INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

A Neural-Network-Based Approach to Identifying Speakers in Novels

Yue Chen[1], *Zhen-Hua Ling*[1], *Qing-Feng Liu*[1,2]

[1]National Engineering Laboratory for Speech and Language Information Processing, University of Science and Technology of China, Hefei, P. R. China
[2]iFLYTEK Co., Ltd., Hefei, P. R. China

cy321@mail.ustc.edu.cn, zhling@ustc.edu.cn, qfliu@iflytek.com

Abstract

Identifying speakers in novels aims at determining who says a quote in a given context by text analysis. This task is important for speech synthesis systems to assign appropriate voices to the quotes when producing audiobooks. However, existing approaches stick with manual features and traditional machine learning classifiers, which constrain the accuracy of speaker identification. In this paper, we propose a method to tackle this challenging problem with the help of deep learning. We formulate speaker identification as a scoring task and build a candidate scoring network (CSN) based on BERT. Candidate-specific segments are put forward to eliminate redundant context information. Moreover, a revision algorithm is designed utilizing the speaker alternation pattern in two-party dialogues. Experiments have been conducted using the dataset built on the Chinese novel *World of Plainness*. The results show that our proposed method reaches a new state-of-the-art performance with an identification accuracy of 82.5%, which outperforms the baseline using manual features by 12%.

Index Terms: speaker identification, BERT, dialogue analysis, speech synthesis

1. Introduction

Identifying speakers in novels is an important task for many downstream applications, such as assigning appropriate voices to utterances when producing audiobooks [1, 2], and creating scripts based on novels [3]. As dialogues serve as an important mean of shaping characters in literature, automatic identification of speakers can also be useful for text mining tasks like extracting the social network of characters [4, 5].

Existing speaker identification methods can be divided into rule-based ones and machine-learning-based ones. Rule-based methods [6, 7] focused on designing linguistic rules so as to make decisions among speaker candidates. These methods heavily relied on the knowledge of developers and usually resulted in low accuracy and poor generalization ability. On the other hand, machine-learning-based methods [8–11] utilized manually-labelled training data to build classifiers, such as support vector machines (SVM) [12] and multi-layer perceptrons (MLP) [13], to determine the speakers of quotes. However, all these studies still adopted hand-crafted linguistic features, which were designed heuristically and failed to describe the input text in a comprehensive way.

In recent years, numerous studies in natural language processing (NLP) have demonstrated that neural-network-based feature extraction is far more effective than manual feature engineering [14, 15], since it can jointly optimize the feature extractor and the classifier, and can therefore learn globally optimal features. Deep neural architectures like recurrent neural networks (RNN) [16] and convolutional neural networks (CNN) [17] have been popularly applied to various NLP tasks and outperformed the traditional machine learning models. More recently, pretrained language models, such as BERT [18] and GPT [19], have achieved state-of-the-art performance on many shared tasks, owing to their attention-based architectures and knowledge obtained from a massive amount of pretraining data.

Therefore, this paper proposes a neural-network-based approach to solve the task of identifying speakers in novels. We formulate this task as a scoring problem and build a candidate scoring network (CSN) to calculate the score for each candidate speaker. In consideration of the limited size of annotated data, the pretrained architecture for natural language understanding, BERT [18], is adopted as the foundation of CSN. To reduce redundant context information, candidate-specific segments are extracted from quote and context sentences, and are sent into BERT to derive the representations of candidates, contexts, and quotes respectively. Then, these representations are concatenated and fed into an MLP scorer to obtain the score for each candidate speaker. The candidate with the highest score is determined as the speaker of the quote. To further improve model performance, a revision algorithm based on the confidence measures given by CSN is designed utilizing the speaker alternation pattern (SAP) in two-party dialogues. Experimental results show that our proposed method achieved an accuracy of 82.5% on the *World of Plainess* dataset which outperformed the machine-learning-based baseline method using manual features by 12% [1].

The main contributions of this paper are twofold. First, we build a candidate scoring network (CSN) based on BERT to tackle the problem of speaker identification in novels. Second, we design a revision algorithm utilizing the confidence measures given by CSN and the speaker alternation pattern in two-party dialogues.

2. Methodology

2.1. Task Definition

In our task, an instance for speaker identification is a textual segment consisting of a number of sentences. We denote it as $I = s_{-ws} \oplus \cdots \oplus s_{-1} \oplus qs \oplus s_1 \oplus \cdots \oplus s_{ws}$ and $\oplus$ is the concatenation operation. qs is the quote sentence whose speaker needs to be identified. $\{s_{-ws}, \cdots, s_{-1}\}$ are the ws context sentences on the left of qs, and $\{s_1, \cdots, s_{ws}\}$ are the ones on the right. ws is the single-sided context window size.

In addition to instances, a name list is provided which contains the characters that occur throughout the novel and a varying number of aliases for each character. For each instance in the dataset, the true speaker of the quote sentence has been

[1]The source code of this paper can be obtained from https://github.com/YueChenkkk/CSN-SAPR

http://dx.doi.org/10.21437/Interspeech.2021-609

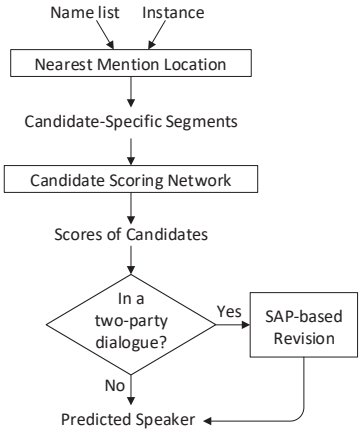

Figure 1: *The flowchart of proposed method, in which "SAP" stands for "speaker alternation pattern".*

annotated manually. Further, we assume that at least one alias of the true speaker in the name list should appear within the context sentences of the instance. Otherwise, the instance should be discarded from the dataset.

Given these prerequisites, our task is to determine the speaker of the quote sentence in an instance given the name list of the novel.

2.2. Framework of Proposed Method

The flowchart of our proposed method is shown in Fig. 1 which consists of three main steps. First, an instance is sent into the nearest mention location (NML) module together with the name list to determine a candidate set for this instance and to extract a candidate-specific segment from the instance for each candidate. Second, each candidate-specific segment is fed into the candidate scoring network (CSN) to produce its score. At last, a speaker alternation pattern (SAP)-based revision is applied to the quote sentences that belong to two-party conversations before finally determining the predicted speakers. The details of these three steps will be introduced in the following subsections.

2.3. Nearest Mention Location (NML)

This module first generates a set of candidate speakers for the input instance. For each character in the name list, if any one of its aliases appears within the context sentences of the instance, this character is added to the candidate set.

For each candidate speaker, there may be more than one occurrences (i.e., mentions) of its aliases. Intuitively, we assume that the mention nearest to the quote sentence carries the most evidence about whether the candidate is the true speaker. Therefore, we locate the *nearest mention* for each candidate which has the least number of words between itself and the quote sentence. Assuming that the nearest mention locates in the context sentence s_{nm} for a candidate speaker, its *candidate-specific segment* (CSS) is defined as

$$CSS = \begin{cases} s_{nm} \oplus \cdots \oplus s_{-1} \oplus qs, & nm < 0, \\ qs \oplus s_1 \oplus \cdots \oplus s_{nm}, & nm > 0. \end{cases} \quad (1)$$

The purpose of extracting candidate-specific segments is to exclude redundant context sentences which may not contribute to judge the correctness of a candidate speaker. In addition to candidate-specific segment, we also define *candidate-specific context* as $CSS \backslash qs$, i.e., removing quote from the candidate-specific segment.

2.4. Candidate Scoring Network (CSN)

Since the number of candidates for an instance varies, we regard speaker identification as a scoring task instead of a standard classification task. That is, a candidate scoring network (CSN) is built which assigns a score for every candidate speaker of an instance, and the candidate with the highest score is the result.

The structure of our proposed CSN is shown in Fig. 2. We first adopt the widely used pretrained language model, BERT [18], to generate contextualized representations. At the encoding step, we feed the candidate-specific segment into BERT and obtain a sequence of hidden states with the same length as the input token sequence. The hidden states corresponding to the nearest candidate mention, the quote sentence, and the candidate-specific context are denoted as $\boldsymbol{H}_{ncm}$, $\boldsymbol{H}_{qs}$ and $\boldsymbol{H}_{csc}$ respectively. Then, each of $\boldsymbol{H}_{ncm}$, $\boldsymbol{H}_{qs}$ and $\boldsymbol{H}_{csc}$ goes through a max-pooling layer and produces the embedding vector for candidate representation, quote sentence representation, and context representation respectively. These three fixed-length representations are concatenated to form the feature vector of this candidate. The feature vector is then fed into an MLP scorer with *tanh* output activation to produce a score within (-1, 1).

At the training stage, a margin ranking loss is adopted since our goal is to assign high scores to true speakers and low scores to distractors. For an instance *I*, its true speaker c_i is paired with another candidate c_j to form a positive-negative example pair. Then, the loss of this pair is calculated as

$$L(I, c_i, c_j) = max\{0, csn(I, c_j) - csn(I, c_i) + mgn\}, \quad (2)$$

where $csn(I, c)$ denotes the score of candidate *c* calculated by the CSN and mgn is a hyper-parameter that controls the ideal score margin between the two candidates. During training, the parameters in CSN are optimized to minimize the overall loss on the training set.

2.5. Speaker-Alternation-Pattern-Based Revision

Continuous multi-turn conversations are common in novels and most of them occur between two speakers. However, the context sentences of a quote in a conversation may seriously overlap with other quotes in the conversation, and thus the CSN may not be able to learn useful context representations for scoring candidates. Therefore, this module aims to revise the speaker identification results of CSN for two-party conversations utilizing the speaker alternation pattern (SAP). SAP means that the speaker of the n^{th} utterance in a two-party conversation is usually the speaker of the $(n+2)^{\text{th}}$ utterance, but is not the speaker of the $(n+1)^{\text{th}}$ utterance [6,9,20]. Based on this pattern, once the speaker of an utterance in the dialogue is determined, the speakers of the rest utterances can be easily inferred.

In this paper, we automatically detect two-party conversations in our dataset following two conditions. First, a conversation should be composed of continuous quote sentences without interrupting context sentences. Let $qs_1 \oplus \cdots \oplus qs_M$ denote a conversation with M quotes and $M \geq 2$. Second, a two-party conversation should have two dominant candidate speakers. Let c_n denote the candidate speaker which has the n-th highest mention frequency f_n in the $2ws$ context sentences of the conversation. This conversation is considered as a two-party one if $f_2 \geq f_3 + th$, where th is a pre-set threshold. Thus, c_1 and c_2 become the two candidate speakers of all quotes in this

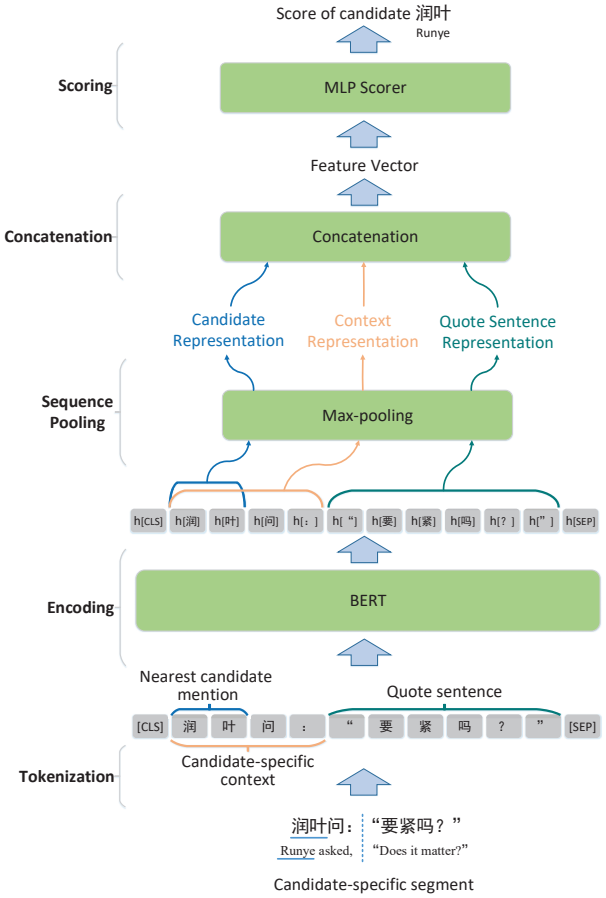

Figure 2: *The structure of our proposed candidate scoring network (CSN). In the example of input candidate-specific segment* ($nm = -1$)*, the vertical dotted line indicates the boundary between its context and its quote sentence, and the nearest candidate mention is underlined.*

conversation.

At the test stage, if the quote of an instance is in a two-party conversation, SAP-based revision is applied. Its basic idea is to first determine the speaker of the first quote or the last quote in a two-party conversation based on their confidence measures considering that the context sentences of these two quotes have the least overlap with other quote sentences. For the m-th quote in a conversation, its confidence measure is calculated as

$$qs_m.cm = |csn(I_m, c_1) - csn(I_m, c_2)|, \quad (3)$$

where I_m is the instance with qs_m as the quote sentence, c_1 and c_2 are the two candidate speakers of the conversation. A higher confidence measure indicates more reliable scores given by the CSN. Then, the speakers of the other quotes in the conversation are determined one by one according to SAP. The complete SAP-based revision algorithm is shown in Algorithm 1.

Algorithm 1 SAP-based revision

Input: a conversation of M quotes $qs_1 \oplus \cdots \oplus qs_M$, candidate speakers c_1 and c_2.

$qs_1.cm = |csn(I_1, c_1) - csn(I_1, c_2)|$;
$qs_M.cm = |csn(I_M, c_1) - csn(I_M, c_2)|$;
if $qs_1.cm > qs_M.cm$ **then**
  $qs_1.speaker = \arg\max_{c \in \{c_1, c_2\}} csn(I_1, c)$;
  for $i = 2; i \leq M; i++$ **do**
    if $qs_{i-1}.speaker = c_1$; **then**
      $qs_i.speaker = c_2$;
    else
      $qs_i.speaker = c_1$;
    end if
  end for
else
  $qs_M.speaker = \arg\max_{c \in \{c_1, c_2\}} csn(I_M, c)$;
  for $i = M - 1; i \geq 1; i--$ **do**
    if $qs_{i+1}.speaker = c_1$; **then**
      $qs_i.speaker = c_2$;
    else
      $qs_i.speaker = c_1$;
    end if
  end for
end if

3. Experiments

3.1. Dataset

The dataset built by Chen et al. [10] was adopted. This dataset was constructed based on the famous Chinese novel *World of Plainness* with human annotations. Its context window size (i.e., ws) was 10. The name list was collected manually and contained 125 roles that occurred throughout the novel, and each role had a varying number of 1-5 aliases. We extended the original dataset by making additional annotations and obtained 2596 instances in total[2]. We kept their original temporal order in the novel, and divided them into a training set with 2000 instances, a development set with 298 instances, and a test set with 298 instances. Following the taxonomy described in Chen's paper [10], we further divided the subsets into 3 categories named *explicit*, *implicit* and *latent* respectively. For a brief introduction, an instance in *explicit* or *implicit* can find the subject of a speech verb within the adjacent two sentences of the center quote. If that subject is a mention of a candidate in the name list, the instance belongs to *explicit*. If the subject is a pronoun, the instance belongs to *implicit*. The *latent* category holds instances that belong to neither *explicit* nor *implicit*. The numbers of instances in all categories are listed in Table 1.

Table 1: *The numbers of instances in our dataset.*

	explicit	*implicit*	*latent*	Total
Training	393	220	1387	2000
Development	44	31	223	298
Test	44	29	225	298

3.2. Settings

The Chinese *BERT-base* model released by Google Research[3] was adopted to build our CSN. For a text input, BERT output a contextualized representation and the embedding size was 768. The MLP scorer in CSN had a hidden layer of 100 units with $tanh$ activation. The margin in Eq. (2) was set as mgn=1.0. The mention frequency threshold in SAP-based revision was

[2] https://github.com/YueChenkkk/Chinese-Dataset-Speaker-Identification

[3] https://github.com/google-research/bert

Table 2: *Speaker identification accuracies of random guess, the Manual+MLP baseline method, our proposed method, and two ablated methods. SAPR means SAP-based revision, and CSS means candidate-specific segments.*

	explicit	*implicit*	*latent*	*ALL*
Random Guess	0.386	0.517	0.356	0.376
Manual+MLP [10]	0.886	0.517	0.693	0.705
Proposed	**0.932**	**0.759**	**0.813**	**0.825**
- SAPR	**0.932**	**0.759**	0.760	0.785
- SAPR & CSS	0.886	0.586	0.622	0.658

set as th=2. We adopted Adam optimizer [21] for training. The learning rate and batch size were set as 16 and 2e-5. The optimal iteration was determined based on the development set performance.

3.3. Experimental Results

The speaker identification accuracies of our proposed method on the test instances of three categories are shown in the third row of Table 2. In this table, *Random Guess* means selecting the speaker from candidates randomly. *Manual+MLP* is the baseline method described in Chen et al. [10] which was based on manually-designed linguistic features and an MLP scorer. We can see that our proposed method outperformed the baseline on all three categories, especially on the *implicit* and *latent* categories. The overall accuracy of our proposed method was higher than the baseline by 12%. These results demonstrate the superiority of the deep-learning-based approach over manual feature engineering on this task.

Ablation studies were conducted to further analyze the effectiveness of the modules in our proposed method. The first ablation was to remove the SAP-based revision and the results are shown in the fourth line of Table 2. By comparing the third and the fourth rows, we can see that this ablation didn't affect the performance of *explicit* and *implicit* categories, and degraded the accuracy of the $latent$ category. The reason is that the quote sentences of *explicit* and *implicit* instances in two-party conversations were usually the first or last quotes, thus they were not aimed to be revised. Actually, there were 20 *latent* instances in the test set which changed their decisions after the SAP-based revision. Among them, 16 revisions amended the wrong decisions of CSN, while 4 instances were incorrectly revised given the correct decisions of CSN. Furthermore, Table 3 shows the average confidence measures given by CSN for the test quotes at different positions in two-party conversations. It can be seen that the first and last quotes in conversations had significantly higher confidence measures than the other quotes (p=0.0027 in t-test). This is consistent with our assumption for designing the SAP-based revision algorithm that the quotes in the middle of conversations may not have as reliable scores as the first and the last quotes in conversations.

The second ablation further discarded candidate-specific segments at the encoding step, which means it directly fed the whole instance into BERT to derive the representations of candidate, context, and quote sentence. We name it *candidate-irrelevant encoding*. After this ablation, the average character-level length of BERT input increased from 111.7 to 534.4. The results of this ablated method are shown in the last line of Table 2. Comparing the results in the last two lines, we can see that using candidate-specific segments played an important role in our proposed method. The accuracies of the method with candidate-irrelevant encoding were even worse than the Manual+MLP baseline. We also tried to shorten the width of the context window for candidate-irrelevant encoding by reducing ws from 10 to 2. In this way, the average character-level length of BERT input declined from 534.4 to 125.7 which was close to that of using candidate-specific segments (111.7). For each instance, its candidate set was the same as that of ws=10. If a candidate didn't have any mention in the truncated context window, a score of zero was assigned to him or her by default. Finally, this approach yielded an overall accuracy of 0.698 without SAP-based revision, which was still much lower than that of using candidate-specific segments. These results indicate the necessity of filtering out redundant context information and modeling different context sentences for scoring different candidates in our neural-network-based approach.

Table 3: *Average confidence measures (Ave. CM) given by CSN for the first and last quotes in two-party conversations and for the other quotes in two-party conversations. In the brackets are standard deviations.*

	First & Last	Others
Ave. CM	1.384(±0.745)	0.826(±0.715)

…处罚金光亮[1]的事定下来之后，副支书金俊山[2]顺便提起了孙玉厚[3]在分给个人的责任田里栽树的问题。他婉言对玉亭[4]说：“…” …

...After the punishment of Jin Guangliang[1] was settled, the deputy party secretary Jin Junshan[2] incidently mentioned the problem of Sun Yuhou[3] planting trees in the public field. He said to Yuting[4] mildly, “...” ...

Figure 3: *A test instance and its English translation. The nearest mentions of its 4 candidates are underlined and numbered. The quote and other context sentences are omitted.*

3.4. Case Study

Fig. 3 presents a test instance with the nearest mentions of its 4 candidates. For this instance, CSN correctly predicted the speaker “Jin Junshan” without the help of SAPR, while the Manual+MLP baseline mistook “Yuting” for the speaker. One possible reason is that the baseline paid too much attention to the distance between the mention and the quote sentence which was a manually designed feature in this model. In contrast, CSN excelled in processing the input text in a comprehensive way owing to its raw token inputs and deep model architecture, and successfully solved the true speaker for this instance.

4. Conclusion

In this paper, we have proposed a deep-learning-based method to tackle the problem of identifying speakers in novels. Compared with conventional methods using manually designed linguistic features, our BERT-based model can utilize textual inputs more effectively and lead to better accuracy of speaker identification. In the future, more sophisticated mechanisms will be explored to model the interaction between the quote and its context. Besides, the approaches based on other techniques such as speaker modeling in dialogues [22, 23] and social network extraction [4, 5] are also worth further investigation.

5. References

[1] Jason Y Zhang, Alan W Black, and Richard Sproat. Identifying speakers in children's stories for speech synthesis. In *Eighth European Conference on Speech Communication and Technology*, 2003.

[2] Erica Greene, Taniya Mishra, Patrick Haffner, and Alistair Conkie. Predicting character-appropriate voices for a tts-based storyteller system. In *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[3] Cristian Danescu-Niculescu-Mizil and Lillian Lee. Chameleons in imagined conversations: a new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the 2nd Workshop on Cognitive Modeling and Computational Linguistics*, pages 76–87, 2011.

[4] Apoorv Agarwal, Anup Kotalwar, and Owen Rambow. Automatic extraction of social networks from literary text: A case study on alice in wonderland. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1202–1208, 2013.

[5] Snigdha Chaturvedi, Mohit Iyyer, and Hal Daume III. Unsupervised learning of evolving relationships between literary characters. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[6] Kevin Glass and Shaun Bangay. A naive salience-based method for speaker identification in fiction books. In *Proceedings of the 18th Annual Symposium of the Pattern Recognition Association of South Africa (PRASA'07)*, pages 1–6, 2007.

[7] Grace Muzny, Michael Fang, Angel Chang, and Dan Jurafsky. A two-stage sieve approach for quote attribution. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 460–470, 2017.

[8] David Elson and Kathleen McKeown. Automatic attribution of quoted speech in literary narrative. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, 2010.

[9] Hua He, Denilson Barbosa, and Grzegorz Kondrak. Identification of speakers in novels. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1312–1320, 2013.

[10] Jia-Xiang Chen, Zhen-Hua Ling, and Li-Rong Dai. A chinese dataset for identifying speakers in novels. In *INTERSPEECH*, pages 1561–1565, 2019.

[11] Yuxiang Jia, Huayi Dou, Shuai Cao, and Hongying Zan. Speaker identification and its application to social network construction for chinese novels. In *2020 International Conference on Asian Language Processing (IALP)*, pages 13–18. IEEE, 2020.

[12] Marti A. Hearst, Susan T Dumais, Edgar Osuna, John Platt, and Bernhard Scholkopf. Support vector machines. *IEEE Intelligent Systems and their applications*, 13(4):18–28, 1998.

[13] Leonardo Noriega. Multilayer perceptron tutorial. *School of Computing. Staffordshire University*, 2005.

[14] Danqi Chen and Christopher D Manning. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750, 2014.

[15] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, 2016.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Yann Lecun and Yoshua Bengio. Convolutional networks for images, speech, and time-series. In *The handbook of brain theory and neural networks*. MIT Press, 1995.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[19] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. 2018.

[20] Adam Ek, Mats Wirén, Robert Östling, Kristina Nilsson Björkenstam, Gintarė Grigonytė, and Sofia Gustafson-Capková. Identifying speakers and addressees in dialogues extracted from literary fiction. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.

[22] Kaixin Ma, Catherine Xiao, and Jinho D Choi. Text-based speaker identification on multiparty dialogues using multi-document convolutional neural networks. In *Proceedings of ACL 2017, Student Research Workshop*, pages 49–55, 2017.

[23] Zhao Meng, Lili Mou, and Zhi Jin. Towards neural speaker modeling in multi-party conversation: The task, dataset, and models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.