Automatically Annotate TV Series Subtitles for Dialogue Corpus Construction

Leilan Zhang† , Qiang Zhou*
† Department of Computer Science and Technology, Tsinghua University, Beijing, China
E-mail: zll17@mails.tsinghua.edu.cn
* Center for Speech and Language Technologies, Tsinghua University, Beijing, China
E-mail: zq-lxd@mail.tsinghua.edu.cn

***Abstract*—In recent years, the scarcity of dialogue corpus is becoming the bottleneck of Chinese dialogue generation systems. Although subtitles provide favorable material to construct dialogue corpus because of their abundance and diversity, lacking speaker information makes it hard to extract dialogues from subtitles directly. To utilize these resources, we proposed an improved method to automatically annotate bilingual TV subtitles with speaker and scene tags using their corresponding scripts. First, tags of speakers and scene boundaries in the scripts are mapped to the subtitles through an information retrieval method. Then, the mapping errors are detected with a convolutional network and corrected by heuristic strategies to improve the annotation quality. We applied this method on 779 bilingual subtitle files of 4 TV series and obtained a Chinese dialogue corpus Tv4Dialog[1] containing 260674 utterances. Experiment result shows that our method can achieve an accuracy of 94.62% on speaker tag annotation, improving nearly 12% on the previous state-of-the-art result.**

I. Introduction

At present, large-scale multi-turn dialogue corpus is a necessary constituent in a data-driven dialogue generation system. However, it is a surprising fact that current researches on multi-turn dialogue datasets are relatively scarce: among the 56 surveyed available corpora, only 9 of them contain more than 100K dialogues [17]. The situation in the Chinese field is even more severe. Available Chinese dialogue corpora are either task-oriented datasets, like CASIA-CASSIL [23], or datasets collected from social media, like Weibo [18] and Tieba [9], which are in the form of post-response pairs but not like human daily conversation. Dataset DailyDialog is a natural multi-turn dialogue corpus, however, it contains only 13118 English dialogues and is manually labeled, which makes it hard to expand [10].

It is noteworthy that TV series subtitles provide a rich resource of dialogue data. Compared with the social media data, subtitles provide a form of multi-turn dialogues and their style is much closer to human daily conversation. Extracting dialogues from TV subtitles, a large-scale Chinese dialogue corpus could be obtained for dialogue generation research.

However, without speaker or scene notes, subtitle lacks structural information. It is hard to judge whether two continuous subtitle lines belong to the same speaker or not because of the lack of obvious marks between them, which prevents drawing out the available dialogue content directly. On the other hand, TV scripts can provide rich internal structure information including speakers and scene boundaries. A material fact is that few Chinses TV scripts is publicly accessible, however, enough amount of English scripts are openly available online. Due to the fact that Chinese and English sentences are naturally aligned in bilingual subtitles and subtitles are usually share most same dialogue content with their scripts, we consider aligning these two types of resources (i.e. English scripts and bilingual subtitles) at the sentence level and annotating the subtitles with speaker and scene boundary tags to construct a Chinese dialogue corpus.

The annotation process can be splited into two main steps: 1) roughly alignment: for each subtitle line, the best matching utterance in the corresponding script is selected through information retrieval techniques and its speaker and scene tags are mapped to the subtitle line; 2) error detection and correction: after detecting some mapping errors produced in the first step based on a convolutional neural network, some heuristic strategies are used to correct these errors for the annotation quality improvement.

We apply this method on subtitles of 4 TV series and manually annotate 4000 utterances as ground truth to evaluate the annotation accuracy. Compared to the previous best result [20], our corpus has been expanded by 3 times in scale and our method has improved the accuracy of speaker annotation by nearly 12%.

The contributions of this paper can be summarized as follows: 1) We proposed a method to annotate bilingual TV subtitles with speaker and scene boundary tags using corresponding scripts; 2) The Chinese dialogue corpus provided in this research achieves the state-of-the-art result both in scale and in annotation accuracy; 3) The method we proposed to identify mapping errors can be generalized to the area of anomaly detection in time-series data.

To facilitate related research, we have publicly released our data including the structured XML scripts (260674 utterances, 18129 scenes, in English) and the annotated subtitles (779 files, both in English and Chinese).

II. Related Work

Although lots of effort has been invested in the construction of dialogue corpus, the situation is still severe. Dialog datasets

[1]It is publicly available at https://github.com/zll17/TV4Dialog

such as Ubuntu [12] and restaurant reservation datasets [3] are large in scale but their conversations are single-turn and task-oriented, which limits the range of their application fields. Dialogue datasets like Switchboard [15] and OpenSubtitles [19] often contain more than 150 turns in one conversation, which are too disperse to grasp the main topic. Compared with the above-mentioned English datasets, the Chinese dialogue datasets are even more insufficient. Chinese spoken dialogue datasets like CSDC [8] and CASIA-CASSIL [23] are usually small-scale and domain-specific. Question-Answering datasets like Douban [22], InsuranceQA [7] and Weibo [18] are collected from social media, whose conversational pattern is different from daily life.

As a huge potential resource, movie and TV subtitles and their scripts have been explored for different purposes. Researches on extracting structure of scripts [5] [13] [2] [11] mostly attempts to match specific patterns of movie scripts. [21] has been successfully extracted conversations from more than 900 movie scripts.

Early researches on script-subtitles alignment usually adopted dynamic programming algorithm to search an optimal path between two sequences and detect an optimal alignment between them [4][6][14]. [11] employed sentence aligners to annotate speaker tags from movie scripts to subtitles. However, they didn't publicly release their data. [20] used information retrieval techniques to solve this alignment task and build a parallel corpus on TV series *Friends*. In their work, TF-IDF indicator is adopted to measure the similarity and a sliding window is imposed to make the annotation more accurate. They finally achieved an accuracy of 81.79% on speaker annotation and 98.64% on scene boundary annotation.

In terms of building the corpus, TV series have two main advantages over movies. On the one hand, due to the age of production, most public English movie scripts have no corresponding Chinese subtitles. On the other hand, compared with TV series's scripts' uniformity, the diversity of their formats makes it much harder to expand the movie scripts dataset. Therefore, we only dealt with subtitles of TV series instead of those of movies.

III. Align Scripts and Subtitles

We collected scripts of 4 TV series (*Castle*[2], *Friends*[3], *House*[4] and *The Big Bang Theory*[5] (*TBBT*)) and their corresponding subtitles[6] from Internet. Since the script is a semi-structured text, they must be parsed before the data can be utilized. Scripts of each TV series have their own format features. Generally, there are three kinds of elements in TV series scripts : 1) scene heading, which usually contains strings like 'Scene', 'INT', 'EXT' etc., marks the start of a scene and can be adopted to indicate the scene boundary; 2) speaker name, which usually appears at the begin of a line and followed by a colon; 3) dialogue content, which usually follows a speaker name and lasts until the end of the line. In this paper, a scene is defined as all the contents between two adjacent scene headings, and an utterance is defined as a speaker with its corresponding speaking content, Figure 1 displays a snapshot of a script taken from *TBBT* Season 2 Episode 8 (S02E08).

```
Sheldon: Oh look, Saturn 3 is on.
Raj: I don't want to watch Saturn 3. Deep Space Nine is better.
Sheldon: How is Deep Space Nine better than Saturn 3?
Raj: Simple subtraction will tell you it's six better.
Leonard: Compromise. Watch Babylon 5.
Sheldon: In what sense is that a compromise?
Leonard: Well, five is partway between three... Never mind.
Raj: I'll tell you what, how about we go rock-paper-scissors?
```

Fig. 1. Excerpt of a raw script

According to those features, we can design parsers to extract the above-mentioned elements. The algorithm in this section is more engineering. First, it will filter out all action instructions and scene descriptions (usually enclosed in parentheses). Then, it will scan the script line by line and use different patterns to detect specific elements. Once a scene heading was detected, it will end the previous scene and start a new scene. Finally, those transformed utterances will be output as XML (Extensible Markup Language) format.

```
<utterance uid="1-1">
  <speaker>Sheldon</speaker>
  <content>Oh look, Saturn 3 is on.</content>
</utterance>
<utterance uid="1-2">
  <speaker>Raj</speaker>
  <content>
    I don't want to watch Saturn 3. Deep Space Nine is better.
  </content>
</utterance>
<utterance uid="1-3">
  <speaker>Sheldon</speaker>
  <content>
     How is Deep Space Nine better than Saturn 3?
  </content>
</utterance>
<utterance uid="1-4">
  <speaker>Raj</speaker>
  <content>
     Simple subtraction will tell you it's six better.
  </content>
</utterance>
<utterance uid="1-5">
  <speaker>Leonard</speaker>
  <content>Compromise. Watch Babylon 5.</content>
</utterance>
```

Fig. 2. The processed script

Figure 2 displays the processed format of the above script in Figure 1, in which the <scene> tag stands for a scene boundary, the **id** attribute means its index number. The <utterance> tag marks a dialogue utterance, its **uid** attribute indicates its order number in a scene, for example, uid= '4-3', means that this utterance is the **3rd** utterance of the **4th** scene. Therefore, two utterances with same prefix of their uid (eg. 5-6 and 5-2) are within a same scene. In a script of an episode, every

[2]http://dustjackets.wikifoundry.com/page/Transcripts
[3]https://fangj.github.io/friends
[4]https://clinic-duty.livejournal.com/
[5]https://bigbangtrans.wordpress.com/
[6]http://assrt.net/

utterance owns a unique **uid** and every **uid** attribute can be used to represent an utterance. Therefore, once known the corresponding **uid** of a subtitle line, the speaker name then can be determined and projected to the subtitle line.

Subtitles are structured in blocks, each of which is composed of a text content line and a timestamp line. In our paper, subtitle line refers to the text content line.

Generally, utterances in script can not always match the subtitle line exactly. Modification, deletion or actors' impromptu performance could all produce the differences between script utterances and subtitle lines. Considering the case that a long utterance in a script could be split into several short subtitle lines or the case that two short utterances in the script could be merged into a single subtitle line, the correspondence from a script to a subtitle might be one-to-one, one-to-many or many-to-one.

Therefore we consider handling this problem using information retrieval techniques. Utterances in scripts are taken as documents and each subtitle line is treated as a query, the task is to select the best matching utterance (a document) for the subtitle line (the query) and project the utterance's **uid** to the subtitle line. To be specific, BM25 is employed as the score function to measure the relevance between a document and a query [16].

In practice, we choose ElasticSearch[7] (an open source search engine) for the indexing and search task, and annotate the subtitle line with **uid** of its best match utterance in the script. Figure 3 displays two snapshots of the annotated subtitles (taken from *Friends* S02E08 and *TBBT* S02E11 respectively), **uid**s are wrapped with two brackets at the begining of subtitle lines.

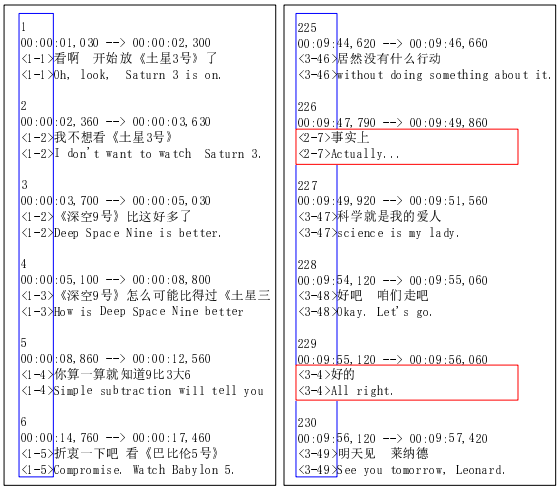

Fig. 3. Examples of annotated subtitles

IV. Error Detection and Correction

A. Error Analysis

As mentioned aboved, the fact that differences in both content and format between subtitles and their corresponding scripts would increase the difficulty of alignment, phenomena such as tranformed expressions, repeated dialogue content, short text may lead to alignment errors occurring during the mapping process.

In genereal, the mapping error is caused because there exists such utterance, which has the highest similarity with the mismatched subtitle line owing to its tokens and length. For example, When a long utterance in a script is splited into several lines in a subtitle, and there exists another similar but shorter utterance in the script, then the shorter utterance may have higher similarity with the queried subtitle line since the shorter length makes higher term frequency. Figure 4 shows such error case: the first subtitle line ("Peter?") in the left blue box should have matched the utterance (**uid**="14-7") in the bottom right blue box which contains 6 tokens, while it is actually mapped to the utterance (**uid**="7-7) in the top right red box since the latter contains only one token "Peter". (*Friends* S03E24)

Fig. 4. Mapping Error Case I

These mapping errors would result in a situation that wrong **uid**s will be placed among correct annotated **uid**s. As the left snapshot in Figure 3 shows, **uid** tags appearing in the blue box indicates that the whole tag sequence of a subtitle is actually an approximately ordered array of number pairs. However, the mapping errors will lead to inconsistence: the blue box in the right snapshot of Figure 3 is a tag sequence contains alignment errors, and the red boxes point out the wrong projected **uid**s, the whole sequence is [⟨3-46⟩,⟨2-7⟩,⟨3-47⟩,⟨3-48⟩,⟨3-4⟩,⟨3-49⟩], according to the tag context, **uid** appearing in the first red box should be ⟨3-46⟩ or ⟨3-47⟩ while it is ⟨2-7⟩ in reality and **uid** appearing in the second red box should be ⟨3-48⟩ or ⟨3-49⟩ while it appears as ⟨3-4⟩ actually. Therefore, **uid** tags extracted from the annotated subtitles can be considered as increasing point pairs series, and mapping errors which usually leads to inconsistency can be treated as anomaly in the series. Then this task can be handled with two main steps: first, tag sequence will be feed into a model and anomalies will be detected and substituted with a special tag ⟨0,0⟩; second, those special tags will be restored using some strategies according to their tag context.

B. Error Detection

Since the **uid** tag sequence is an almost ordered sequence, an intuitive method is to calculate its difference sequence and

[7]https://www.elastic.co

detect the positions whose differences are abnormally larger than others'. We attempted to impose a sliding window to handle this task. However, the size of the window became a problem: if the size was set too small, the window could be covered by a segment of continuous wrong **uid** tags, then it cannot correct any one of them; if the size was set too large, there might be several segments of wrong **uid** tags in the window and it cannot tell which are outliers or not. The difficulty of this problem is the wrong tags should be detected according to the right ones, however, those right tags cannot be recognized unless the pattern of the sequence was known. Therefore, this problem could be handled as sequence modeling task.

Temporal Convolutional Networks (TCN for short), which beat RNN on 11 standard sequence modeling tasks and achieve 10 the-state-of-the-art results [1], uses the convolutional operation to deal with sequence modeling problems. Compared with RNN, the convolutional architecture allows TCN compute in parallel and results in a faster speed. TCN is designed with two features: 1) its output has the same length as the input ; 2) it only uses the past information and there's no leakage from the future to past. Those features make it a suitable model for this anomaly detection task. A classifier based on TCN can be used to classify whether a **uid** tag in a sequence is a normal tag or an anomaly. Figure 5 displays the architecture of our detection model. The input is a **uid** sequences, each dimension of the **uid** is taken as one channel of the input layer. The network has 8 hidden layers and its kernel size is set as 7 empirically, for the reason that the length of a continuous anomaly segment is usually no more than 7. The output is a binary array with the same length as the **uid** tag sequence, and the anomalies would be labeled with 1. For example, in Figure 5 the **uid** ⟨3-2⟩ highlighted with red color in the input sequence is an anomaly tag, so the network outputs 1 to demonstrate its classification result.

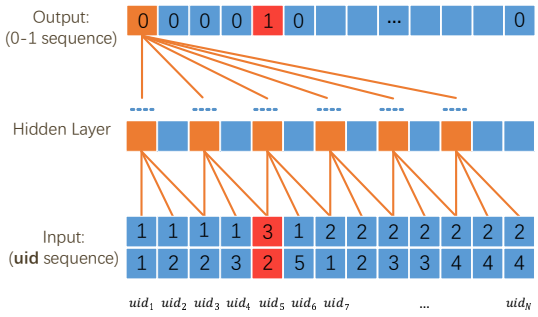

Fig. 5. Architecture of Error Detection Model.

This detection model is a supervised learning method, which requires labeled training data. To tackle the problem of lacking training data, we decided to artificially generate the training data, which is as close to the real data as possible, specifically, to ensure that they have same distribution on the number and lengths of scenes in a tag sequence. Since the first and the second dimension of a **uid** stands for the index of a scene in a tag sequence and the index of an utterance in a scene respectively, the maximum value of this first dimension is equal to the number of scenes in a tag sequence and the length of a scene is equal to the maximum value of the second dimension of **uid**s in this scene. Given the number of scenes in a sequence and the length's range for each scene, we can produce an ordered tag sequence automatically. Therefore, we made statistics to estimate the distributions of **uid** in different TV series.

Algorithm 1 Generate Ordered Sequence

Input: $s\mu$,$s\sigma$,$u\mu$,$u\sigma$
Output: Tag Sequence with Order

Let $lst = [\,]$.
Let $sMax = Normal(s\mu, s\sigma)$
while $sMax < 2$ **do**
  $sMax = Normal(s\mu, s\sigma)$
end while
for $sid = 0$ to $sMax - 1$ **do**
  $sid = sid + 2$
  $uMax = Normal(u\mu, u\sigma)$
  while $uMax < 2$ **do**
    $uMax = Normal(u\mu, u\sigma)$
  end while
  Let $tmp = [\,]$
  for $uid = 0$ to $uMax - 1$ **do**
    $tmp = tmp + [[sid, uid + 1]]$
  end for
  $uLst = uLst + tmp$
end for
return $uLst$

Figure 6 displays the statistical results. The plots on the first column correspond to the number of scenes per script and the plots on the second column correspond to the lengths of scenes (the numbers of utterances per scene). The results show that the two indicators of the 4 TV series all approximate normal distributions. And it illustrates the reasonability of the normal distribution assumption in Algorithm I. According to the mean value and the standard variance of the scenes' numbers and lengths shown in Figure 6, sequences which have the same statistical features with real tag sequences can be generated using Algorithm I. The algorithm receives 4 parameters: $s\mu, s\sigma, u\mu, u\sigma$, meaning the mean value and the standard variance of the number and the lengths of scenes respectively. The algorithm also uses two while loops to ensure that the sequence it produced contains more than 2 scenes.

Compared with real tag data, those sequences are strictly ordered without inconsistency. Therefore, anomalies should be added intentionally into the sequences. According to section 4.1, the formation of mapping errors means that some **uid**s are placed at wrong positions and some of them are even misplaced for several times. To imitate this process, techniques such as switching two randomly chosen **uid**s, repeatedly

inserting a **uid** for multiple times are utilized to imitate anomalies in real tag data. Labels are constructed simultaneously: when an anomaly is added into a sequence, the anomaly's corresponding label will be set as one. For each TV series, we constructed the corresponding training set containing 200K sequences to train its error detection model.

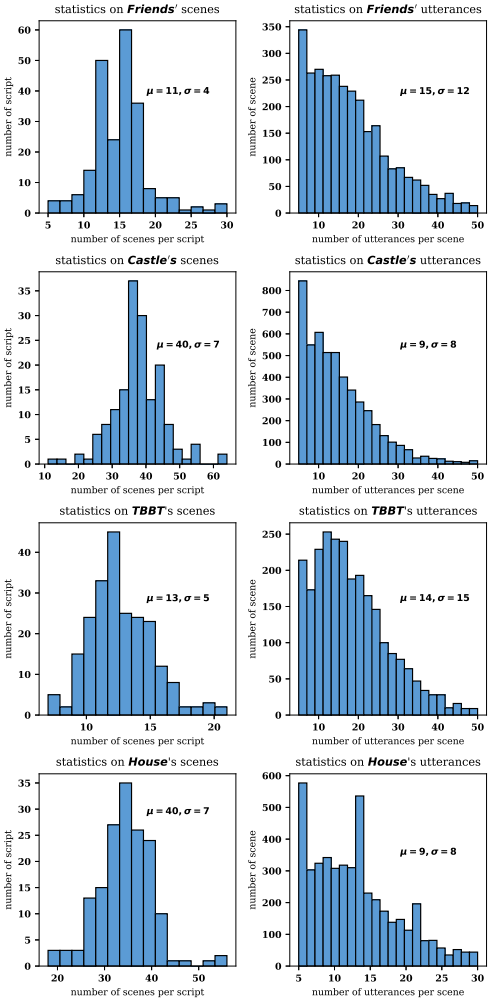

Fig. 6. The distribution of the number and lengths of the 4 TV series' scenes

C. Error Correction

To correct the detected anomalies in the sequence, another TCN model has been established using the similar method described above, however, experiment result shows it might change some right **uid**s into wrong ones due to the continuous property of the neural network. Therefore, a heuristic algorithm is proposed to rebuild the sequence with detected anomalies. The algorithm uses several strategies to rebuild the identified errors:

1) if the anomaly tag is an isolated point, it will be replaced with its front or back point's **uid** according to whether it located at the boundary or at the internal area of a scene;

Fig. 7. Several cases for correction algorithm

2) if the anomaly tags form a continuous segment which are totally in the interal of a scene, they will be substitued directly by same number of points which are calculated using linear interpolation algorithm;
3) if the anomaly tags form a continuous segment and the segment intersects with two scenes, the back part will be replaced by progressively decreased point pairs, then the front part then will be calculated by interpolation algorithm.

Figure 7 enumerates several cases and their correction results using these strategies.

Combine the well-trained TCN and the correction algorithm, mapping errors could be detected and corrected. Then both the speaker tags and **uid** sequence can be annotated to subtitles. If a **uid** created by the correction algorithm cannot be found in original script, its speaker name would be replaced by the previous one. Figure 8 shows the corrected subtitle appearing in Figure 3.

V. Experiments and Results

We first conducted experiment to estimate the capability of the TCN model. An independent test set is constructed under the same generation algorithm. Evaluation shows that this model can achieve a F1 score of 0.97 on the test set.

We also measure the ability of this heuristic algorithm with restoring accuracy: $acc = n/len$, where n is the number of same **uid** tags between the restored sequence and the original one, and len is the tag sequence's length. In our case, the algorithm achieves an average restoring accuracy of 0.95.

Applying the method to all the subtitles of the four TV series, we finally obtained 779 structured scripts and 779 annotated Chinese-English subtitles containing 18129 scenes. The main statistics are presented in Table I and Table II.

The validation dataset is constructed as follows: for each TV series, we select a subtitles from each season and manually

annotated 100 lines per subtitle file with speaker and **uid** tags according to their corresponding scripts. For example, for TV series *TBBT*, subtitles of S01E01, S02E02, ..., S10E10 were chosen, and for the *i*th subtitle S*0i*E*0i*, the lines range from 1 to 100 were manually labeled. Automatic annotation results are then compared to these manually annotated tags to evaluate the accuracy on utterances and scene boundaries respectively. Experiment results are displayed in Table III and Table IV in detail.

In TableIII, a noteworthy point is that the ratio of right mapped tags are all higher than 80% with BM25 only, which gives chance to correct those mapping errors according to the right ones. Although our method can correctly align most of the subtitle lines with their corresponding utterances, the cases that two short utterances are merged into one single subtitle line still cannot be handled, and such cases acount for 0.037 in all subtitle lines. Other possible reasons why this method cannot completely correct all the mapping errors are: 1) there exist differences between the distribution of the artificially generated training data and that of the real tag sequences; 2) the restoring strategy is heuristic, which possibly deviates from the practical situation.

TABLE I
STATISTICS OF THE CORPUS

Item	Size
Total num of structured scripts	779
Total num of scenes	18129
Total num of utterances	260674
Average num of scenes per script	23
Average num of utterances per scene	14

TABLE II
STATISTICS OF THE 4 TV SERIES' SCRIPTS

	tbbt	house	friends	castle
num of episodes	225	164	227	163
num of scenes	2839	5652	3499	6139
num of uttrances	50161	69380	59859	81274
num of speakers	484	1039	691	2074
spkrs per scene	3.56	3.00	3.47	3.30
avg uttr length	11.23	11.37	10.13	12.29

According to [20], using TF-IDF as weighting factor along with moving window strategy to align script utterances and subtitle lines of *Friends* can achieve 81.79% and 98.64% on utterance and scene annotation accuracy respectively. Our experiment on the same TV series *Friends* shows that only using BM25 as rank function without error correct procedure or other strategies can achieve an accuracy of 85.2% on utterance and 93.3% on scene boundaries, which indicates that BM25 is more powerful in short text query than TF-IDF. The evaluation of this method was also carried out on other three TV series and it achieved an average accuracy of 84.4% on utterance.

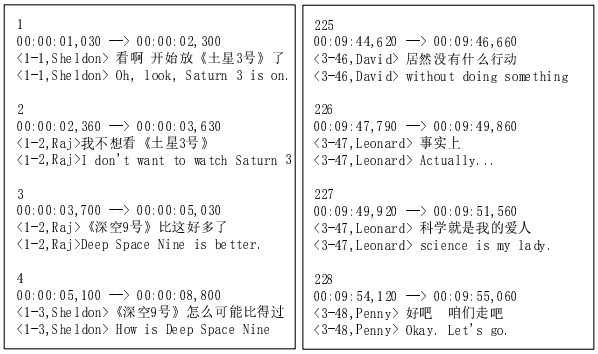

Fig. 8. Subtitles with corrected annotation

TABLE III
AVERAGE ANNOTATION ACCURACY ON UTTERANCE

	TBBT	*Friends*	*Castle*	*House*
TFIDF	0.825	0.818	0.793	0.776
BM25	0.887	0.852	0.812	0.809
BM25+TCN	**0.949**	**0.933**	**0.952**	**0.951**

TABLE IV
AVERAGE ANNOTATION ACCURACY ON SCENE BOUNDARIES

	TBBT	*Friends*	*Castle*	*House*
TFIDF	0.952	0.986	0.936	0.932
BM25	0.943	0.962	0.926	0.934
BM25+TCN	**0.992**	**0.989**	**0.975**	**0.983**

The accuracy on utterance is calculated as m_u/n_u, where m_u denotes the number of right annotated utterance, and n_u denotes the total number of utterances in a subtitle. The accuracy on scene boundaries is computed as m_s/n_s, where m_s and n_s denote the number of right annotated scene boundaries and the total number of scene boundaries in a subtitle respectively.

The result shows that tag sequence through the correction process can achieve a 94.62% accuracy on the utterance, which is 10.62% higher than that without denoising procedure, indicating the effectiveness of our error correction method.

VI. CONCLUSION

This paper expanded the research work conducted by [20] on both the scale and quality of data. We built a dialogue corpus annotated with scene and speaker tags using TV subtitles. Through the method proposed in this paper, we moved out annotation errors effectively and obtained a dialogue corpus with 18129 dialogues and 260674 utterances. Researches in the field of dialogue system could be benefit from this corpus.

ACKNOWLEDGMENT

We would like to give thanks to the administrator of the website assrt.net[8] for the help of providing the mirror of their subtitle database.

[8]http://assrt.net/

REFERENCES

[1] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling." *CoRR* vol. abs-1803-01271 (2018)

[2] Rafael E. Banchs, "Movie-dic: a movie dialogue corpus for research and development." *In Proc. of the 50 th Annual Meeting of the ACL.* (2012)

[3] Antoine Bordes and Jason Weston, "Learning end-to-end goal-oriented dialog." *ICLR* (2017)

[4] Timothee Cour, Chris Jordan, Eleni Miltsakaki, and Ben Taskar, "Movie/script: Alignment and parsing of video and text transcription." *Computer Vision – ECCV 2008*, pages 158–171, Berlin, Heidelberg. Springer Berlin Heidelberg. (2008)

[5] Cristian Danescu-Niculescu-Mizil and Lillian Lee, "Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs." *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*. (2011)

[6] Mark Everingham, Josef Sivic, Andrew Zisserman, "Taking the bite out of automated naming of characters in tv video." *The 17th British Machine Vision Conference*. pp. 27(5):545–559. (2009)

[7] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou, "Applying deep learning to answer selection: A study and an open task." *IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU) (pp. 813-820).IEEE* (2015)

[8] Changliang Li and Xiuying Wang, "Building large chinese corpus for spoken dialogue research in specific domains." *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, pages 320–324. (2017)

[9] Jing Li, Yan Song, Haisong Zhang, and Shuming Shi, "A manually annotated chinese corpus for non-task-oriented dialogue systems." *CoRR abs/1805.05542* (2018)

[10] Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu, "Dailydialog: A manually labelled multi-turn dialogue dataset." *Proceedings of the Eighth International Joint Conference on Natural Language Processing*, vol 1, pp. 986–995. (2017)

[11] Pierre Lison and Raveesh Meena, "Automatic turn segmentation for movie and tv subtitles." *2016 IEEE Spoken Language Technology Workshop (SLT)*, pp. 245–252. (2016)

[12] Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau, "The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems." *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. [*Prague, Czech Republic,* pp. 285–294].(2015)

[13] Christos Makris and Pantelis Vikatos, "Community detection of screenplay characters." *AIAI*. (2016)

[14] S. Park, H. Kim, H. Kim, and G. Jo, "Exploiting script-subtitles alignment to scene boundary dectection in movie." *2010 IEEE International Symposium on Multimedia*, pp. 49–56. (2010)

[15] Volha Petukhova, Martin Gropp, Dietrich Klakow, Gregor Eigner, Mario Topf, Stefan Srb, Petr Motlícek, Blaise Potard, John Dines, Olivier Deroo, Ronny Egeler, Uwe Meinz, Steffen Liersch, and Anna Schmidt, "The dbox corpus collection of spoken human-human and human-machine dialogues." *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. (2014)

[16] Stephen Robertson, Hugo Zaragoza, and Michael Taylor, "Simple bm25 extension to multiple weighted fields." *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 42–49, New York, NY, USA. ACM.(2004)

[17] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau, "A survey of available corpora for building data-driven dialogue systems." *arXiv preprint arXiv:1512.05742*.(2015)

[18] Lifeng Shang, Zhengdong Lu, and Hang Li, "Neural responding machine for short-text conversation." *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing* , vol 1, pp. 1577–1586.(2015)

[19] Jörg Tiedemann, "News from OPUS—A Collection of Multilingual Parallel Corpora with Tools and Interfaces", volume 5, pp. 237–248.(2009)

[20] Longyue Wang, Xiaojun Zhang, Zhaopeng Tu, Andy Way, and Qun Liu. , "Automatic construction of discourse corpora for dialogue translation. *CoRR*(2016)

[21] David Winer and R. Young, "Automated screenplay annotation for extracting storytelling knowledge", *Thirteenth Artificial Intelligence and Interactive Digital Entertainment Conference*.(2017)

[22] Yu Wu, Wei Wu, Ming Zhou, and Zhoujun Li, "Sequential match network: A new architecture for multi-turn response selection in retrieval-based chatbots. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, vol 1, pp. 496–505, Vancouver, Canada (2016)

[23] Keyan Zhou, Aijun Li, Zhigang Yin, and Chengqing Zong, "Casia-cassil: a chinese telephone conversation corpus in real scenarios with multi-leveled annotation. *Proceedings of the Seventh conference on International Language Resources and Evaluation*.(2010)