feed-forward neural networks** (without any CNN, RNN)
- Attention is all you need (Vaswani, A., Shazeer, M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A, N., Kaiser, K., Polosukhin, I. 2017)

Two Evaluation Measures of DQ

Normalized Match Distance (NMD)

Cross-Bins Measures

Definition

- There are 2 normalized distributions (p, q)
- $cp(i) = \sum_{k=0}^{i} p(k)$
- $cq(i) = \sum_{k=0}^{i} q(k)$
- $MD(p, q) = \sum_i |cp(i) - cq(i)|$
- $NMD(p, q) = \frac{MD(p,q)}{length-1}$

Example

- $p = [0,0,1]$
- $q_1 = [0.2,0.8,0], q_2 = [0.8,0.2,0]$

- $cp = [0,0,1]$
- $cq_1 = [0.2,1,1], cq_2 = [0.8,1,1]$

- $NMD(p, q_1) = \frac{0.2+1+0}{3-1} = \mathbf{0.6}$
- $NMD(p, q_2) = \frac{0.8+1+0}{3-1} = \mathbf{0.9}$
- q_1 is better than q_2

Two Evaluation Measures of DQ (cont.)

Root Symmetric Normalized Order-Aware Divergence (RSNOD)

Cross-Bins Measures consider the distance between a pair of bins

Definition

- There are 2 normalized distributions (p, q)
- $DW(i) = \sum_j |i-j|(p(j)-q(j))^2$
- $OD(p,q) = \frac{1}{|B^*|}\sum_{i \in B^*} DW(i)\,,\ \boldsymbol{B^* = \{i | p(i) > 0\}}$
- $SymmetricOD(p,q) = \frac{OD(p,q)+OD(q,p)}{2}$
- $RSNOD(p,q) = \sqrt{\frac{SymmetricOD(p,q)}{length-1}}$

DW: Distance-Weighted sum of squares
OD: Order-Aware Divergence

Two Evaluation Measures of ND

Jensen-Shannon divergence (JSD)

Evaluate the similarity between 2 normalized distributions

Definition

- If there are 2 normalized distributions (p, q)
- Define $m = \frac{1}{2}[p(+)q]$ (element-wise addition)
- $JSD = \frac{1}{2}[KL(p, m) + KL(q, m)]$ with log base = 2
- $0 \leq JSD \leq 1$

The lower JSD means the similar distributions

Two Evaluation Measures of ND (cont.)

Root Normalized Sum of Squared Errors (RNSS)

Evaluate the similarity between 2 normalized distributions

Definition

- If there are 2 normalized distributions (p, q)
- $RNSS = \sqrt{\frac{\sum_i (p_i - q_i)^2}{2}}$
- $0 \leq RNSS \leq 1$

The lower $RNSS$ means the similar distributions

ND as a traditional sequence labeling problem

ND subtasks take label probability distribution as label

- We could only apply softmax layer instead of CRF layer

We consider the ND subtask as a traditional sequence labeling problem

- Convert the label distribution to one-hot labeling
- Solve the ND subtask by CRF instead of softmax
- Evaluate the performance by precision / recall / f1-score

Preprocessing

Distribution labels -> one-hot labels

- Choice the nugget with highest probability as label

For labels with 2 highest probability nuggets

- Create 2 one-hot labels for both nuggets as golden answers

Nugget	CNUG*	CNUG	CNaN	CNUG0	HNUG*	HNUG	HNaN
Original Label	0.158	**0.421**	**0.421**	0	0	0	0

One-hot Label 1	0	**1**	0	0	0	0	0
One-hot Label 2	0	0	**1**	0	0	0	0

ND as sequence labeling Performance

HCNN-BERT outperform HCNN-skipGram in accuracy, macro P and macro F

Accuracy is much more higher than macro P/R/F

- Some nugget types are difficult to correctly recognized

Model	Accuracy	Macro P	Macro R	Macro F
HCNN-skipGram	88.8%	75.6%	**74.8%**	75.2%
HCNN-BERT	**89.9%**	**83.4%**	74.6%	**78.7%**

Confusion Matrix

Rows: prediction / Columns: Label

Nugget pairs that are easily confused

- [CNUG, CNaN]
- [CNUG*, CNUG]
- [CNUG*, CNaN]
- [HNUG*, HNUG]
- [HNUG, HNaN]

Nugget	CNUG*	CNUG	CNaN	CNUG0	HNUG*	HNUG	HNaN
CNUG*	19	16	10	0	0	0	0
CNUG	9	431	43	1	0	0	0
CNaN	3	23	57	0	0	0	0
CNUG0	0	0	12	374	0	0	0
HNUG*	0	0	0	0	27	14	2
HNUG	0	0	0	0	17	619	31
HNaN	0	0	0	0	0	21	70

➢ We doubt that whether these pairs are also confused by human

Confusion of Human Annotation

The table shows the avg probability difference of 2 highest nugget of utterances

- The higher difference means the higher probability to confused by human

The easily confused nugget pairs of models

- [CNUG, CNaN]
- [CNUG*, CNUG]
- [CNUG*, CNaN]
- [HNUG*, HNUG]
- [HNUG, HNaN]

Are also with confused by human

Nugget pair	Avg prob diff	# Pairs	Pct %
CNUG0, CNUG*	0.842	13	0%
CNUG0, CNUG	0.731	242	3%
CNUG0, CNaN	0.696	1,508	22%
CNUG*, CNUG	0.348	232	3%
CNUG*, CNaN	0.339	36	1%
CNUG, CNaN	0.455	1,793	26%
HNUG*, HNUG	0.307	865	13%
HNUG*, HNaN	0.118	8	0%
HNUG, HNaN	0.401	2,220	32%