ROBUST UNSTRUCTURED KNOWLEDGE ACCESS IN CONVERSATIONAL DIALOGUE WITH ASR ERRORS

Yik-Cheung Tam, Jiacheng Xu, Jiakai Zou, Zecheng Wang, Tinglong Liao, Shuhan Yuan*

NYU Shanghai
Department of Computer Science
1555 Century Avenue, Pudong New District, Shanghai, China 200122

ABSTRACT

Performance of spoken language understanding (SLU) can be degraded with automatic speech recognition (ASR) errors. We propose a novel approach to improve SLU robustness by randomly corrupting clean training text with an ASR error simulator, followed by self-correcting the errors and minimizing the target classification loss in a joint manner. In the proposed error simulator, we leverage confusion networks generated from an ASR decoder without human transcriptions to generate variety of error patterns for model training. We evaluate our approach on DSTC10 challenge targeted for knowledge-grounded task-oriented conversational dialogues with ASR errors. Experimental results show effectiveness of our proposed approach, boosting the knowledge-seeking turn detection (KTD) F1 significantly from 0.9433 to 0.9904. Knowledge cluster classification is boosted from 0.7924 to 0.9333 in Recall@1. After knowledge document re-ranking, our approach shows significant improvement in all knowledge selection metrics, from 0.7358 to 0.7806 in Recall@1, from 0.8301 to 0.9333 in Recall@5, and from 0.7798 to 0.8460 in MRR@5 (Mean Reciprocal Rank) on the test set. On the recent DSTC10 evaluation, our approach demonstrates significant improvement in knowledge selection, boosting Recall@1 from 0.495 to 0.7105 compared to the official baseline. Our source code is released in GitHub [1].

***Index Terms*—** ASR error simulation, joint error correction and classification, knowledge turn detection, knowledge selection, DSTC10

1. INTRODUCTION

Knowledge-grounded task-oriented dialogue modeling has drawn a lot of attention in recent years due to its practical applications. In addition to dialogue state tracking in conventional dialogue, users can ask questions about a referenced restaurant of a dialogue such as "does the hotel offer free wifi?", requiring accessing an external knowledge base for answers. Virtual personal assistants listen to users' speech input with disfluencies or barge-ins, giving additional challenge to robust language understanding. Although accuracy of automatic speech recognition (ASR) has been improved significantly, ASR is not perfect and generates mis-recognized tokens. According to the Dialogue System Technology Challenge (DSTC10 [1]), deep neural models [2] well-trained on clean written text are vulnerable to ASR errors. For instance, knowledge-seeking turn detection (KTD) F1 is dropped dramatically from 0.9911 to 0.7602 [2]. Therefore, robust approaches are needed to bridge the mismatch.

In this paper, we address the robustness of knowledge selection with ASR errors by training a deep neural network to self-correct erroneous inputs and perform knowledge cluster classification in a joint fashion. First, we generate random errors from a proposed error simulator and inject the errors with corrective labels into data batches during model training. Second, we introduce knowledge cluster classification to quickly identify a subset of relevant documents from topically consistent knowledge clusters. Together with entity linking and tracking, document candidates are further reduced for re-ranking and delivering knowledge selection results.

Our contributions are three-folded: First, we propose an effective ASR error simulator learned from word confusion networks without human transcribed data. This implies that an ASR error simulator can be created and updated automatically by decoding speech audios. Second, we formulate knowledge selection with ASR errors using a joint model to self-correct simulated errors and perform knowledge cluster classification. Third, we perform extensive experiments and show effectiveness of our approach on the DSTC10 challenge.

2. RELATED WORK

To overcome erroneous ASR utterances, many research have been focusing on ASR error detection [3, 4] and error correction as individual modules, trying to detect and correct the errors so that the processed text are fed to downstream language understanding modules trained with clean text. [5] employed warped language model, a variant of masked language model [6], to correct errors in ASR transcripts. [7] proposed an augmented transformer to take in phonetic and orthographic information for entity correction using an error simulator based on word N-gram confusion matrix. [8] leveraged aligned N-best lists as inputs to a deep model so that the model was optimized jointly for N-best re-ranking and language understanding. [9] proposed a soft-masked BERT to detect errors and mask out erroneous tokens with the mask-token embedding, followed by error correction in an end-to-end manner. These approaches have shown effectiveness but require high-quality human transcripts to be aligned with ASR transcripts to produce labels for model training. [10] flattened a word confusion network and used position embeddings and attention masks to encode the graph structure of a confusion network, trying to make use of word confusion information for robust language understanding.

*Affiliation: Tandon School of Engineering, New York University

[1] https://github.com/yctam/dstc10_track2_task2.git

[2] https://github.com/alexa/alexa-with-dstc10-track2-dataset/blob/main/task2/baseline/README.md

Inspired by ASR error simulation [7, 11, 12, 13] and masked language modeling [6], our approach addresses robust SLU in a joint modeling framework. Starting from clean dialogue, we employ an error simulator to randomly replace word tokens in an input utterance with noisy tokens during model training such that our model attempts to achieve two goals: 1) self-correct the noisy tokens; 2) use the "repaired" hidden states for classification. Unlike [7], our error simulator does not require manual transcription for aligned labels. Instead, we analyze confusion networks and learn what words are confusing with each other from the ASR viewpoint. We leverage word confusion pairs to learn how a word is converted into another word via letter-to-letter alignment (e.g "deliver" $\longrightarrow$ "delver" after replacing i by the deletion symbol *). Our error simulator does not "hard-code" the error patterns from the word confusion matrix but generates variety of erroneous tokens during online data batching, enforcing a deep model to correct these "negative" tokens during training. In summary, we learn a Bert model so that simulated errors are self-corrected via an language modeling (LM) head, while the model simultaneously learns how to classify an input utterance. From the masked LM viewpoint [6], instead of masking tokens or replacing tokens with unrelated random tokens, we randomly replace input tokens in clean text with erroneous tokens that are ASR-error aware.

3. PROPOSED APPROACH

Our proposal relies on an ASR error simulator to generate random error pattern followed by a joint model to correct errors and classify using LM and classification heads. Before knowledge selection, we introduce knowledge cluster classification to quickly identify relevant knowledge clusters.

3.1. ASR error simulation

Given speech audio, an ASR decoder decodes the audio and generates word lattices or N-best lists that are aligned into confusion networks [14]. A confusion network provides alternative word candidates at each word position. For instance, a confusion network representation "ok do they (delver | deliver | delo | over | lover | del | dolo)" gives a word confusion set {delver, deliver, delo, over, lover, del} where the words sound confusing from an ASR decoder viewpoint. Here, we do not know which word token is correct in a confusion set. For creating an ASR error simulator, we focus on capturing and re-generating error patterns from confusion sets. In particular, we enumerate all possible word pairs from a confusion set, giving "delver-deliver", "delver-delo",...,"del-dolo". We perform letter-to-letter alignment between each word pair so that position-dependent letter rewriting probabilities $Pr(t|s,i)$ are estimated based on co-occurrence statistics, where i denotes the position of a source letter s to be rewritten as letter t including the deletion symbol "*". To generate a random erroneous word from a clean word:

1. Randomly choose the number of edits uniformly from 1, 2 or 3.
2. Randomly choose a letter position i uniformly across the length of a word.
3. Randomly choose an error type with $Pr(replacement) = 0.9$ and $Pr(insertion) = 0.1$.
4. If the error type is replacement, draw a letter $t \sim Pr(t|s,i)$. Replace s with t in position i.
5. If the error type is insertion, draw a letter $t \sim Pr_*(t|s,i)$. Insert t at position $i+1$.
6. Repeat step 2 until the desired number of edits is achieved.

Table 1 shows samples of simulated errors from clean words. The simulated errors "look" similar to their clean words. The ability of simulating new errors is crucial to avoid overfitting a downstream SLU model to limited error patterns.

Table 1. Samples of ASR error simulation.

clean word	Simulated versions
'alcohols'	'alchols', 'alcohos', 'alcohls', 'alchos'
'alimentum'	'alimentm', 'alimetum', 'alimenum', 'alietum'
'wifi'	'wifr', 'wivi', 'wrfi', 'wifie'

3.2. Knowledge cluster classification with self-correction

Inspired by masked LM, we randomly pick two words from a clean user utterance in a dialogue session, and apply error simulation in Section 3.1 to replace clean tokens with erroneous ones. We didn't replace more than 2 words in an utterance since injecting too much noise may corrupt the semantics of an utterance badly. With the generated errors and the corresponding clean word, we feed an erroneous dialogue utterance into a deep model, e.g. Bert [6] and learn to detect and correct the errors using the LM head.

Using multi-task learning, our ultimate target is to train a robust SLU model to detect and correct errors, and to classify an utterance with a knowledge cluster label. Given joint model parameters $\Theta = \{W_{bert}, W_{lm}, W_{cls}\}$ where Θ denotes parameters of a pre-trained Bert model W_{bert}, the language modeling head W_{lm} and the classification head W_{cls} respectively, our joint loss function becomes:

$$L(\Theta) = L_{cls}(W_{bert}, W_{cls}) + \lambda \cdot L_{lm}(W_{bert}, W_{lm}) \quad (1)$$

where λ denotes the LM weight for correcting the ASR errors. When λ is set to zero, the model just trains with noise, ignoring the error-correction mechanism. Figure 1 shows the proposed system architecture with double heads.

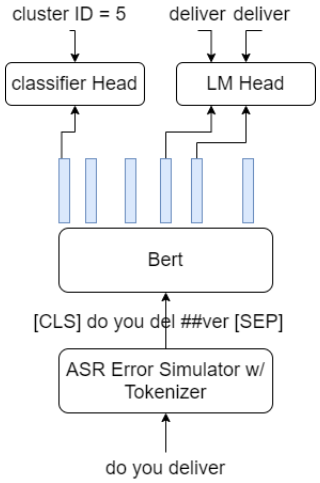

Fig. 1. Proposed system architecture with an ASR error simulator.

4. EXPERIMENTS

4.1. Data

We employed datasets from Dialogue System Technology Challenge, namely DSTC9 [15] and DSTC10 [1]. The challenge had 3

sub-tasks: 1) Knowledge-seeking turn detection (KTD) to determine if the last user turn had knowledge-seeking intention. For example, after a system returned a list of hotel recommendations that satisfy user's requirements, users may ask questions like: "does this hotel provide free wifi?" as positive knowledge seeking; 2) Knowledge-selection (KS) to search for a document from an external knowledge base to answer the question. In multi-turn dialogues, entity linking and tracking was required during searching for relevant documents. In this paper, we mainly reported results for KTD and KS. DSTC9 datasets contained clean dialogue while DSTC10 dataset had noisy dialogue with ASR errors. According to the track organizer, their ASR decoder in DSTC10 was based on wav2vec 2.0 [16] pre-trained on 960 hours of Librispeech and fine-tuned with 10% of in-domain speech data with word error rate of 24.09%. DSTC10 dataset came with N-best lists. We used SRILM toolkit [17] to generate confusion networks from N-best lists. Since only user utterances contained ASR errors while system utterances were clean, we simulated errors on user utterances only. Last-turn user utterances were considered for error simulation and model training. Table 2 shows samples of knowledge-seeking user utterances containing ASR errors and their clean versions.

Table 2. Samples of erroneous user utterances.

perfect and what is the check **kin** time for that
ok do they **delver** food
o ea do you know if that's a good place uh for **marge** groups
uh **taff ally** uh do they offer daily house keeping
perfect and what is the check in time for that
do they deliver food
by chance do you know if that's a good place uhhh for large groups
uhhh possibly uhhh do they offer daily house keeping

One difficulty of our work was that the provided DSTC10 dataset had only 263 dialogue sessions. For model tuning and evaluation, we split them into two equal parts. The first half was used as a development set while the second half was used as a test set.

To create our training set, we employed the DSTC9 evaluation set comprising only clean dialogue sessions. We didn't use the DSTC9 training and validation sets for our model training because they did not give any improvement. Since the DSTC10 dataset had overlapped dialogues with the DSTC9 evaluation set, we excluded the overlapped dialogues from training, yielding 3867 training dialogues. To enhance data coverage, we augmented titles from the knowledge base as the last "user-turn" into our training set, yielding 15906 training dialogues as our final training set.

4.2. Model training

For knowledge selection, an entity-document pair defines a target label. We observed that many entities in the knowledge base shared the same/similar titles. For example, "A and B Guest House" and "Acorn Guest house" share the same question regarding the "wifi" topic. We employed an iterative automatic algorithm to group questions of similar topics into knowledge clusters. Intuitively, questions with similar vocabulary (after removing stopwords and word normalization) formed initial knowledge clusters. Then we trained a supervised binary classification model to detect positive/negative question-answer pairs. Once a model was trained, we enumerated question-answer pairs from pairs of knowledge clusters which were classified by the model. Pairs of knowledge clusters were combined when majority of question-answer pairs were positively classified according to a threshold. We repeated the grouping step until no more knowledge clusters can be formed. To ensure good quality of knowledge clusters, we manually inspected and adjusted the knowledge clusters so that questions within each cluster were topically consistent. Additionally, we included a non-knowledge cluster containing user utterances that were non knowledge-seeking. In total, there were 95 knowledge clusters derived from 12039 question-answer pairs in the knowledge base covering hotel, restaurant, attraction, train and taxi domains. Table 3 shows samples of a knowledge cluster for "wifi" and "trail".

Table 3. Samples of knowledge clusters.

Are there any WiFi?
How much is the WiFi?
Do you offer free Wifi?
Does it offer free WiFi?
Do you have Wifi?
Hi! Does it offer any WiFi services?
What is the length of it?
How much time it will take to finish the guided tour?
How long is the trail?
How long does it take to complete a hike?
Is there any trail?

The motivation of using the knowledge clusters as labels was to avoid brute-force evaluation of the relevance between a last-turn user utterance and possible documents in the knowledge base. Therefore, we addressed knowledge selection in two steps. First, we predicted the knowledge cluster of the last user turn using the joint Bert classifier trained with error simulations and correction in Section 3.2. This step narrowed down the document candidates based on top-K predicted clusters. To pinpoint relevant documents further, we built entity-linking and entity-tracking systems based on techniques such as Trie-based and Bert-based named entity recognition, fuzzy match in Elasticsearch, and multiple-choice-headed [18] Bert for entity tracking. Due to limited space, we did not cover further details in this paper.

We trained Bert (bert-base-uncased) without ASR error simulation as our baseline. Then we trained joint models with different LM weights ranging from 0 (trained with noise without error correction), 0.1, 0.5, and 1.0. All models shared the same modeling architecture and were trained using the same hyper-parameters. We used the HuggingFace Transformers library [19] for pretrained models and the DSTC9 code base [3] for our implementation. We trained all models with 100 epochs with batch size of 16 on a single GPU. We used Adam [20] for optimization with learning rate of 6.25e-5 and ϵ of 1e-8. Gradient clipping was applied with maximum gradient norm of 1.0. Linear scheduling without warm-up was used. All random seeds were set to 42. For knowledge cluster classification, we only used the last user utterance since performance was hurt using all dialogue turns. The last user turn usually carried the most crucial information for knowledge cluster classification.

4.3. Results

Table 4 shows the knowledge cluster classification results using proposed approach with various LM weights on the development set. First, the proposed approach outperformed the baseline on all metrics. In particular, Recall@1 was boosted drastically from 0.8118 to 0.94 when the LM weights were set to 0.1 or 0.5. Tuning the LM weight was crucial as performance dropped LM weights were set to 0.0 and 1.0. It was surprising that simply adding noise without

[3] https://github.com/alexa/alexa-with-dstc9-track1-dataset

self-correction still improved performance. On the other hand, we observed that its model training would result in early stopping based on the optimal performance on the development set. In contrast, models took more steps to train when error correction was considered. This may be because the models took more effort to minimize the joint loss function with more variety of simulated errors. As a by-product of knowledge cluster classification, proposed model addressed knowledge-seeking turn detection (KTD) when the predicted label was either zero or non-zero. Consistently, KTD F1 was also improved from 0.9306 to 0.96 compared to the baseline.

Table 4. Knowledge cluster classification on the development set using the proposed approach with various LM weights. MRR: Mean Reciprocal Rank.

LM weight	F1 (KTD)	R@1	R@5	MRR@5	Steps
Baseline	0.9306	0.8118	0.8712	0.8333	14925
0.0	**0.96**	0.90	**0.94**	0.915	13930
0.1	**0.96**	**0.94**	**0.94**	**0.94**	84575
0.5	**0.96**	**0.94**	**0.94**	**0.94**	59700
1.0	0.94	0.92	0.92	0.92	32835

Table 5 shows the knowledge cluster classification results on the test set. We observed consistent improvement on all metrics. In particular, we observed a drastic improvement in Recall@1 from 0.7924 to 0.9333 when the optimal LM weight according to the development set. KTD F1 was also boosted significantly from 0.9433 to 0.9904. Performance trend on the development and test sets were similar, demonstrating the effectiveness of the proposed approach on noisy utterances with ASR errors.

Table 5. Knowledge cluster classification on the test set using the proposed approach with various LM weights.

LM weight	F1 (KTD)	R@1	R@5	MRR@5
Baseline	0.9433	0.7924	0.8301	0.8056
0.0	0.9714	0.8380	0.8571	0.8476
0.1	0.9714	0.8952	0.8952	0.8952
0.5	**0.9904**	**0.9333**	**0.9333**	**0.9333**
1.0	0.9411	0.9019	0.9019	0.9019

Finally, we combined the top-K knowledge cluster predictions with our multiple-choice-headed [18] Bert-based entity tracker, yielding pre-selected knowledge document candidates for further re-ranking. Re-ranking was crucial because knowledge clusters were coarse and may contain documents from different sub-topics as shown in Table 3. For each system, we fed their pre-selected knowledge document candidates into the DSTC9 winner system (Knover) for knowledge re-ranking [2]. Table 6 shows knowledge turn detection and knowledge selection results on the test set. Consistently, proposed approach showed improved performance across all metrics. In particular, our approach yielded 4.5%, 10% and 6.62% absolute improvement on Recall@1, Recall@5 and MRR@5 respectively compared to the baseline. We also compared to the case where our entity-tracker was disabled so that the Knover system performed knowledge re-ranking directly from documents in the predicted top-K knowledge clusters. However, the results were worse as shown in Table 6, showing that our entity-linking and entity tracker were effective to remove a lot of irrelevant entities from re-ranking. In a separated experiment, re-ranking with Knover was still effective compared to the un-ranked documents within the tracked top entities sorted by document IDs.

Table 7 shows the DSTC10 evaluation results with 1988 test dialogues. Our proposed approach shows effectiveness in all metrics using a single model. In particular, Recall@1 was boosted dramatically from 0.495 to 0.7105 compared to the Knover baseline [2] trained only on clean DSTC9 dialogues. Using unsupervised clusters (w/o manual adjustment) results in performance degradation.

Table 6. Knowledge selection result on the test set using the proposed approach with various LM weights after knowledge re-ranking using Knover [2]

LM weight	F1 (KTD)	R@1	R@5	MRR@5
Baseline	0.9433	0.7358	0.8301	0.7798
0.0	0.9714	0.7238	0.8571	0.7873
0.1	0.9714	0.7619	0.8952	0.8253
0.5	**0.9904**	**0.7809**	**0.9333**	**0.8460**
1.0	0.9411	0.7647	0.9019	0.8300
0.5 (w/o entity tracker)	**0.9904**	0.6666	0.8380	0.7333
0.5 (w/o Knover)	**0.9904**	0.7619	0.8952	0.8101

Table 7. DSTC10 evaluation results on knowledge selection using the proposed approach.

System	F1 (KTD)	R@1	R@5	MRR@5
Knover baseline [2]	0.7692	0.495	0.6472	0.5574
Proposed (Single)	0.8774	0.7105	0.7976	0.7493
Post-eval (Single)	**0.8786**	**0.7144**	**0.8032**	**0.7541**
Unsup knowledge clusters	0.8742	0.6979	0.7846	0.7369

5. DISCUSSION

For ASR error simulation, at first we replaced a clean word randomly by sampling a noisy word from the word confusion matrix. However, knowledge cluster classification results on the test set were much worse with Recall@1 only at 0.8118 compared to 0.9333. This may be due to the lack of ASR error coverage in the word confusion matrix to cope with unseen error patterns in tests.

Having a trainable ASR error simulator was attractive. We tried finetuning GPT2 [18] with word confusion pairs and generating words letter by letter using an LM head or a pointer generator head [21, 22]. However, none of these approaches could generate reasonable variants of new words. The lack of training data may be the cause. With more ASR decoded training data, we would further investigate it in the future.

The proposed error simulator is limited to editing a clean word to generate errors. Moreover, our simulator is hard to generate phonetically similar words or phrases such as "kommen toor" from "coit tower". It may be helpful to integrate phonetic dictionary [7] to improve the variety of the simulated errors.

Lastly, we will perform more experiments using the DSTC10 evaluation set after labels will be released. It would be interesting to evaluate our work on dialogue state tracking with ASR errors.

6. CONCLUSIONS

We have proposed ASR error simulation to improve knowledge selection in knowledge-grounded task oriented dialogues. Our approach does not require human transcription but leverage word confusion networks to build an error simulator and simulate variety of errors for robust model training. We have proposed a joint modeling approach using double heads to correct the simulated errors and perform knowledge cluster classification. Our proposed approach has shown significant improvement in knowledge-turn detection and knowledge selection on our test set and the DSTC10 evaluation set.

7. REFERENCES

[1] Seokhwan Kim, Yang Liu, Di Jin, Alexandros Papangelis, Behnam Hedayatnia, Karthik Gopalakrishnan, and Dilek Hakkani-Tur, "DSTC10 track proposal: Knowledge-grounded task-oriented dialogue modeling on spoken conversations," in *DSTC10 Workshop @ AAAI (to appear)*, 2022.

[2] Huang He, Hua Lu, Siqi Bao, Fan Wang, Hua Wu, Zheng-Yu Niu, and Haifeng Wang, "Learning to select external knowledge with multi-scale negative sampling," in *Proceedings of DSTC9 Workshop @ AAAI*, Feb. 2021.

[3] Yik-Cheung Tam, Yun Lei, Jing Zheng, and Wen Wang, "ASR error detection using recurrent neural network language model and complementary ASR," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 2312–2316.

[4] Jinyi Yang, Lucas Ondel, Vimal Manohar, and Hynek Hermansky, "Towards automatic methods to detect errors in transcriptions of speech recordings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 3747–3751.

[5] Mahdi Namazifar, John Malik, Li Erran Li, Gokhan Tur, and Dilek Hakkani Tür, "Correcting Automated and Manual Speech Transcription Errors Using Warped Language Models," in *Proc. Interspeech*, 2021, pp. 2037–2041.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT*, 2019, vol. 1, pp. 4171–4186.

[7] Haoyu Wang, Shuyan Dong, Yue Liu, James Logan, Ashish Agrawal, and Yang Liu, "ASR error correction with augmented transformer for entity retrieval," in *Proceedings of Interspeech*, Oct. 2020.

[8] Yue Weng, Sai Sumanth Miryala, Chandra Khatri, Runze Wang, Huaixiu Zheng, Piero Molino, Mahdi Namazifar, Alexandros Papangelis, Hugh Williams, Franziska Bell, and Gökhan Tür, "Joint contextual modeling for ASR correction and language understanding," in *Proceedings of ICASSP)*, May 2020, pp. 6349–6353.

[9] Shaohua Zhang, Haoran Huang, Jicong Liu, and Hang Li, "Spelling error correction with soft-masked BERT," in *Proceedings of ACL*, July 2020.

[10] Chen Liu, Su Zhu, Zijian Zhao, Ruisheng Cao, Lu Chen, and Kai Yu, "Jointly encoding word confusion network and dialogue context with BERT for spoken language understanding," in *Proceedings of Interspeech*, Oct. 2020.

[11] Edwin Simonnet, Sahar Ghannay, Nathalie Camelin, and Yannick Estève, "Simulating ASR errors for training SLU systems," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC)*, Miyazaki, Japan, May 2018, European Language Resources Association (ELRA).

[12] Rohit Voleti, Julie M. Liss, and Visar Berisha, "Investigating the effects of word substitution errors on sentence embeddings," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Brighton, United Kingdom, May 2019.

[13] César Montenegro, Roberto Santana, and Jose A. Lozano, "Analysis of the sensitivity of the end-of-turn detection task to errors generated by the automatic speech recognition process," *Engineering Applications of Artificial Intelligence*, vol. 100, pp. 104189, 2021.

[14] Lidia Mangu, Eric Brill, and Andreas Stolcke, "Finding consensus among words: Lattice-based word error minimization," in *Proceedings of EUROSPEECH*, 1999, pp. 495–498.

[15] Seokhwan Kim, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, and Dilek Hakkani-Tur, "Beyond domain apis: Task-oriented conversational modeling with unstructured knowledge access track in DSTC9," in *Proceedings of DSTC9 Workshop @ AAAI*, Feb. 2021.

[16] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proceedings of NeurIPS*, 2020.

[17] Andreas Stolcke, "SRILM - an extensible language modeling toolkit," in *Proceedings of Interspeech*, 2002.

[18] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," in *Technical report, OpenAI*, 2018.

[19] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of EMNLP: System Demonstrations*, Online, Oct. 2020, pp. 38–45, Association for Computational Linguistics.

[20] Diederik Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations*, 2015.

[21] Abigail See, Peter J. Liu, and Christopher D. Manning, "Get to the point: Summarization with pointer-generator networks," in *Proceedings of ACL*, 2017.

[22] Seppo Enarvi, Marilisa Amoia, Miguel Del-Agua Teba, Brian Delaney, Frank Diehl, Stefan Hahn, Kristina Harris, Liam McGrath, Yue Pan, Joel Pinto, Luca Rubini, Miguel Ruiz, Gagandeep Singh, Fabian Stemmer, Weiyi Sun, Paul Vozila, Thomas Lin, and Ranjani Ramamurthy, "Generating medical reports from patient-doctor conversations using sequence-to-sequence models," in *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, Online, July 2020, pp. 22–30, Association for Computational Linguistics.