Dialogue Topic Segmentation via Parallel Extraction Network with Neighbor Smoothing

Jinxiong Xia*
School of Software and Microelectronics, Peking University
Beijing, China
xiajx98@gmail.com

Cao Liu†
Meituan
Beijing, China
liucao@meituan.com

Jiansong Chen
Meituan
Beijing, China
chenjiansong@meituan.com

Yuchen Li
Meituan
Beijing, China
liyunchen04@meituan.com

Fan Yang
Meituan
Beijing, China
yangfan79@meituan.com

Xunliang Cai
Meituan
Beijing, China
caixunliang@meituan.com

Guanglu Wan
Meituan
Beijing, China
wanguanglu@meituan.com

Houfeng Wang†
MOE Key Lab of Computational Linguistics, Peking University
Beijing, China
wanghf@pku.edu.cn

ABSTRACT

Dialogue topic segmentation is a challenging task in which dialogues are split into segments with pre-defined topics. Existing works on topic segmentation adopt a two-stage paradigm, including text segmentation and segment labeling. However, such methods tend to focus on the local context in segmentation, and the inter-segment dependency is not well captured. Besides, the ambiguity and labeling noise in dialogue segment bounds bring further challenges to existing models. In this work, we propose the **P**arallel **E**xtraction **N**etwork with **N**eighbor **S**moothing (**PEN-NS**) to address the above issues. Specifically, we propose the parallel extraction network to perform segment extractions, optimizing the bipartite matching cost of segments to capture inter-segment dependency. Furthermore, we propose neighbor smoothing to handle the segment-bound noise and ambiguity. Experiments on a dialogue-based and a document-based topic segmentation dataset show that PEN-NS outperforms state-the-of-art models significantly.

CCS CONCEPTS

• **Computing methodologies** → **Discourse, dialogue and pragmatics**; *Information extraction.*

KEYWORDS

Dialogue topic segmentation; parallel extraction; boundary ambiguity; data noise; neighbor smoothing.

ACM Reference Format:
Jinxiong Xia, Cao Liu, Jiansong Chen, Yuchen Li, Fan Yang, Xunliang Cai, Guanglu Wan, and Houfeng Wang. 2022. Dialogue Topic Segmentation via Parallel Extraction Network with Neighbor Smoothing. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3477495.3531817

1 INTRODUCTION

Dialogues are dynamic flows with changing topics [16, 20, 31, 33]. In some scenarios, such as customer service, we can summarize the dialogue sections in various positions into some pre-defined topics. For example, as illustrated in Table 1, the agent may first open up a conversation with a customer, followed by the customer stating his problems (*Problem Statement*). Then the agent confirms the order information with the customer (*Order Confirmation*). After some time, the agent provides the solution to the customer, and they reach a consensus (*Solution*). At last, the conversation ends with closing remarks (*Closing*). Since the conversations are often lengthy, identifying the topics and their positions can provide efficient data for manual and automatic evaluation of the quality of customer service, which is of great commercial value. Besides, it can benefit a variety of downstream tasks such as dialogue summarization [13, 21, 33], question answering [29], information retrieval [32], and dialogue modeling [28, 30].

Existing works [1, 3, 14, 18, 26] on topic segmentation adopt a two-stage paradigm by text segmentation and segment labeling. Arnold et al. [1] proposed SECTOR, which performs topic classification for each sentence and then merges the sentences into segments based on sentence embedding deviations obtained by topic classification. Barrow et al. [3] proposed S-LSTM, which first performs

*The work was done while the author was an intern at Meituan.
†Corresponding authors.

SIGIR '22, July 11–15, 2022, Madrid, Spain

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3531817

Table 1: A customer service conversation, which has been desensitized, is split into multiple segments with topics.

Speaker	Utterance Text	Topic
Agent	Glad to serve you. May I help you?	*Problem Statement*
Customer	I ordered the xxx from the platform, ...	*Problem Statement*
Agent	Uh huh.	*Problem Statement*
...	...	...
Agent	Your order was from xxx, right?	*Order Confirmation*
Customer	Yes.	*Order Confirmation*
...	...	...
Agent	Well, I'll help you complaint about ...	*Solution*
Customer	OK.	*Solution*
...	...	...
Agent	Madam, the platform will send an ...	*Closing*
Customer	OK.	*Closing*
Agent	I wish you a happy life!	*Closing*

segmentation by sentence-level sequence labeling and then labels each segment by pooled BiLSTMs [7]. In addition, Barrow et al. proposed an exploration strategy to recognize the topics of false-boundary segments to allow error recovery. Lo et al. [14] proposed Transformer^2, which leverages the knowledge of the pre-trained language models (PLMs) to segment documents by sentence-level sequence labeling, and multitasks by segmentation and classification. It has achieved state-of-the-art performance in segmentation.

However, some issues in the models mentioned above cannot be ignored. First, the separation of segmentation and classification may lead to error propagation. Second, segmentation methods based on sequence labeling [3, 10, 14, 26] or sentence embedding deviations [1, 23] focus on the local context. Because they have not explicitly modeled the segment-level representations, the topic information in the segments cannot be fully leveraged to help identify the segmentation boundaries. Third, the inter-segment dependency, crucial for a globally optimal solution, is not established. Fourth, the ambiguity and noise of segment bounds in informal dialogues [8, 15, 27] bring further challenges to existing models. For example, the utterance "The refund will arrive in an hour, and what else can I do for you?" in a customer service conversation can either be the end of a segment with the *"Solution"* topic or the start of a segment with the *"Closing"* topic. Moreover, because of the segment boundaries, which are ambiguous and sometimes hard to identify, annotations are prone to noise. An investigation into a real-world conversational dataset shows that false segment bounds exist in 28.7% of the data, which may lead to significant performance drops without carefully handling the noise in real-world applications.

In this work, we address the above issues by proposing PEN-NS: **P**arallel **E**xtraction **N**etwork with **N**eighbor **S**moothing. First, we handle dialogue topic segmentation by segment extraction and propose the Parallel Extraction Network (PEN) to extract the segments for each topic in parallel, thus avoiding the separation of segmentation and classification and the tendency to fall into local optimums. Second, to capture the inter-segment dependency, PEN minimizes the bipartite matching cost of segment representations to the labels in the training stage. Third, we propose Neighbor Smoothing (NS), a label smoothing [19] variant, to handle the ambiguity and noise of segment bounds, which are common in informal dialogues and real-world datasets. NS smooths the labels by giving the neighboring segments some importance to improve the stability and robustness of PEN. Lastly, we conduct experiments on a dialogue-based and a document-based topic segmentation dataset. The results show that PEN-NS outperforms the current state-the-the-art models significantly by at least 3.2% in F_1.

To sum up, our contributions are three-fold:

(1) We view dialogue topic segmentation as an IR task and propose the parallel extraction network to extract the segments in parallel, capturing the inter-segment dependency by minimizing the bipartite matching cost.

(2) We propose neighbor smoothing, a label smoothing variant to handle the ambiguity and noise of segment bounds when applying the parallel extraction network.

(3) We conduct experiments on a dialogue-based and a document-based dataset, and PEN-NS outperforms the current state-of-the-art significantly.

2 TASK FORMULATION

Before introducing PEN-NS, we give the formal definition of dialogue topic segmentation. Let $C = ((u_1, r_1), (u_2, r_2), ..., (u_n, r_n))$ be a conversation with n utterances, where u_i is the text of the i-th utterance and $r_i \in \{1, 2, ..., R\}$ is the role of the corresponding speaker. Let $segment(i, j)$ denote the segment which starts from the i-th utterance and ends at the j-th utterance. The task is to split C into multiple contiguous segments $(s_1, y_1), (s_2, y_2), ..., (s_m, y_m)$ where $s_i = segment(start_i, end_i)$ has a topic label $y_i \in \{1, 2, ..., K\}$, and K is the number of topics.

3 METHOD

As illustrated in Figure 1, PEN is the backbone of our model in combining with NS. PEN has four major components: (1) a hierarchical utterance encoder, where the utterances are encoded into vectors by a two-level encoder, (2) an attentive segment encoder, where the segments are encoded with the attention mechanism and their representations are in parallel. (3) parallel extraction, where segment extractions are performed parallelly followed by conflict resolution, and (4) bipartite matching optimization, where the segment representations are trained to match the labels with the minimum matching cost. Neighbor smoothing transforms the one-hot encoding labels into soft labels, assigning some weights to the neighbors of ground truth segments to deal with the ambiguity and noise in segment boundaries. We next describe PEN-NS in detail.

3.1 Hierarchical Utterance Encoder

The first part of PEN-NS is a hierarchical encoder to encode the utterances. Our model is agnostic to the specific encoder, and here we choose the BiLSTM [6], which is capable of capturing time-related bidirectional dependencies both in short terms and in long terms. Formally, we use R BiLSTMs to encode the speakers' utterances of R roles, respectively. Next, we concatenate the first output, the final output, the max pooling, and the mean pooling over the output of each BiLSTM followed by an MLP to form the utterance representations. To capture dialogue-level context, we feed these representations, plus a learnable vector, into another BiLSTM, the second level of the hierarchical encoder. Without the loss of generality, the output is denoted by $U' \in \mathbb{R}^{n \times d_m}$, representing n utterance with d_m dimensions. The n-th utterance is from the learnable vector, indicating the "NULL" utterance.

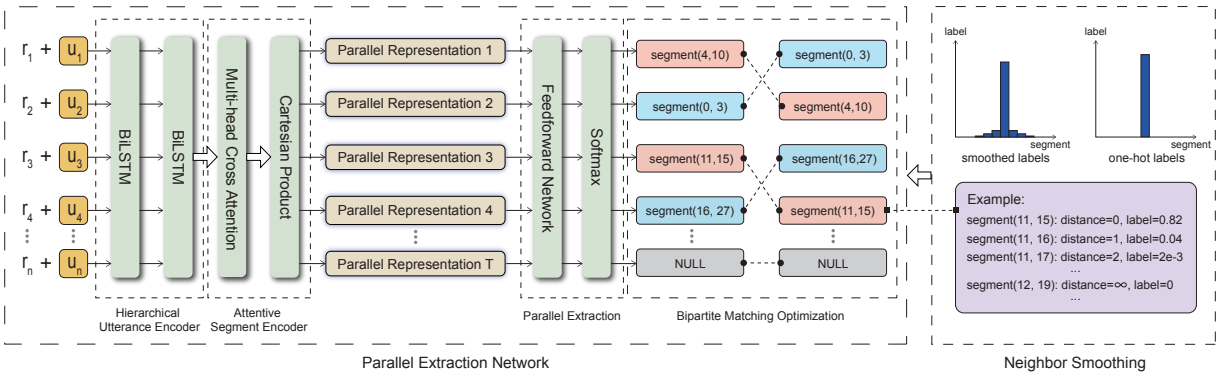

Figure 1: The overall architecture of PEN-NS. We address topic segmentation by segment extraction. For each pre-defined topic, we use the PEN to extract all the segments of that topic in the dialogue. Once the target segments are extracted, we match them to the segment labels with bipartite matching optimization. We use soft labels by neighbor smoothing in the training stage.

3.2 Attentive Segment Encoder

Following the hierarchical utterance encoder, we form the parallel segment representations utilizing the attention mechanism in the attentive segment encoder. We first employ T learnable vectors as the queries to perform multi-head attention [25] with U'. Here, T is a hyperparameter chosen significantly larger than a single topic's average number of segments, which ensures that the number of decoded segments is always greater than or equal to the ground truth segments. The output of the multi-head attention is denoted by $A \in \mathbb{R}^{T\times d_m}$. Next, we add U' and A with the broadcasting mechanism, the output denoted by $V \in \mathbb{R}^{T\times n\times d_m}$, which are the parallel representations of each utterance. Finally, we apply the cartesian product to V and V itself to get $H \in \mathbb{R}^{T\times n\times n\times d_m}$ as the parallel segment representations.

3.3 Parallel Extraction

Given the parallel representations of each segment, we decode them into segment positions by passing H into a feedforward network followed by a softmax layer. Let $P \in \mathbb{R}^{T\times n\times n\times K}$ denote the output of the softmax layer. Therefore, the parallel extractions can be performed as follows:

$$start^{pred}_{t,k}, end^{pred}_{t,k} = \arg\max_{i,j} P_{t,i,j,k} \tag{1}$$

where $start^{pred}_{t,k}$ and $end^{pred}_{t,k}$ denote the t-th segment's predicted start and end positions for the k-th topic, respectively. If some predicted segments share overlapping positions, we resolve the conflict by reserving the one with the highest extracted probability and discarding the others. For the intervals between the segments extracted with Eq. 1, we use the following formula to predict their topics:

$$p^{topic}_{i,j,k} = \frac{\max_t P_{t,i,j,k}}{\sum_{l=1}^{K} \max_t P_{t,i,j,l}} \tag{2}$$

where $p^{topic}_{i,j,k}$ denotes the score of the k-th topic for $segment(i, j)$.

3.4 Neighbor Smoothing

Assuming the t-th segment for topic k is $segment(start_{t,k}, end_{t,k})$, then the one-hot encoding labels are as follows:

$$y_{t,i,j,k} = \begin{cases} 1, & \text{if } i = start_{t,k} \text{ and } j = end_{t,k} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

In this work, we propose neighbor smoothing to smooth the labels among the neighbors of $segment(start_{t,k}, end_{t,k})$. The "neighbors" of a segment are those whose start and end positions are close to the segment. So neighbor smoothing can be formalized as follows:

$$\hat{y}_{t,i,j,k} = \frac{e^{-\lambda \mathcal{D}([i,j],[start_{t,k},end_{t,k}])}}{\sum_{p=1}^{n}\sum_{q=p}^{n} e^{-\lambda \mathcal{D}([p,q],[start_{t,k},end_{t,k}])}} (i \le j) \tag{4}$$

where λ is a hyperparameter, and $\mathcal{D}$ is the distance metric for segment coordinates. When $i > j$, $\hat{y}_{t,i,j,k}$ is set to 0 since it does not represent a valid segment. Neighbor smoothing is agnostic to the specific distance measure, and in implementation, we choose the Manhattan distance for $\mathcal{D}$, but limit the valid *"neighbors"* to the segments which share at least one correct boundary, in order to enhance boundary identification:

$$\mathcal{D}([a,b],[c,d]) = \begin{cases} |a-c| + |b-d|, & \text{if } a = c \text{ or } b = d \\ \infty, & \text{otherwise} \end{cases} \tag{5}$$

3.5 Training

At the training stage, we match the predicted extraction scores $P_{t,i,j,k}$ to the soft labels $\hat{y}_{t,i,j,k}$ to compute the loss. We first compute the loss for each topic and then add them up for the final loss. Assume there are n_k segments for topic k. If $n_k < T$, we pad it with $(T - n_k)$ NULL segments denoted by $segment(n, n)$. Then a permutation $\pi^k \in \Pi(T)$ is assigned to the T segments to compute the negative log-likelihood loss:

$$\mathcal{L}^k(\pi^k) = \sum_{t=1}^{T}\sum_{i,j=1}^{n} -\hat{y}_{\pi^k(t),i,j,k} \log P_{\pi^k(t),i,j,k} \tag{6}$$

Table 2: The statistics of CSTS and WikiSection, where the total number of conversations/documents, the average number of turns/sentences per conversation/document, the average number of segments per conversation/document, and the number of topics are reported.

Dataset	CSTS	WikiSection
total convs/docs	562	19.5k
avr turns/sents per conv/doc	40.2	56.5
avr segs per conv/doc	5.4	8.3
topics	9	30

We optimize the permutation by minimizing the loss:

$$\hat{\pi}^k = \arg\min_{\pi \in \Pi(T)} \mathcal{L}^k(\pi^k) \tag{7}$$

So it becomes a minimum-cost bipartite matching problem, which the Hungarian Algorithm [11] can efficiently solve in polynomial time. After finding the optimal matching, we can add up the losses from each topic to compute the final loss $\mathcal{L}$:

$$\mathcal{L} = \sum_{k=1}^{K} \mathcal{L}^k(\hat{\pi}^k) \tag{8}$$

4 EXPERIMENTS

4.1 Setup

Datasets We manually annotated a dataset named *"Customer-Service Topic Segmentation" (CSTS)* from real-world customer service to support our research. It consists of 562 conversations, each consisting of up to 380 utterances. We defined nine different topics, based on which we employed three annotators to segment the conversations. The dataset was randomly split into train/dev/test by a proportion (9:0.5:0.5).

We also experiment with *WikiSection*[1], which was constructed from wikipedia pages [1]. The segment boundaries were determined by the original section boundaries, and the topic labels were manually annotated given the titles of each section. We choose the en_city domain from WikiSection for experiments.

The statistical information about CSTS and WikiSection is summarized in Table 2.

Evaluation Metrics Following previous works [1, 3], we report P_k [4], F_1, and MAP for evaluation. P_k is the probabilistic error rate of a sliding window of size k when comparing the hypothesis with the reference segmentation. F_1 and MAP are topic classification metrics evaluated based on ground truth segments. All metrics are micro-averaged.

Implementation Details For model compatibility, we set the speaker role number R to 1 on WikiSection. We optimize the losses by an Adam optimizer [9] with $\beta_1 = 0.9, \beta_2 = 0.999, lr = 1e-4$. The hidden size d_m is 300. The dropout [24] rate is 0.1. The number of training epochs is 100 on CSTS and 20 on WikiSection. The smoothing hyperparameter λ in Eq. 4 is 3. The value of k for P_k is half the average segment length. We predict the topics of the ground truth segments with Eq. 2.

[1]https://github.com/sebastianarnold/WikiSection

Table 3: Overall comparisons. "PEN" means parallel extraction network without neighbor smoothing. Our proposed models are PEN and PEN-NS.

Model	CSTS			WikiSection		
	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑
SECTOR	25.3	66.2	73.4	15.5	71.6	81.0
LSTM-LSTM-CRF	17.9	83.6	n/a	9.7	77.5	n/a
S-LSTM	17.5	83.7	89.4	9.1	76.1	83.5
$\text{Transformer}^2_{\text{BERT}}$	16.9	84.5	91.0	9.1	76.3	83.7
PEN (**ours**)	16.2	85.2	91.1	8.2	78.9	85.1
PEN-NS (**ours**)	**15.1**	**87.7**	**92.8**	**8.0**	**80.0**	**86.1**

4.2 Overall Comparisons

Comparison Settings In this experiment, we compare PEN-NS as well as PEN (with one-hot labels) with the following models:

(1) SECTOR [1], which using pre-trained word embeddings [2, 17], first classifies the topic of each sentence or utterance and then merges them based on sentence embedding deviations.

(2) LSTM-LSTM-CRF [3, 12], which uses LSTMs for encoding and adopts IOB tagging [22] for segmentation and classification.

(3) S-LSTM [3], which adopts IOB tagging for sequence labeling and utilizes the exploration strategy to allow error recovery.

(4) $\text{Transformer}^2_{\text{BERT}}$ [14], which utilizes pre-trained BERT [5] and Transformer for encoding, and performs sequence labeling for segmentation. For a fair comparison, we also report the classification results from $\text{Transformer}^2_{\text{BERT}}$ based on sequence labeling.

Comparison Results Table 3 demonstrates the results of the overall comparisons, from which we can know that:

(1) PEN-NS outperforms all baselines by a notable margin in terms of both segmentation and classification metrics. Specifically, PEN-NS exceeds $\text{Transformer}^2_{\text{BERT}}$, the state-of-the-art model, by 1.8 points in P_k and 3.2 points in F_1 on CSTS, and by 1.1 points in P_k and 3.7 points in F_1 on WikiSection.

(2) Even without neighbor smoothing, PEN has outperformed existing models, including the pre-trained model $\text{Transformer}^2_{\text{BERT}}$. Note that PEN, S-LSTM and LSTM-LSTM-CRF all adopt LSTMs as the base encoder, so it is clear that the parallel extraction network has outperformed the sequence labeling methods that tend to focus on local contexts and the two-stage paradigm of segmentation and classification.

(3) Apart from our proposed models, $\text{Transformer}^2_{\text{BERT}}$ achieves the overall best performance due to the powerful capability of PLMs. The LSTM baselines perform better than SECTOR, which first performs classification and then decides segmentation points. This is because the topics are based on a whole segment instead of a single sentence or utterance, so the model often fails to identify the segment topics based on a single sentence or utterance.

4.3 Effectiveness of Parallel Extraction Network

Comparison Settings To explore the usefulness of the various components of PEN, we conduct the following experiments:

(1) Without parallel extraction (w/o parallel), segment extractions are performed serially on a single representation of segments.

(2) Without Bipartite Matching Optimization (w/o BMO), the matching is by the fixed order in which the segments appear.

Comparison Results The results in Table 4 reveal that:

(1) The replacement of parallel extraction with serial extraction leads to drastic performance drops. For example, the F_1 score decreases by 10.5 points on CSTS. The reason is that a single representation is not enough to handle multiple extractions, and the serial extraction policy is greedy without global optimization.

(2) The removal of BMO leads to performance degradations as well. For example, the P_k score increases by 0.8 points and the F_1 score decreases by 0.2 points on CSTS. It indicates that BMO is essential for establishing inter-segment dependency.

4.4 Effectiveness of Neighbor Smoothing

Comparison Settings To validate the effectiveness of neighbor smoothing, we compare it with the following alternatives:

(1) One-hot encoding labels (One-hot).

(2) Label smoothing with the uniform distribution (Uniform).

(3) Label smoothing with a random distribution (Random).

(4) Label smoothing with the Poisson distribution (Poisson).

Comparison Results The results are summarized in Table 5, from which we can know that:

(1) Neighbor smoothing outperforms the one-hot labels and the label smoothing variants in all metrics. Specifically, it exceeds the one-hot labels by 1.1 points in P_k and 2.5 points in F_1 on CSTS, and by 0.2 points in P_k and 1.1 points in F_1 on WikiSection.

(2) Neighbor smoothing gains consistent advantages on both datasets. It indicates that NS is effective not only on noisy data but also on well-segmented data such as WikiSection. Note that the performance gains on CSTS are more significant than on WikiSection since CSTS suffers from boundary ambiguity and data noise more frequently.

(3) Among the label smoothing variants, the Poisson distribution performs the best by classification metrics. Like neighbor smoothing, the Poisson distribution gives nearer segments more importance, so it reveals that smoothing the labels towards the neighhboring segments is beneficial for topic classification.

(4) Only neighbor smoothing demonstrates consistent performance gains in the segmentation metric P_k. The reason is that neighbor smoothing limits the neighbors to segments with at least one correct boundary to improve boundary identification.

4.5 Discussion

Performance on Noisy Data In this study, we empirically investigate the effectiveness of PEN and NS under noisy conditions by manually adding noise to the data. We apply a range of 10% to 50% of noise to the training data of CSTS by randomly moving the segment bounds one step forward or backward. As depicted in Figure 2, PEN-NS gains consistent advantages against PEN. Besides, PEN-NS and PEN consistently outperform the current state-of-the-art models in both metrics. Furthermore, the performance gap between our models and previous models widens as the noise ratio increases. The reasons are three-fold: (1) The combination of segmentation and classification reduces error propagation. (2) PEN performs extractions based on segment semantics to avoid focusing on the local context. (3) NS assigns more importance to the neighboring segments and simultaneously emphasizes the boundaries to increase robustness to data noise. So the results from Figure 2 further validate the above three assumptions.

Table 4: The effectiveness of the parallel extraction network. "w/o parallel" indicates segment extraction in a serial manner. "w/o BMO" indicates the removal of bipartite matching optimization.

Model	CSTS			WikiSection		
	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑
PEN-NS	**15.1**	**87.7**	**92.8**	**8.0**	**80.0**	**86.1**
w/o parallel	19.3	77.2	84.8	10.3	71.7	79.3
w/o BMO	15.9	87.5	92.4	8.1	79.8	85.9

Table 5: The effectiveness of neighbor smoothing. "One-hot" indicates training with one-hot labels. "Uniform", "Random", and "Poisson" indicates label smoothing with a Uniform distribution, with a random distribution, and with the Poisson distribution, respectively. "NS" indicates neighbor smoothing.

Method	CSTS			WikiSection		
	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑	$P_k \downarrow$	$F_1 \uparrow$	MAP ↑
One-hot	16.2	85.2	91.1	8.2	78.9	85.1
Uniform	16.1	78.3	86.1	8.5	79.2	85.3
Random	17.2	81.4	88.8	8.2	79.7	85.7
Poisson	17.7	86.2	91.9	8.3	79.6	85.6
NS	**15.1**	**87.7**	**92.8**	**8.0**	**80.0**	**86.1**

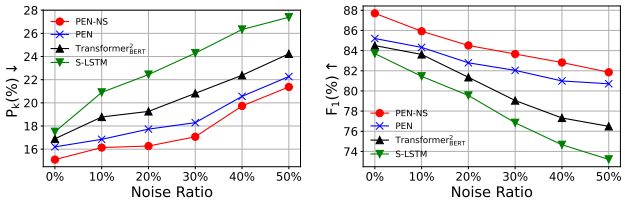

Figure 2: P_k (left) and F_1 (right) results on the CSTS dataset after some noise is added to the segment boundaries.

5 CONCLUSION AND FUTURE WORK

In this work, we propose a unified framework PEN-NS for dialogue topic segmentation. First, we propose the parallel extraction network to perform segment extractions for each topic and capture the segment dependency by minimizing the bipartite matching cost. Second, we propose neighbor smoothing to handle the ambiguity and noise of segment bounds. Experimental results show that PEN-NS outperforms existing state-of-the-art models on a dialogue-based and a document-based topic segmentation dataset. We also perform experiments under noisy conditions by moving the segment boundaries of the training data. The results show that PEN-NS is more robust to the noise of segment boundaries than the baseline models. In the future, we would like to explore other structures of dialogue topics like nested structure since PEN-NS is intrinsically applicable to the nested structure.

6 ACKNOWLEDGEMENTS

The work is supported by National Natural Science Foundation of China under Grant No.62036001 and PKU-Baidu Fund (No. 2020BD021).

REFERENCES

[1] Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A. Gers, and Alexander Löser. 2019. SECTOR: A Neural Model for Coherent Topic Segmentation and Classification. *Transactions of the Association for Computational Linguistics* 7 (2019), 169–184.

[2] Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A Simple but Tough-to-Beat Baseline for Sentence Embeddings. In *ICLR*.

[3] Joe Barrow, R. Jain, Vlad I. Morariu, Varun Manjunatha, Douglas W. Oard, and Philip Resnik. 2020. A Joint Model for Document Segmentation and Segment Labeling. In *ACL*.

[4] Doug Beeferman, Adam L. Berger, and John D. Lafferty. 2004. Statistical Models for Text Segmentation. *Machine Learning* 34 (2004), 177–210.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

[6] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9 (1997), 1735–1780.

[7] Jeremy Howard and Sebastian Ruder. 2018. Universal Language Model Fine-tuning for Text Classification. In *ACL*.

[8] Joo-Kyung Kim, Guoyin Wang, Sungjin Lee, and Young-Bum Kim. 2021. Deciding Whether to Ask Clarifying Questions in Large-Scale Spoken Language Understanding. *2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2021), 869–876.

[9] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2015).

[10] Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text Segmentation as a Supervised Learning Task. In *NAACL*.

[11] Harold W. Kuhn. 2010. The Hungarian Method for the Assignment Problem. In *50 Years of Integer Programming*.

[12] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *NAACL*.

[13] Chunyi Liu, Peng Wang, Jiang Xu, Zang Li, and Jieping Ye. 2019. Automatic Dialogue Summary Generation for Customer Service. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (2019).

[14] Kelvin Lo, Yuan Jin, Weicong Tan, Ming Liu, Lan Du, and Wray L. Buntine. 2021. Transformer over Pre-trained Transformer for Neural Text Segmentation with Enhanced Topic Coherence. In *EMNLP*.

[15] Fabrizio Macagno and Sarah Bigi. 2018. Types of dialogue and pragmatic ambiguity.

[16] Ryo Masumura, Takanobu Oba, Hirokazu Masataki, Osamu Yoshioka, and Satoshi Takahashi. 2014. Role play dialogue topic model for language model adaptation in multi-party conversation speech recognition. *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (2014), 4873–4877.

[17] Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. In *ICLR*.

[18] Pedro Mota, Maxine Eskénazi, and Luísa Coheur. 2019. BeamSeg: A Joint Model for Multi-Document Segmentation and Topic Identification. In *CoNLL*.

[19] Rafael Müller, Simon Kornblith, and Geoffrey E. Hinton. 2019. When Does Label Smoothing Help?. In *NeurIPS*.

[20] Artem Popov, Victor Bulatov, Darya Polyudova, and Eugenia Veselova. 2019. Unsupervised dialogue intent detection via hierarchical topic model. In *RANLP*.

[21] MengNan Qi, Hao Liu, Yuzhuo Fu, and Ting Liu. 2021. Improving Abstractive Dialogue Summarization with Hierarchical Pretraining and Topic Segment. In *EMNLP*.

[22] Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text Chunking using Transformation-Based Learning. *ArXiv* cmp-lg/9505040 (1995).

[23] Imran A. Sheikh, D. Fohr, and Irina Illina. 2017. Topic segmentation in ASR transcripts using bidirectional RNNS for change detection. *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)* (2017), 512–518.

[24] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15 (2014), 1929–1958.

[25] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *NIPS*.

[26] Linzi Xing, Bradley Alexander Hackinen, Giuseppe Carenini, and Francesco Trebbi. 2020. Improving Context Modeling in Neural Topic Segmentation. In *AACL*.

[27] Xingqian Xu, Zhifei Zhang, Zhaowen Wang, Brian L. Price, Zhonghao Wang, and Humphrey Shi. 2021. Rethinking Text Segmentation: A Novel Dataset and A Text-Specific Refinement Approach. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), 12040–12050.

[28] Yi Xu, Hai Zhao, and Zhuosheng Zhang. 2021. Topic-Aware Multi-turn Dialogue Modeling. In *AAAI*.

[29] Seunghyun Yoon, Joongbo Shin, and Kyomin Jung. 2018. Learning to Rank Question-Answer Pairs Using Hierarchical Recurrent Encoder with Latent Topic Clustering. In *NAACL*.

[30] Hainan Zhang, Yanyan Lan, Liang Pang, Hongshen Chen, Zhuoye Ding, and Dawei Yin. 2020. Modeling Topical Relevance for Multi-Turn Dialogue Generation. In *IJCAI*.

[31] Yujun Zhou, Changliang Li, Saike He, Xiaoqi Wang, and Yiming Qiu. 2019. Pre-trained Contextualized Representation for Chinese Conversation Topic Classification. *2019 IEEE International Conference on Intelligence and Security Informatics (ISI)* (2019), 122–127.

[32] Lin Zhu, Xinnan Dai, Qihao Huang, Hai Xiang, and Jie Zheng. 2019. Topic Judgment Helps Question Similarity Prediction in Medical FAQ Dialogue Systems. *2019 International Conference on Data Mining Workshops (ICDMW)* (2019), 966–972.

[33] Yicheng Zou, Lujun Zhao, Yangyang Kang, Jun Lin, Minlong Peng, Zhuoren Jiang, Changlong Sun, Qi Zhang, Xuanjing Huang, and Xiaozhong Liu. 2021. Topic-Oriented Spoken Dialogue Summarization for Customer Service with Saliency-Aware Topic Modeling. In *AAAI*.