INCORPORATING ASR ERRORS WITH ATTENTION-BASED, JOINTLY TRAINED RNN FOR INTENT DETECTION AND SLOT FILLING

Raphael Schumann

Heidelberg University
Institute for Computational Linguistics
INF 325, 69120 Heidelberg, Germany
rschuman@cl.uni-heidelberg.de

Pongtep Angkititrakul

Robert Bosch LLC
HMI-Conversational AI
384 Santa Trinita Ave., Sunnyvale, CA 94085
Pongtep.Angkititrakul@us.bosch.com

ABSTRACT

The real-world performance of slot filling and intent detection task generally degrades due to transcription errors generated by speech recognition engine. The insertion, deletion, and mis-recognition errors from speech recognizer's front-end cause the mis-interpretation and mis-alignment of the language understanding models. In this work, we propose a new jointly trained model of intent detection and slot filling with consideration of speech recognition errors. The attention-based encoder-decoder recurrent neural network first decodes the intent information from an utterance, and then corrects errors in the word sequence, if any, before extracting the slot information. The triple joint training framework maximizes the probability of a correct understanding given an input utterance. Our experimental results showed that the proposed model obtained 2.87% absolute gain over the joint model without ASR error correction for slot filling and 0.73% absolute error rate reduction for intent detection.

Index Terms— ASR Errors, Joint Training, Slot Filling, Intent Detection, Attention-based Model, Recurrent Neural Network

1. INTRODUCTION

Spoken Language Understanding (SLU) systems process language expressed by human speech into a semantic representation understandable by the machines. SLU is the key component of all conversational AI systems. The general tasks of SLU involve intent determination and slot filling from an utterance. The intent determination task can be considered as a semantic utterance classification problem, while the slot filling task can be tackled as a sequence labeling problem of contiguous words. Previous approaches to solving these two related tasks were typically proposed as two separated systems such as Support Vector Machines (SVMs) for intent determination [1] and Conditional Random Fields (CRFs) for slot filling [2].

Recent advances in neural networks, especially recurrent neural networks (RNNs), allow joint training model of both intent determination and slot filling [3, 4, 5, 6]. This framework showed advantages over the previous state-of-the-art techniques, and has gained much attention in research community. The success of joint models is contributed by the attention mechanism [7, 8, 9] and the encoder-decoder model [10, 11]. The attention mechanism allows optimize selection of input sequence for decoding for both content and location information.

In general, an SLU system is deployed as a downstream task of spoken dialog systems where its inputs are outputs from the front-end Automatic Speech Recognition (ASR) engine. The errors in word sequences generated by ASR engine cause the performance degradation of intent detection and slot filling. In most real-world applications (e.g, far field with noises and reverberation effect), such errors are still unavoidable even with deployment of more robust ASR techniques. In this work, we propose a model which combines error correction of ASR with joint intent detection and slot filling training. We start with a joint model proposed in [6] as our motivation and baseline, and then extend the framework with error correction of input word sequence. The attention-based encoder-decoder RNN first predicts user's intent from a recognized word sequence, corrects any recognized word errors from ASR engine, and then extracts slot contents from the predicted intent and context information.

We evaluated our proposed model on intent detection and slot filling tasks using degraded audio generated by texts from the ATIS corpus. We show that our purposed joint model can improve the F1 score of slot filling and reduce intent detection rate, as well as reduce the word error rate of ASR output. To the best of our knowledge, there has not been other work published on jointly tackling ASR error correction, intent detection, and slot filling.

This paper is organized as follows. In Section 2, we introduce the background architecture of RNN with encoder-decoder and attention mechanism. We describe our proposed jointly trained model in Section 3. In Section 4, we discuss the experimental setup and results. Finally, Section 5 concludes our paper.

2. BACKGROUND

2.1. RNN Encoder-Decoder

The recurrent neural network (*RNN*) *Encoder-Decoder* framework [10] [11] has been successfully applied in machine translation [7] and text summarization [12]. The main idea relies on a large network trained with an end-to-end fashion which make it generalized well with wide range of word sequences. The first component, *Encoder*, computes a representation c for each input sentences, and hence the second component, *Decoder*, generates one output word at a time based on the conditional probability of previous words and c. That is, the encoder represents input sequence $\mathbf{x} = (x_1, ..., x_{T_x})$ with a vector c:

$$h_t = f(x_t, h_{t-1}), \tag{1}$$

$$c = q(\{h_1, ..., h_{T_x}\}), \tag{2}$$

where h_t is a hidden state at time t. f, q are some nonlinear functions. The last hidden state h_{T_x} carries information of the complete input sequence and can be used as c [11]. Consequently, the decoder defines a probability of the output sequence $\mathbf{y} = (y_1, ..., y_{T_y})$ as:

$$p(\mathbf{y}) = \prod_{i=1}^{T_y} p(y_i | \{y_1, ..., y_{i-1}\}, c). \tag{3}$$

Note that T_x and T_y can be different, as there is no explicit alignment between input and output sequences. However, some problems like POS-tagging [13] require $T_x = T_y$ to have an exact alignment between input and output sequences.

2.2. Attention Mechanism

Attention mechanism was first introduced by [7], later modified by [14]. The attention concept allows models to learn the alignment or attention weights by focusing on the relevant elements of the input states h. In this work, unlike the conventional encoder-decoder model, the decoder computes a different context vector c_i for every decoding step i as the weighted sum over a sequence of input vectors $(h_1, ..., h_{T_x})$ [7]:

$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j. \tag{4}$$

The weight a_{ij} is computed using a feedforward neural network with h_j and the previous hidden state s_{i-1} as input. The decoder RNN hidden state s_i is calculated as:

$$s_i = f(s_{i-1}, y_{i-1}, c_i). \tag{5}$$

In this work, however, we employed *global* attention [14] where a variable-length alignment weight vector a_{ij} is inferred by the current target state h and all source states $\bar{h}$, as:

$$a_{ij} = \frac{\exp(\bar{h}_i^\top W_a h_j)}{\sum_{j'} \exp(\bar{h}_i^\top W_a h_{j'})}, \tag{6}$$

where the decoder hidden state $\bar{h}_i$ can be called query vector. Given $\bar{h}_i$ and the context vector c_i, the attentional hidden state $\widetilde{h}_i$ is computed as follows:

$$\widetilde{h}_i = \tanh(W_c[c_i, \bar{h}_i]). \tag{7}$$

The final output label is obtained by:

$$y_i = argmax(softmax(W_p \widetilde{h}_i)). \tag{8}$$

3. PROPOSED JOINTLY TRAINED MODEL

In this section, we describe our proposed model which is jointly trained for intent detection, word sequence correction, and slot filling. The underlying structure is based on RNN Encoder-Decoder framework [10, 11] with one decoder for each subtask. Similar to [7], the initial state of the decoders is computed as:

$$s_0 = tanh(W_s[fh_{T_x}, bh_1]). \tag{9}$$

3.1. Encoder

The encoder consists of a bidirectional RNN with LSTM cell as a basic RNN unit which reads forward and backward through the input word sequence $\mathbf{x}$. At each encoding time step $t = \{1, ..., T_x\}$, the forward and backward hidden states fh_t, bh_t are concatenated as the source hidden state $h_t = [fh_t, bh_t]$.

3.2. Intent Decoder

Intent detection is considered as a classification problem; therefore, the decoder produces a single label instead of a label sequence. This can be seen as a single step decoder RNN, where the only hidden state $\bar{h}$ is computed by $W_i s_0$. Eq. 4 and 7 are used to compute context vector c_i and attentional hidden state $\widetilde{h}$. The intent label y_i is predicted by Eq. 8.

3.3. Corrected-Word Decoder

The corrected-word decoder is a unidirectional RNN. At each decoding timestep i, the inputs are the embeddings of the predicted intent label y_i and the previously emitted word y_{i-1}^w. We used scheduled sampling [15] to mix them with the true labels during training. In conjunction with the previous state s_{i-1}, the current hidden state $\bar{h}_i$ of the cell is computed and used to generate context vector c_i^w and subsequently attentional hidden state $\widetilde{h}_i$ using Eq. 4 and 7. The decoding stops once Eq. 8 predicts the <EOS> token, which defines the last decoding timestep T_y. In order to get a richer representation

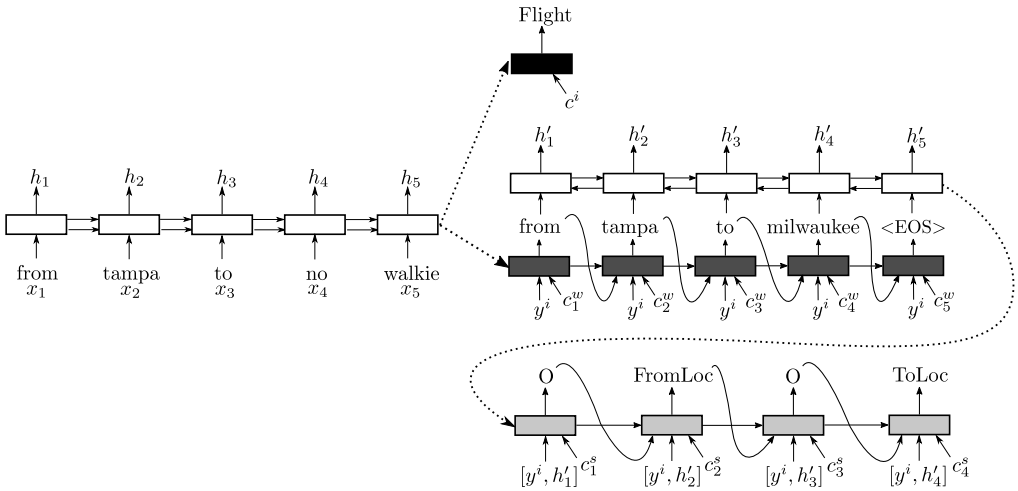

Fig. 1. Attention-based RNN for jointly trained intent detection, ASR correction, and slot filling model. The encoder uses bidirectional RNN. The intent decoder is a single step decoder RNN. The word decoder is a unidirectional RNN with the context information from parts of the input sequence. The slot decoder is a unidirectional RNN with predicted intent label and hidden state from the word decoder as inputs.

Input:

words	show	flights	from	boston	to	no	work

Labels:

intent	flight							
words	show	me	flights	from	boston	to	new	york
slots	O	O	O	O	B-fromloc .city_name	O	B-toloc .city_name	I-toloc .city_name

Fig. 2. Example instance from the extended ATIS dataset. Chunks are marked by bold boxes.

of the predicted word output sequence y^w, it is fed to the same encoder structure as the input sequence. h' are the newly produced encoder hidden states.

3.4. Slot Decoder

The initial state s_0 of the unidirectional slot decoder RNN is computed by Eq. 9 using fh'_{T_y} and bh'_1. The inputs at each timestep $i = \{1, ..., T_Y - 1\}$ are the predicted intent label y^i embedding, previously emitted slot label y^s_{i-1} embedding, and the hidden state h'_i. Again scheduled sampling[15] of the slot labels is used to improve training. Current slot decoder hidden state $\bar{h}_i$ is used to calculate c^s_i and subsequently $\widetilde{h}_i$ following Eq. 4 and 7. To get the output slot sequence y^s, Eq. 8 is applied at each time step i.

4. EXPERIMENTS

4.1. Data

In order to evaluate our model, we start with the widely used ATIS (Airline Travel Information Systems) dataset [16], and

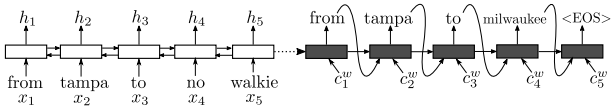

Fig. 3. ASR correction model learns to correct ASR errors. Here, the encoder-decoder model learns to correct "from tampa to no walkie" as "from tampa to milwaukee <EOS>".

modify it by adding ASR hypotheses. The original dataset has a training set of 4978 sentences from which we randomly sampled 893 sentences for development set. The provided test set also counts 893 utterances. For each sentence, there is one of 18 different intent labels and a sequence of slots as shown in Fig. 2. The 128 unique slot tokens follow the in/out/begin (IOB) schema [17]. To produce ASR hypotheses, we created audio for each sentence using Google Text to Speech (gTTS) API[1]. In this study, in order to degrade the audio quality such that the ASR errors started contributing to the overall performance, we added white noise and reverberation to the synthesized speech using the *SoX* tool. The recognized text from the ASR engine is then fed to the NLU model. We selected the top-3 hypotheses and add them as new instances to the corresponding set. The intent and slot output sequence are taken from the initial instance. Additionally the correct transcription is added as the word output sequence for the initial and the newly created instances. This results in a training, development, and test set of size 11841, 2583 and 2606 respectively. Then number of unique words increases from 950 words to 3178 words. The word error rate (WER) between input and output word sequence for training, development, and

[1] https://github.com/pndurette/gTTS

Models	WER (%)	Slot (F1)	Intent Error (%)
Joint Slot&Detection	14.55	84.26	5.80
ASR Correction + Joint Slot&Detection	10.43	86.85	5.20
Proposed Joint Model	10.55	87.13	5.04

Table 1. Experimental results on the extended ATIS dataset.

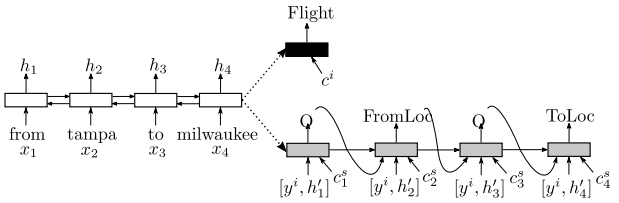

Fig. 4. Attention-based encoder-decoder RNN model for joint intent detection and slot filling.

test set is 13.80%, 13.64%, and 14.55% respectively.

4.2. Baseline

We compared our proposed jointly trained model with two other serial models. The first model, as shown in Fig. 3, solely learns how to predict the correct word sequence from the given ASR hypotheses. The model is based on the Encoder-Decoder framework previously described in Sec. 2.1. While the encoder (Sec. 3.1) is identical to the one used in the joint model, the decoder differs from the previously described word decoder (Sec. 3.3) as it cannot take into account the predicted intent label.

The model for intent detection and slot filling, as shown in Fig. 4 is a modification of the *Encoder-Decoder RNN* with aligned inputs & attention model proposed by [6]. The authors showed that this model can achieve the state of the art performance on the intent detection and slot filling. In our proposed model, the slot decoder also takes the predicted intent label as an input of RNN.

4.3. Model Training

We employed LSTM cell for all RNN units [18]. The scheduled sampling probability is set to 0.5 and parameters are trained by Adam optimization [19]. We utilized scikit-optimize (skopt[2]) to efficiently search for the best combination of hyperparameters. Word embedding of size 172 were randomly initialized and shared between encoder and decoders with the slot embedding size is 38, and the intent embedding size is 19. The mini-batch training size is 128, with learning rate of 0.004 and dropout rate of 0.55 applied to the non-recurrent connections. In the joint model the weights of the input encoder are shared with the one applied to the predicted word sequence.

[2]https://scikit-optimize.github.io

4.4. Evaluation Metrics

Quality of the predicted word sequence is quantified by word error rate (WER). Intent detection is evaluated by classification error rate, that is, the percentage of wrongly predicted intent labels. Slot labeling performance is measured by F1-score, similar as described in [20] for the CoNLL Chunking Shared Task. Following the in/out/begin (IOB) schema [17] a chunk begins with a 'B-X' token and may be extended by an 'I-X' token. The 'O' token tags words outside of any chunk. The same chunk tagged to the same word in the gold and predicted sequence is considered a true positive.

4.5. Results

Table 1 compares the F1 scores for slot filling and error rate of intent detection among 3 RNN models. After degrading the audio quality, the ASR engine generates the transcriptions with 14.55% WER. We started our experiments with this WER level as to reflect moderate ASR performance. Our proposed jointly trained model (Intent + Word Correction + Slot) outperforms the joint model (Intent + Slot) for both F1 and detection errors by absolute 2.87% and 0.73%, respectively. The proposed framework can also significantly reduce WER of the ASR engine by 4% absolute.

To further support the benefits of the proposed joint training framework, we compare the proposed joint model with the cascade of individual model of ASR correction followed by jointly trained intent+slot model. The individual trained model for ASR correction provides lower WER than the proposed model as the main objective is solely to reduce the ASR errors. However, the experimental results show consistent improvement of proposed joint model over the cascade of individual trained models for both slot filling and intent detection.

5. CONCLUSION

In this paper, we proposed jointly trained model of intent detection, ASR error correction, and slot filling. The attention-based encoder-decoder RNN first detects intent of a sentence, corrects any recognized word error from ASR engine, and then extract slot contents from the predicted intent and context information. The experimental results showed advantage of the proposed jointly-trained model over the joint model without taking into account the ASR errors and those individual trained models.

6. REFERENCES

[1] P. Haffner, G. Tur, and J. H. Wright, "Optimizing svms for complex call classification," in *Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03). 2003 IEEE International Conference on*, April 2003, vol. 1, pp. I–632–I–635 vol.1.

[2] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, San Francisco, CA, USA, 2001, ICML '01, pp. 282–289, Morgan Kaufmann Publishers Inc.

[3] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, and G. Zweig, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, March 2015.

[4] D. Guo, G. Tur, W. t. Yih, and G. Zweig, "Joint semantic utterance classification and slot filling with recursive neural networks," in *2014 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2014, pp. 554–559.

[5] P. Xu and R. Sarikaya, "Convolutional neural network based triangular crf for joint intent detection and slot filling," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec 2013, pp. 78–83.

[6] Bing Liu and Ian Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," *CoRR*, vol. abs/1609.01454, 2016.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *CoRR*, vol. abs/1409.0473, 2014.

[8] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," *CoRR*, vol. abs/1506.07503, 2015.

[9] Feng Wang and David M. J. Tax, "Survey on the attention based RNN model and its applications in computer vision," *CoRR*, vol. abs/1601.06823, 2016.

[10] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, vol. abs/1406.1078, 2014.

[11] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le, "Sequence to sequence learning with neural networks," *CoRR*, vol. abs/1409.3215, 2014.

[12] Ramesh Nallapati, Bing Xiang, and Bowen Zhou, "Sequence-to-sequence rnns for text summarization," *CoRR*, vol. abs/1602.06023, 2016.

[13] Laurent Besacier Yin-Lai Yeong Keng Hoon Gan Tien-Ping Tan, Bali Ranaivo-Malanon and Enya Kong Tang, "Evaluating lstm networks, hmm and wfst in malay part-of-speech tagging," *JTEC*, vol. 9, pp. 79–83, 2017.

[14] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning, "Effective approaches to attention-based neural machine translation," *CoRR*, vol. abs/1508.04025, 2015.

[15] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *CoRR*, vol. abs/1506.03099, 2015.

[16] Charles T. Hemphill, John J. Godfrey, and George R. Doddington, "The atis spoken language systems pilot corpus," in *Proceedings of the Workshop on Speech and Natural Language*, Stroudsburg, PA, USA, 1990, HLT '90, pp. 96–101, Association for Computational Linguistics.

[17] Lance A. Ramshaw and Mitchell P. Marcus, "Text chunking using transformation-based learning," *CoRR*, vol. cmp-lg/9505040, 1995.

[18] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[19] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[20] Erik F. Tjong Kim Sang and Sabine Buchholz, "Introduction to the conll-2000 shared task: Chunking," in *Proceedings of the 2Nd Workshop on Learning Language in Logic and the 4th Conference on Computational Natural Language Learning - Volume 7*, Stroudsburg, PA, USA, 2000, ConLL '00, pp. 127–132, Association for Computational Linguistics.