Interspeech 2018
2-6 September 2018, Hyderabad, India

# Robust Spoken Language Understanding via Paraphrasing

*Avik Ray, Yilin Shen, and Hongxia Jin*

Samsung Research America, Mountain View, California, USA
{avik.r, yilin.shen, hongxia.jin}@samsung.com

## Abstract

Learning intents and slot labels from user utterances is a fundamental step in all spoken language understanding (SLU) and dialog systems. State-of-the-art neural network based methods, after deployment, often suffer from performance degradation on encountering paraphrased utterances, and out-of-vocabulary words, rarely observed in their training set. We address this challenging problem by introducing a novel paraphrasing based SLU model which can be integrated with any existing SLU model in order to improve their overall performance. We propose two new paraphrase generators using RNN and sequence-to-sequence based neural networks, which are suitable for our application. Our experiments on existing benchmark and in house datasets demonstrate the robustness of our models to rare and complex paraphrased utterances, even under adversarial test distributions.

**Index Terms**: spoken language understanding, paraphrase, neural network

## 1. Introduction

Voice controlled personal agents (e.g. Alexa, Google Assistant, Bixby) are becoming popular due to their ability to understand a wide variety of user utterances, and perform different actions/tasks as requested by the user. Spoken language understanding (SLU) unit, or a semantic parser lie at its core which enables the agent to map a user utterance to the corresponding action desired by the user. Commercial semantic parsers represent the meaning of an utterance in terms of *intent* and *slot* labels, which can then be mapped to an action. *Intent detection* refers to the sub task of classifying an utterance into a semantic intent label, where as *slot tagging* is the sub task of providing a slot label to each word in the utterance.

Traditional approaches treat intent detection as a semantic classification problem, and slot tagging as a sequence labeling problem. A wide variety of algorithms have been proposed e.g. SVMs [1], hidden Markov models [2], CRFs [3], and more recently neural networks [4, 5, 6, 7]. State-of-the-art deep neural network models are trained jointly to solve the two tasks simultaneously using recurrent and sequence-to-sequence networks [6, 7, 8, 9]. These models are trained end to end using labeled training data in the form of (utterance, intent label, slot labels) tuple. However, such datasets are expensive to collect, and are never exhaustive. As a result, after deployment, these data driven models suffer from poor accuracy on utterances which occur infrequently in their training data e.g. utterances with out-of-vocabulary words as well as various sentential paraphrases of the training utterances. The fundamental difficulty stems due to shortcoming of these models trained using likelihood maximization objective, that they do not generalize well to rare examples in training data. Unfortunately, this occur often in personal agent applications, since each individual user has their own personal vocabulary and paraphrase preferences.

In this work, we try to tackle this problem by making the following important observation; often these infrequent and personalized user utterances have a paraphrased utterance which is more frequent in the training data. We try to answer the question; *instead of building a parser which perform well even for infrequent utterances, can we simply map such utterances to an utterance observed more frequently in the training data?* Subsequently, we can parse this more frequent utterance to understand meaning of the original utterance. Towards this end, we propose a new modular paraphrase driven parsing model, which can be integrated with any existing parser, to make it more robust to out-of-vocabulary and paraphrased utterances. In our proposed hybrid approach, we augment a parser with a paraphrase generator, which can be used to map an infrequent utterance to a more frequent paraphrased utterance. Traditional neural paraphrase generators, trained on large paraphrase corpus, however do not perform well in our setting with limited parser training data. Therefore, we further develop novel RNN and multi-task sequence-to-sequence based paraphrase generators, as well as techniques to build custom paraphrase datasets for their training. In our experiments, on both benchmark and custom in house datasets, we show that our hybrid paraphrase driven parsers can improve both accuracy and robustness of existing state-of-the-art and commercial parsers.

## 2. Problem and background

In this section we formally define the intent classification and slot labeling problem, as well as discuss existing approaches. We are provided with a labeled training dataset $T = \{\mathbf{x}_i, \mathbf{y}_i, I_i\}_{i=1}^{N}$, where $\mathbf{x}_i$ are the utterances with words in a vocabulary $\mathcal{V}_T$, $\mathbf{y}_i$ represent the sequence of slot tags from a slot vocabulary $\mathcal{S}$, and $I_i \in \mathcal{I}$ represent the intent label of the utterance. A SLU unit consists of a parser $\mathcal{P}$ which can map an utterance $\mathbf{x}$ to its slot and intent labels $(\mathbf{y}, I)$. Figure 1 shows some example labeled utterances from benchmark ATIS dataset.

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Utterance 1** | i | need | a | flight | from | chicago | to | san | francisco | on | a | thursday |
| **Slots** | O | O | O | O | O | B-from.city | O | B-to.city | I-to.city | O | O | B-day.name |
| **Intent** | atis_flight | | | | | | | | | | | |
| **Utterance 2** | show | all | thursday | flights | from | chicago | to | san | francisco | | | |
| **Slots** | O | O | B-day.name | O | O | B-from.city | O | B-to.city | I-to.city | | | |
| **Intent** | atis_flight | | | | | | | | | | | |

Figure 1: *Examples of labeled utterances from our paraphrase dataset generated from ATIS training corpus.*

**Recurrent and sequence-to-sequence models:** State-of-the-art and commercial neural network based parsers often use a single sequence-to-sequence and recurrent network to jointly infer the intent and slot labels [6, 7, 8, 9]. Such encoder–decoder based deep neural networks for sequence learning have received



10.21437/Interspeech.2018-2358

considerable attention in the recent past due to its success in a variety of NLP tasks e.g. machine translation [10, 11], parsing [12, 13, 14], text generation [15], paraphrasing [16] and so on. Incorporating more encoder side information during decoding has been shown to further improve performance of these models [17]. A basic sequence-to-sequence neural network consists of an encoder $E$, and a decoder $D$, where each of them can be made up of multiple stacked recurrent units (e.g. LSTM). An input sequence $\mathbf{x} = (x_1, \ldots, x_n)$ is first encoded by repeatedly passing consecutive input symbols and previous hidden state $h_{t-1}$ through the encoder unit, $t \in [n]$. During decoding the decoder is first initialized with the final hidden state of the encoder, also called the context vector $c = h_n$. Subsequently, the decoder hidden state $h'_t$ is updated at the decoder using the previous hidden state $h'_{t-1}$ and output symbol $y_{t-1}$, $t \in [m]$. The outputs are predicted using a softmax of the projected hidden decoder states as $p(y_t = y|y_{<t}) = softmax(W_o h'_t)\mathbf{1}_y$, using a projection matrix $W_o$. A sequence-to-sequence model is trained by maximizing the likelihood function:

$$p(y_1, .., y_m|x_1, .., x_n) = \Pi_{t=1}^{m} p(y_t|y_1, .., y_{t-1}, c) \quad (1)$$

# 3. Our models

Neural network parsers suffer from poor generalization on examples seen infrequently in their training data. In voice controlled personal agents this is of major concern since individual users often like to use their own personalized vocabulary and paraphrased utterances, which may not be present in the training data. Instead of adapting the parser directly to infrequent examples, we can choose to pre-process the original input to a more frequent example in the training data. This motivates our hybrid paraphrase driven parser for SLU discussed next.

The key idea behind our model is the following. Suppose there exists a base parser $\mathcal{P}_{base}$, trained using a dataset $T$ with vocabulary $\mathcal{V}_T$. We augment this base parser with a paraphrase generator $\mathcal{P}_{para}$ trained on paraphrases from the training dataset $T$, or unlabeled user log dataset. Now, when the base parser is unable to find the intent and slots of an infrequent utterance $\mathbf{x}$ with sufficient confidence, it chooses to retrieve a more frequent paraphrase of this utterance $\mathbf{x}'$ using the paraphrase generator $\mathcal{P}_{para}$. The base parser then proceeds to infer the intent and slots from this paraphrased utterance $\mathbf{x}'$. Since the paraphrase generator finds a frequent paraphrase $\mathbf{x}'$, the base parser is expected to achieve a higher parsing confidence on this new utterance. In essence the paraphrase generator acts as a translator between the user and $\mathcal{P}_{base}$.

**Algorithm:** Our paraphrase based parsing algorithm works as shown in Figure 2. Suppose for each utterance $\mathbf{x}$, the base parser generates a confidence score $S(\mathbf{x})$ on the quality of the inferred intent and slots. Although we would like the paraphrase generator to paraphrase infrequent utterances, we do not want it to negatively effect the performance of the base parser $\mathcal{P}_{base}$. Therefore, only the utterances with low parsing confidence $S(\mathbf{x}) < \tau$ are sent for paraphrasing. Computing confidence score of neural network output has been studied in various applications e.g. question answering [18], semantic parsing [19]. We compute a separate confidence score of the output intent label as the probability of the label from output softmax layer, $S_{intent}(\mathbf{x}) = P(I = \ell)$, $\ell \in \mathcal{I}$. Similarly, using the output probability of each of the slot tags, we compute an overall slot tagging score $S_{slot}(\mathbf{x}) = \exp\left(\frac{1}{m}\sum_j \log P(y_j = s_j)\right)$, $s_j \in \mathcal{S}$, using the normalized log likelihood of the tag sequence. The final score is computed as the minimum of these two scores $S(\mathbf{x}) = \min\{S_{intent}(\mathbf{x}), S_{slot}(\mathbf{x})\}$.

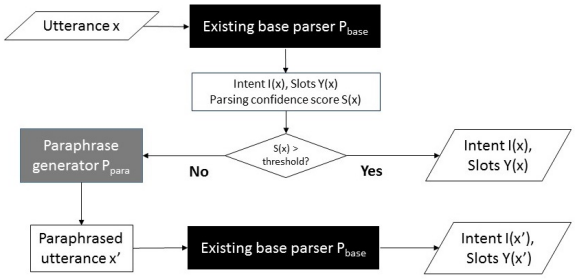


Figure 2: *Flowchart illustrating our paraphrase generator augmented robust parser for intent classification and slot tagging.*

We next describe the design of our paraphrase generator $\mathcal{P}_{para}$. Many paraphrase generation techniques have been studied in literature [20, 21, 22, 23, 24]. However, most of these require additional labeled paraphrase training data which may not be always available. More importantly, these techniques do not guarantee that the generated paraphrase $\mathbf{x}'$ of $\mathbf{x}$ is a more frequent example which is well understood by the base parser. Therefore, we design two new paraphrase generation algorithms which are most suited for our parsing application. The first algorithm leverages an in–domain RNN language model to generate paraphrases using multiple word replacement. The second algorithm employs a neural paraphrase generation technique using a multi-task sequence-to-sequence model.

### 3.1. RNN language model based paraphrase generator

RNN based language models have been widely used in automated speech recognition [25, 26]. We use a similar language model as the main component in our first paraphrase generator. Note that, we require our paraphrases to be similar to training utterances of the base parser $P_{base}$. Therefore we leverage the same training data, without any labels, to train our RNN language model. When the training dataset is small, this may not be sufficient to obtain a well trained language model. In such cases, we can leverage the large un-annotated user log data of a deployed personal agent, to train our language model. Such user log data is easily available in practical applications. We also train simultaneously two language models; $\mathcal{L}_f$ in the forward direction which predicts the probability of the $i^{th}$ word $w_i$ as $P(w_i|w_{i-1}, ..., w_{i-k})$; and $\mathcal{L}_b$ in the backward direction using a reversed corpus, predicting $P(w_i|w_{i+1}, ..., w_{i+k})$, for a chosen $k$. Next we describe how these two language models are used for paraphrase generation.

This model is motivated by the following observation. We use the term *context words* as words having the slot label "O" (which are non-informational), and *slot words* as the remaining informational words. For example in Figure 1, words {"chicago", "san francisco", "thursday"} are slot words, and the remaining are context words. We observe that, often when the base parser $P_{base}$ fail to identify the correct slot labels, it can still identify the position of the slot words (but not their exact labels) with sufficient confidence. Since, context words play a major role in enabling identification of slot words, we would like to replace context words having low parser confidence with more frequent words, thereby generating a new paraphrase. This enables the slot words to be correctly labeled using this paraphrase. After $P_{base}$ identifies slot words in utterance $\mathbf{x}$,

we assume the remaining words are context words, and find the average slot confidence $\overline{S}_C(\mathbf{x})$ over these context words $C$. We generate a paraphrase template $T(\mathbf{x}) = (u_i, \ldots, u_n)$ as follows; $u_i = x_i$, if slot probability $P(y_i = s_i) > \overline{S}_C(\mathbf{x})$, or if $x_i$ is a slot word, else we replace $u_i = \langle ? \rangle$, a special blank token. We then run a modified beam search algorithm using the forward language model $\mathcal{L}_f$ over the template $T(\mathbf{x})$, such that the beams are constrained to generate $u_i = x_i$ for non blank tokens, but are allowed to generate new words to replace the blank tokens $\langle ? \rangle$ in the template. However, these hard constraints tend to reduce the normal beam search quality of the RNN. To mitigate this, we also perform a similar reverse beam search using a reversed language model $\mathcal{L}_b$. Finally, all generated beams are scored by both language models, and the one having the highest average score is output as paraphrase $\mathbf{x}'$. As an example, a possible template $T(\mathbf{x})$ for utterance 1 in Figure 1 is *"$\langle ? \rangle$ $\langle ? \rangle$ a flight from chicago to san francisco on $\langle ? \rangle$ thursday"*; after beam search this may produce a paraphrase *"show me a flight from chicago to san francisco on next thursday"*.

### 3.2. Multi-task neural paraphrase generator

The paraphrases generated by RNN language model based generator can improve the slot identification performance of a parser (shown in Section 4). However, the parser may still fail to correctly determine intent when the input utterance $\mathbf{x}$ is a structural paraphrase of some training utterance. Word replacement based paraphrase generators can never produce such structural variation. To tackle this issue our second paraphrase generator uses a neural multi-task sequence-to-sequence model.

Sequence-to-sequence based neural paraphrase generator has been proposed recently by Prakash et al. [27]. However, we observe that the basic attention based sequence-to-sequence model do not perform well in our setting due to difficulty in paraphrasing utterances with rare slot words. Our paraphrase generator incorporates a single sequence encoder $E$, but two separate sequence decoders $D_1$ and $D_2$ as shown in Figure 3. During forward pass, both the decoders are initialized with the same encoder context vector $c$, and then proceeds to decode the sequences independently. However, during training, we constraint the second decoder $D_2$ to generate the exact same input utterance $\mathbf{x}$, while the first decoder generates the paraphrase $\mathbf{x}'$. Such additional autoencoder constraint has also been used in models for domain adaptation in order to obtain a better hidden representation vector of an utterance [28]. In our application, this better shared hidden representation encourages the correct reproduction of slot words even at the first decoder output. The model is trained using the joint multi-task objective function of the sum of the individual sequence loss functions at decoders $D_1, D_2$. In addition, as a metric to determine the quality of the model during validation, we use a sum of BLEU score between input $\mathbf{x}$ and decoder 1 output $\mathbf{x}'$, and the reconstruction accuracy of the input at decoder 2. Note that, although decoder 2 is trained as an autoencoder, during inference it may not always produce the same sequence as the input. We observe that decoder 2 output is also often a paraphrase of $\mathbf{x}$, having less structural variation. Therefore, we can use both the decoder outputs as paraphrases of $\mathbf{x}$, to be parsed by base parser $\mathcal{P}_{base}$.

**Paraphrase dataset generation:** In order to train our multi-task neural paraphrase generator, we generate a paraphrase dataset $T_{para}$ from the base parser training set $T$ as follows. First, we convert each utterance $\mathbf{x} \in T$, to a *tagged utterance* where the slot words have been replaced by slot labels. For example, the tagged utterance corresponding to utterance 1 in Figure 1 is *"i need a flight from @from.city to @to.city on a @day.name"*. Now we observe that, tagged utterances having the same intent, and identical set of slot labels are paraphrases, since they are intended to convey the same meaning. This enables us to construct a tagged paraphrase dataset $T_{tagged}$ consisting of tuples of distinct tagged utterances $(\mathbf{z}, \mathbf{z}')$ which have the same intent and slot set. We then replace back the slot words from the parent utterance of $\mathbf{z}$ in both $\mathbf{z}$ and $\mathbf{z}'$, and vice versa. This generates the paraphrased dataset $T_{para}$ having tuples of paraphrases $(\mathbf{x}, \mathbf{x}')$. Figure 1 shows a paraphrase sample from this dataset. In addition, we also consider all training examples $\mathbf{x} \in T$ as identity paraphrases $(\mathbf{x}, \mathbf{x})$ and add them to $T_{para}$. This prevents the paraphrase generator to perform poorly, when it encounters an utterance which did not have any other paraphrase in $T$.

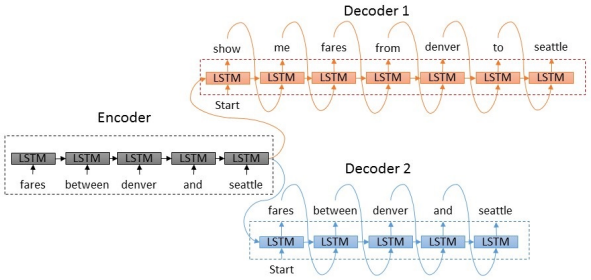


Figure 3: *Figure showing the architecture and training strategy of our multi-task sequence-to-sequence paraphrase generator. Additional attention structure from encoder to both decoders has been omitted in the diagram for clarity.*

## 4. Experiments

In this section we describe our experimental results. We want to evaluate the intent and slot tagging accuracy gains using our paraphrase models compared to just a standalone parser.

Table 1: *Examples of complex utterances in our simulated ATIS log corpus.*

| Intent | Utterance |
|---|---|
| atis flight | show me trip that leaves tuesday on american airline going from baltimore leaving early night arriving in pittsburgh |
| atis airfare | give me the fares with continental leaving from long beach for flights one way with first class arriving in tacoma |

**Datasets:** For evaluation we use the benchmark ATIS dataset [29], which is popularly used for evaluating parsers for spoken language understanding. The ATIS dataset contain 5,871 utterances related to airline reservation with 4,978 training and 893 test utterances. Overall it contain 17 intent labels and 79 slot labels. Example utterances from this dataset is shown in Figure 1. In order to show the accuracy gains for various sizes of training set, we further sample training sets of such sizes, but test parser performance on the full ATIS test set.

After deployment of an intelligent personal agent, often the distribution of the observed utterances turn out to be significantly different from those used in training. This is because each individual users have their own preferred choice of paraphrase and vocabulary, and this can change over time [28]. In order to test the robustness of our models in such adversarial

scenario, we generate a simulated *ATIS log* dataset as follows. Starting with the original ATIS dataset, we use data recombination techniques similar to [13], to generate a variety of long and complex utterances. Then, human linguistic experts prune any incorrect utterance. We train all models on the original ATIS training set, then we test their performance on a set of 1,000 ATIS log utterances for testing. Example utterances from our dataset are shown in Table 1.

Table 2: *Comparison of* 10 *fold average test intent accuracy percentage of all models on ATIS corpus with increasing size of training set, using the attention BiRNN [7] as the base parser.*

| Parsers | Training dataset size | | | | |
|---|---|---|---|---|---|
| | **500** | **1500** | **2500** | **3500** | **4500** |
| BiRNN (Liu and Lane) | 87.95 | 93.06 | 94.26 | 96.08 | 96.47 |
| Seq-to-seq paraphrase + BiRNN | 88.16 | 93.10 | 94.54 | **96.26** | **96.58** |
| RNN paraphrase + BiRNN | **88.63** | **93.39** | **94.55** | 96.20 | 96.52 |

Table 3: *Comparison of* 10 *fold average slot tagging F1 score percentage of all models on ATIS corpus with increasing size of training set, using the attention BiRNN [7] as the base parser.*

| Parsers | Training dataset size | | | | |
|---|---|---|---|---|---|
| | **500** | **1500** | **2500** | **3500** | **4500** |
| BiRNN (Liu and Lane) | 79.96 | 88.57 | 90.83 | **91.33** | 92.02 |
| Seq-to-seq paraphrase + BiRNN | 79.83 | 88.44 | 90.76 | 91.28 | 91.98 |
| RNN paraphrase + BiRNN | **80.01** | **88.62** | **90.84** | 91.29 | **92.02** |

**Baselines and parameters:** We use two baseline parsers for evaluation. First, we use the state-of-the-art *Attention BiRNN* based neural network parser by Liu and Lane [7]. As a second baseline parser we use the open source *RASA* parser [30], in order to demonstrate the applicability in commercial agents and dialog systems. We augment both these parsers with our paraphrase generation models and compare their performance with the former. For attention BiRNN, we use the Tensorflow implementation made available by Liu et al. with its default parameters. We also use the RASA parser with its default settings. Our neural models were implemented in Tensorflow. The confidence threshold $\tau$ in our paraphrase models were set to $0.8$. For RASA we were unable to use RNN based paraphrase model, since it does not return the slot tagging probabilities required for paraphrase template construction. In paraphrase models, we generate two best paraphrases using the paraphrase generators, and perform a simple majority voting to predict the final intent and slot labels.

Table 4: *Comparison of* 10 *fold average test intent accuracy and slot tagging F1 score percentages of the sequence-to-sequence model on ATIS corpus with increasing size of training set, using RASA [30] as the base parser.*

| Parsers | Metric | Training dataset size | | | |
|---|---|---|---|---|---|
| | | **500** | **1500** | **2500** | **3500** |
| RASA | Accuracy | 83.70 | 86.81 | 88.33 | 88.44 |
| Seq-to-seq paraphrase + RASA | Accuracy | **84.64** | **89.32** | **90.06** | **91.62** |
| RASA | F1 | 75.39 | 81.34 | 83.41 | 84.71 |
| Seq-to-seq paraphrase + RASA | F1 | 75.25 | 81.32 | 83.22 | 84.55 |

### 4.1. Results

First we compare the performance of different models on the benchmark ATIS dataset. In Table 2 we compare the 10 fold average intent detection accuracy of our models when combined with the attention BiRNN baseline model. We observe that both our models improve the accuracy of the baseline parser. Further, the accuracy gain is higher when the training set size is small. In Table 3 we compare their corresponding average slot tagging F1 scores. The RNN paraphrase model is observed to improve F1 score of the baseline model, while the sequence-to-sequence model do not. This is expected, since the RNN paraphrase model only replaces the low confidence context words with more frequent words, which enables the base parser to better identify the slot labels. In contrast, the sequence-to-sequence paraphrase model may alter the sentence structure and slot words in its paraphrases, hence doesn't always improve slot tagging F1 score. We further observe that it often improves the recall, but not its precision.

Table 5: *Comparison of* 10 *fold average test intent accuracy percentage of all our models on simulated ATIS log corpus with increasing size of training set, using both attention BiRNN [7] and RASA [30] as the base parser. The models are trained on original ATIS dataset but tested on ATIS log corpus.*

| Parsers | Training dataset size | | | |
|---|---|---|---|---|
| | **500** | **1500** | **2500** | **3500** |
| BiRNN (Liu and Lane) | 80.49 | 82.22 | 82.71 | 82.65 |
| Seq-to-seq paraphrase + BiRNN | 82.41 | 82.51 | 83.54 | 82.87 |
| RNN paraphrase + BiRNN | **83.25** | **83.31** | **84.24** | **83.22** |
| RASA | 77.60 | 83.16 | 83.56 | 84.26 |
| Seq-to-seq paraphrase + RASA | **79.66** | **85.14** | **86.08** | **84.52** |

Next, we compare the performance of the sequence-to-sequence paraphrase model, when used with RASA as the base parser. RASA uses a kernel SVM classifier along with feature selection. Hence, in general it has a worse performance than attention BiRNN parser. However, it has the advantage of fast training time. Table 4 compares both the intent and slot tagging performance. Once again, we observe that the paraphrase model is able to achieve high gains in intent accuracy over the baseline RASA parser. The slot tagging performance do not improve with this model as previously observed with attention BiRNN parser. As mentioned before, due to the lack of slot tagging confidence scores in RASA, we are unable to use our RNN based paraphrase model with RASA.

Finally, to validate the robustness of our models in an adversarial post deployment scenario, we test the performance of our models on the simulated ATIS log corpus. Table 5 reports the intent classification accuracy of all our models. Due to a distribution mismatch with the ATIS training data, all models perform worse in this dataset, as expected. However, we still observe that, irrespective of the base parser used, our paraphrase models achieve an improved intent detection accuracy.

## 5. Conclusion

Commercial parsers trained using data driven approaches, often have poor performance after deployment, when it encounters a variety of complex paraphrases and out-of-vocabulary words that were unseen or infrequent in its training data. In this paper, we propose a novel paraphrase driven parsing approach, where during parsing such complex paraphrases are first converted to a more familiar utterance using a paraphrase generator. We propose two new paraphrase generation techniques suitable to use in our application. Our experimental results validate that, irrespective of the base parser being used, or the test data distribution being observed, our combined models are able to greatly improve the performance of the standalone base parser.

# 6. References


[1] P. Haffner, G. Tur, and J. H. Wright, "Optimizing svms for complex call classification," in *Proc. of ICASSP'03*, vol. 1. IEEE, 2003.

[2] Y.-Y. Wang, L. Deng, and A. Acero, "Spoken language understanding," *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 16–31, 2005.

[3] C. Raymond and G. Riccardi, "Generative and discriminative algorithms for spoken language understanding," in *INTERSPEECH 2007*, 2007, pp. 1605–1608.

[4] 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*, 2013, pp. 78–83.

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

[6] D. Hakkani-Tür, G. Tür, A. Celikyilmaz, Y.-N. Chen, J. Gao, L. Deng, and Y.-Y. Wang, "Multi-domain joint semantic frame parsing using bi-directional rnn-lstm." in *INTERSPEECH*, 2016, pp. 715–719.

[7] B. Liu and I. Lane, "Attention-based recurrent neural network models for joint intent detection and slot filling," in *Interspeech 2016*, 2016, pp. 685–689.

[8] Y. Kim, S. Lee, and K. Stratos, "ONENET: joint domain, intent, slot prediction for spoken language understanding," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop*, 2017, pp. 547–553.

[9] Y. Wang, Y. Shen, and H. Jin, "A bi-model based RNN semantic frame parsing model for intent detection and slot filling," in *Proc. of the 2018 NAACL-HLT, New Orleans, Louisiana, USA*, 2018, pp. 309–314.

[10] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. of the 2014 EMNLP*, 2014, pp. 1724–1734.

[11] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. of NIPS*, 2014, pp. 3104–3112.

[12] O. Vinyals, L. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. E. Hinton, "Grammar as a foreign language," in *Proc. of NIPS*, 2015, pp. 2773–2781.

[13] R. Jia and P. Liang, "Data recombination for neural semantic parsing," in *Proc. of the 54th ACL*, 2016.

[14] L. Dong and M. Lapata, "Language to logical form with neural attention," in *Proc. of the 54th ACL 2016*, 2016.

[15] A. M. Rush, S. Chopra, and J. Weston, "A neural attention model for abstractive sentence summarization," in *Proc. of the 2015 EMNLP*, 2015, pp. 379–389.

[16] A. Prakash, S. A. Hasan, K. Lee, V. V. Datla, A. Qadir, J. Liu, and O. Farri, "Neural paraphrase generation with stacked residual LSTM networks," in *Proc. of COLING 2016*, 2016, pp. 2923–2934.

[17] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of the ICLR, San Diego, California*, 2015.

[18] D. Gondek, A. Lally, A. Kalyanpur, J. W. Murdock, P. A. Duboué, L. Zhang, Y. Pan, Z. Qiu, and C. Welty, "A framework for merging and ranking of answers in deepqa," *IBM Journal of Research and Development*, vol. 56, no. 3, p. 14, 2012.

[19] L. Dong, C. Quirk, and M. Lapata, "Confidence modeling for neural semantic parsing," *arXiv preprint arXiv:1805.04604*, 2018.

[20] D. Kauchak and R. Barzilay, "Paraphrasing for automatic evaluation," in *Proc. of HLT–NAACL*, 2006, pp. 455–462.

[21] S. Zhao, H. Wang, T. Liu, and S. Li, "Pivot approach for extracting paraphrase patterns from bilingual corpora." in *ACL*, vol. 8, 2008, pp. 780–788.

[22] C. Quirk, C. Brockett, and W. Dolan, "Monolingual machine translation for paraphrase generation," in *Proc. of EMNLP 2004*, 2004.

[23] S. Zhao, C. Niu, M. Zhou, T. Liu, and S. Li, "Combining multiple resources to improve smt-based paraphrasing model." in *ACL*, 2008, pp. 1021–1029.

[24] S. Zhao, X. Lan, T. Liu, and S. Li, "Application-driven statistical paraphrase generation," in *Proc. of the Joint Conference of the 47th ACL and the 4th IJCNLP of the AFNLP*, 2009, pp. 834–842.

[25] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *INTERSPEECH 2010*, 2010, pp. 1045–1048.

[26] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *INTERSPEECH 2012*, 2012, pp. 194–197.

[27] A. Prakash, S. A. Hasan, K. Lee, V. V. Datla, A. Qadir, J. Liu, and O. Farri, "Neural paraphrase generation with stacked residual LSTM networks," in *COLING 2016*, 2016, pp. 2923–2934.

[28] Y. Kim, K. Stratos, and D. Kim, "Adversarial adaptation of synthetic or stale data," in *Proc. of ACL 2017*, 2017, pp. 1297–1307.

[29] C. T. Hemphill, J. J. Godfrey, G. R. Doddington *et al.*, "The atis spoken language systems pilot corpus," in *Proceedings of the DARPA speech and natural language workshop*, 1990, pp. 96–101.

[30] RASA, 2018, https://rasa.ai/.