SEGMENTING MULTI-INTENT QUERIES FOR SPOKEN LANGAUGE UNDERSTANDING

Rohan Shet, Elena Davcheva, Christian Uhle

Fraunhofer Institute of Integrated Circuits IIS,Erlangen,Germany
rohan.mangalore.shet@iis.fraunhofer.de

Abstract: With the rising popularity of voice assistants, automatic speech recognition (ASR) systems play a crucial role in translating speech to text in order to enable natural language understanding (NLU) models to process human commands. However, NLU models lack the proper resources to handle certain natural human speech characteristics, such as dividing or segmenting several intents expressed in one spoken sentence. This study presents an innovative approach to sentence boundary segmentation from ASR output using a neural network model. We improve on previous attempts by removing the need for complex model output postprocessing, as well as reporting higher accuracy than previous studies on the subject.

Introduction

In spoken language understanding (SLU), various natural language processing (NLP) tasks are carried out in order to enable voice assistants to understand user goals and fulfill requests. Outside the context of SLU systems, working with textual data provides the benefit of using punctuation as a signal for segmenting sub-queries. However, when dealing with output from automatic speech recognition (ASR) systems, punctuation can be absent, therefore for applications such as voice assistants, researchers are yet to come up with a compelling solution for this problem. Incorrect query segmentation will have negative effects on downstream NLP tasks such as intent classification, slot filling, or named entity recognition. In this study we focus on intent classification as a downstream task. As shown in Figure 1, when multi-intent queries are improperly segmented, for every word incorrectly predicted outside of the ground truth segment boundary, the error of the downstream NLU intent classifier doubles. This study addresses query segmentation in the context of voice assistants by putting forth a novel approach to train

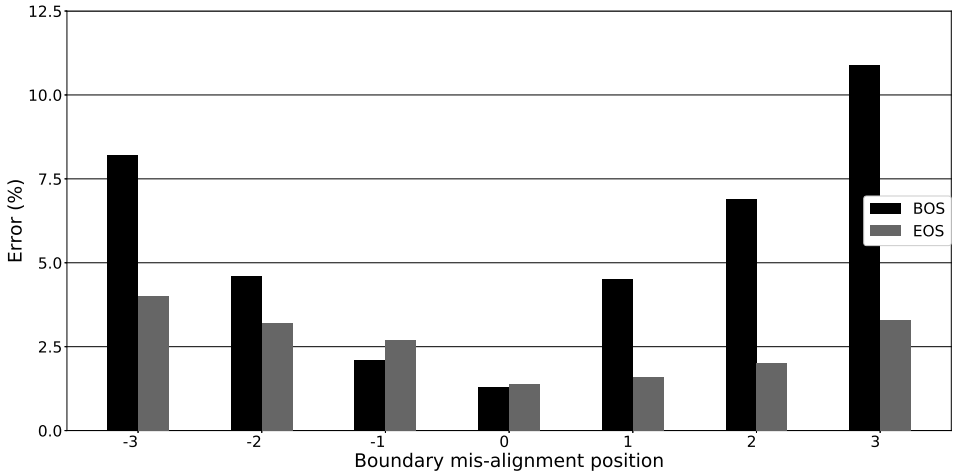

Figure 1 – Evaluating the performance degradation of an intent classifier due to improper segmentation

deep learning models in how to segment multi-intent input sequences into queries of single intents. Table 1 shows an example of a single- and multi-intent sentence. The resulting single-intent queries are subsequently processed within the NLP pipeline of an SLU system in order to identify an intent. The proposed method improves on previous attempts by removing the need for grammar-based rules, thus greatly expanding the cases in which segment boundaries can be identified, as well as removing additional steps such as hypothesis generation and evaluation, or the need to add additional learning features.

The model is evaluated in two stages. First, a segmentation model is evaluated in two separate experiments, and in another experiment we test the downstream NLP task of intent classification with the output of the segmentation model serving as input. The results of the experiments show that our approach consistently outperforms previously proposed solutions.

Table 1 – An example for two types of input sentences that occur when interacting with a voice assistant

Type of sentence	Example
Single-intent	Find the Big Bang Theory tv show
Multi-intent	Find the Big Bang Theory tv show and play it

Related work

Previous work on multi-intent query segmentation for SLU includes using words with specific grammatical roles as basic signals for where an intent ends and/or another one starts. Oftentimes, grammar constructs such as conjunctions (words that link other words, phrases, or clauses together, e.g. "and", "but", "yet") are used as keywords for separation. Such models tend to be rule-based, in other words the model employs no mechanism to learn possible other intent change signals from training data. Furthermore, such models depend on complex multi-step processing, including hypothesis generation and evaluation [1]. The top accuracy that such a model can produce was about 80%. Other studies have used n-gram patterns as additional features, as well as either multiple separate or concatenated intent labels [2][3]. Multi-intent or concatenated intent labels have shown poor performance, with an accuracy of 82% for intent classification [2].

What previous models have in common is complex data preprocessing and model output post-processing, where several additional learning features need to be constructed from the training data, and after a model outputs predictions, different interpretations of these predictions must take place. Furthermore, an accuracy of 80% suggests much room for improvement.

Methodology

In order to perform multi-intent classification we propose a two-stage approach as shown in Figure 2: 1) query segmentation, followed by 2) intent classification. For this purpose, we have built and tested a model which performs segmentation on a multi-intent query and decomposes a single utterance into multiple utterances. The model takes a sentence as input and outputs the sequence of tags corresponding to segmentation boundaries. We limit our model to detect maximum 2 segments.

We assume that the input source sequence is a sequence of words output by an ASR system without any punctuations. The sequence of words is then converted to a word vector using ELMo (Embeddings from Language Models) - state-of-the-art word embeddings for the English language [6]. The word vector representations from ELMo are character-based, which allows the network to use morphological clues to form robust representations for unseen words. The

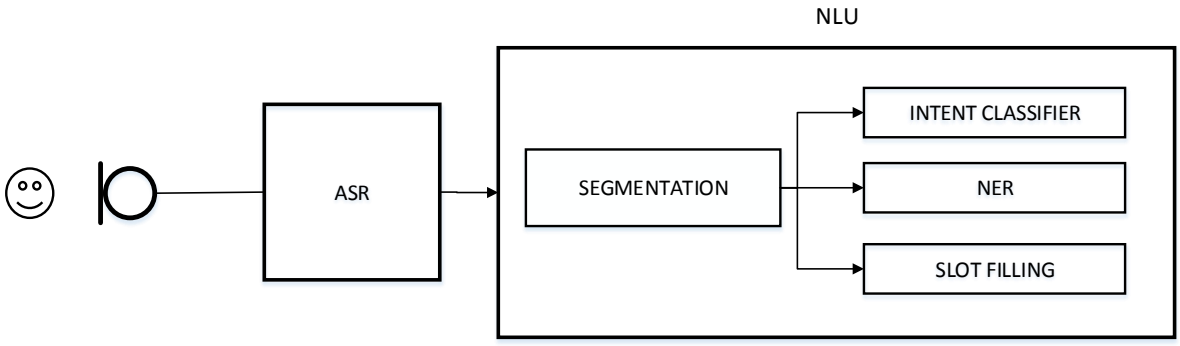

Figure 2 – The partial NLP pipeline used in this study - from ASR output, an utterance gets processed by the segmentation model, from where single-intent queries can be classified by downstream NLP tasks such as intent classification or named entity recognition (NER)

embedding for each word depends not only on the word itself but also on any context in which it has been observed.

The segmentation model uses a bi-directional neural network (BiRNN) with a gated recurrent unit (GRU)[4] as cell architecture. BiRNNs have been successfully applied in NLP tasks such as slot filling or intent recognition. We use gated recurrent units (GRUs) as they have a better ability to model long-term dependencies than a basic cell architecture. Both LSTM (long short term memory)[5] and GRU cells perform similarly well in our experiments. Conditional random fields (CRF) are added as a last layer for classification. The queries in the training data are prepared by explicitly labeling start (BOS) and end (EOS) tokens of the separate segments, and all other words are labeled with a token I. This allows a neural network to organically learn to segment sub-queries, without relying on predefined semantic rules or additionally learning features.

In the segmentation task, our goal is to map a sequence of words $x = (x_1, ..., x_T)$ to its corresponding boundary label $y = (y_1, ..., y_T)$, $y_k \in \{bos, i, eos\}$, where *bos*, *i*, *eos* stand for begin-of-segment, inside-segment and end-of-segment, respectively.

A BiRNN consists of a forward and a backward RNN. The forward RNN $\overrightarrow{f}$ reads the input sequence as it is ordered $(x_1, ..., x_T)$ and calculates a sequence of forward hidden states $(\overrightarrow{h}_1, ..., \overrightarrow{h}_T)$. Then, the backward RNN $\overleftarrow{f}$ reads the sequence in the reverse order $(\overleftarrow{h}_T, ..., \overleftarrow{h}_1)$ resulting in a sequence of backward hidden states. We can obtain an annotation for each word x_i by concatenating the forward hidden state $\overrightarrow{h}_i$ and the backward one $\overleftarrow{h}_i$. i.e., $h_i = [\overleftarrow{h}_j, \overrightarrow{h}_j]$. In this way, the annotation contains the summaries of words both preceding and following x_i. Due to the tendency of RNNs to better represent recent inputs, the annotation h_i will be focused on the words around x_i .

The concatenated outputs from the BiRNN pass through a feed-forward neural layer, which projects the high-dimensional output from the BiRNN to low-dimension output. Instead of modelling the output sequence tags independently, we model them jointly using a conditional random field layer, and apply softmax over all possible output sequence tags. i.e., we try to maximize $O(\Theta)$ - the log-likelihood probability of the correct tag sequence $y = (y_1, ..., y_T)$ for a given input sequence $x = (x_1, ..., x_T)$.

$$O(\Theta) = \sum_{i=1}^{N} \log p_\Theta\left(y_{(1...T)} | x_{(1...T)}\right) \tag{1}$$

Experiments and Results

In the following experiments, we train and test a segmentation model and an intent classifier. In Experiment 1, the segmentation model has been evaluated based on the number of segments predicted, and in experiment 2 - based on the segment boundary misalignment between ground truth and prediction. Finally, in experiment 3 we evaluate the performance degradation of the classifier on downstream tasks, i.e. we evaluate how much the performance on the task "intent classification" worsens due to errors in the task "query segmentation".

Dataset

To test our models, we have used the publicly available SNIPS and ATIS datasets. SNIPS is used to benchmark various NLU tasks [7]. It consist of 16.000 crowdsourced utterances on 7 intents for domains such as weather, music, books, or restaurant bookings. SNIPS is a balanced dataset with approximately 2400 utterances per intent. The Airline Travel Information System Dataset (ATIS) contains transcribed audio recordings of people making flight reservations [8]. It consists of 4978 utterances for training and 893 utterances for testing. There are in total 127 distinct slot labels and 18 different intent types.

These datasets consist of a single intent per utterance. Since our experiments deal with multiple segments per utterance where each segments indicates a different intent, we create a multi-intent dataset by randomly selecting 2 sentences from these datasets and combining them with or without a conjunction to form a multi-intent sentence. We perform 5-fold cross-validation in all our experiments. 50% of the data for training and testing consist of single-segment sentences, and the other 50% consist of two-segment sentences.

Evaluation Metrics

We evaluate our method by concatenating sentences from the datasets to arrive at synthesized multi-request queries [1]. The segmentation evaluation is done by means of comparing the number of found and missed segment boundaries, as well as quantifying the shift (or misalignment) between true and predicted segment boundaries. The subsequent extrinsic evaluation of the identified segments is carried out in the context of intent classification for concatenated queries.

The nature of the boundary segmentation task does not render itself to the use of conventional performance metrics such as recall, precision, or accuracy, which is why specifically for the segmentation task we apply an alternative way to calculate accuracy. To evaluate the accuracy of the segmentation model, we compare the segments detected by the model with the true segments present in the sentence via the following equation.

$$E_N = N_r - N_p \tag{2}$$

where E_N is the difference in segmentation between the number of true segments N_r and detected segments N_p by the segmentation model. The metric proposed above is specific to the case where maximum two segments can be predicted. It gives a broad overview on the preformance measure of the segmentation task, but does not allow for detailed evaluation. Hence, in order to evaluate segmentation in a fine-grained manner, we propose evaluation on the basis of segment boundary positions. Table 2 provides a rough overview on the evaluation metrics proposed. In the table, Case 1 predicted the first segment longer than the ground truth, for which it received a misalignment score of +1; case 2 predicted the second segment too long, hence its misalignment score is -1.

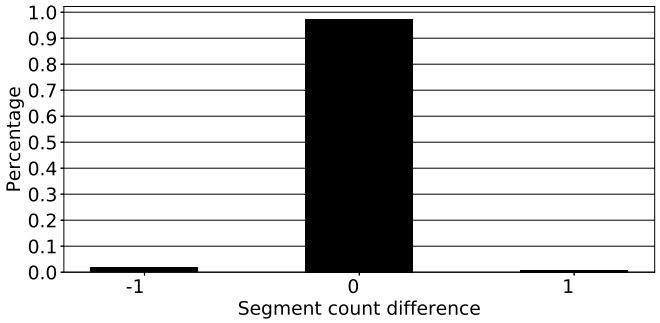

Figure 3 – Evaluating segmentation model based on number of segments detected on SNIPS data when trained with SNIPS data

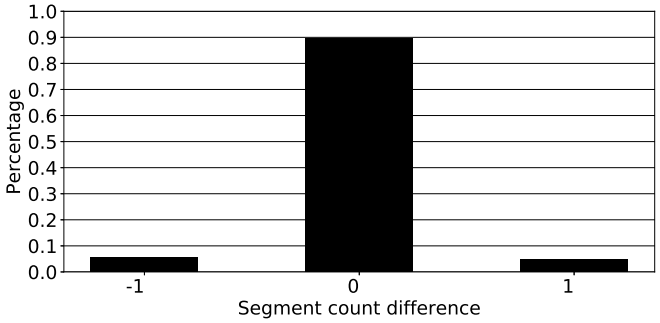

Figure 4 – Evaluating segmentation model based on number of segments detected on ATIS data when trained with SNIPS data

The output of the segmentation module will be used as input for the downstream task of intent classification. Therefore, we calculate an overall performance of the combined segmentation and intent classification tasks.

Table 2 – An example of boundary misalignment for an example sentence.

Example sentence:	**what**	**is**	**the**	**weather**	**and**	**book**	**a**	**restaurant**	**in**	**geddes**
Ground Truth	BOS	I	I	EOS	I	BOS	I	I	I	EOS
Predicted case 1	BOS	I	I	I	EOS	BOS	I	I	I	EOS
misalignment score	0	-	-	+1	-	0	-	-	-	0
Predicted case 2	BOS	I	I	EOS	BOS	I	I	I	I	EOS
misalignment score	0	-	-	0	-	-1	-	-	-	0

Experiment 1

Figure 3 and Figure 4 present the performance evaluation of the segmentation model based on the number of segments detected on the SNIPS and ATIS datasets, respectively. When the segment difference between the ground truth and the prediction is 0, the prediction contains the correct number of segments, which in this case is 97% of the time. When testing on SNIPS, in around 2% of cases the prediction contained one more segment than ground truth, and in around 2% it contained one less segment. There was never a case where the predicted segments had two or more segments than ground truth. Testing on ATIS data (having trained on SNIPS) showed slightly worse performance, with 89% of the cases predicting the number of segments correctly, and about 11% predicting one more or one less segment.

Table 3 – Confusion matrix for the segmentation task

No. of segments	1	2
predicted as 1 segment	15545	86
predicted as 2 segment	282	15222

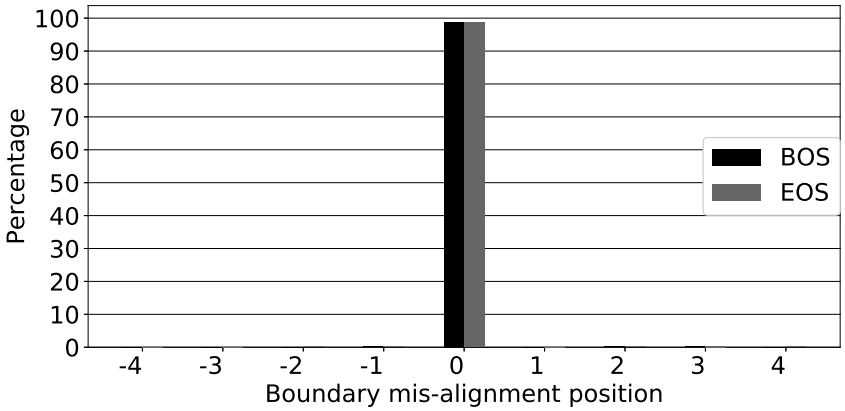

Figure 5 – Evaluating segmentation model based misalignment between the ground truth and predicted boundary positions on SNIPS data when trained on SNIPS data

Experiment 2

When evaluating the misalignment between true and detected segment boundaries, the results are similar as experiment 1. Testing on SNIPS data shows that there is no boundary misalignment for the start (BOS) and end (EOS) of a segment in around 95% of the cases. It is interesting to note that when boundaries were misaligned, they were usually misaligned by many places (5 or 4) instead of by 1 or 2. Around 3% of cases had no predicted (NP) start token, and around 3% had the end token at position n+5. Testing on ATIS data shows (similarly as in experiment 1) lower performance than on SNIPS data, however with an accuracy of 81%, which already matches the achievement of previous segmentation models proposed [1].

Experiment 3

Experiment 3 presents the performance of the intent classifier after it receives the segmented queries from the segmentation model. Table 4 presents the intent classification accuracy for two-intent queries, where accuracy is given separately for segments 1 and 2. After running the segmentation model, 15,222 utterances were correctly identified to have two segments each,

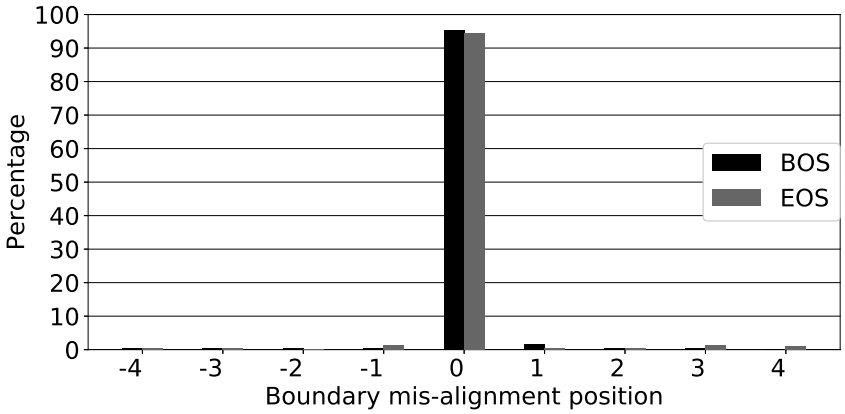

Figure 6 – Evaluating segmentation model based misalignment between the ground truth and predicted boundary positions on ATIS data when trained on SNIPS data

and only 282 utterances were not identified as being multi-segment as shown in Table 3. The intent classification performs just as well on the ground truth as on the predicted segments, even with certain boundary misalignment.

Figures 5 and 6 show the boundary misalignment when testing on SNIPS and ATIS data, respectively. Misalignment often occurs with shorter sentences e.g. "play my favorites playlist and repeat", where segment 2 is only one word: "repeat".

Table 4 – Evaluating the performance degradation of the intent classifier on SNIPS data when tested with ground truth segments and predicted segments boundary

	Segment 1 (%)	**Segment 2 (%)**
Ground Truth	98.41	98.73
Prediction	98.31	98.6

Conclusion

Based on the conducted experiments in this study, the proposed query segmentation method clearly outperforms previous models both in accuracy, as well as in streamlining the predictive process architecture. We remove the need to engineer additional learning features for the task of query segmentation, and by not requiring hypothesis evaluation, we show that it is possible to use the segmentation and intent classification models in an end-to-end NLP pipeline without further model output postprocessing. This approach was evaluated with three different experiments - number of segments predicted by the segmentation model, boundary misalignment, and performance of the downstream intent classifier.

References

[1] KIM, B., S. RYU, and G. G. LEE: *Two-stage multi-intent detection for spoken language understanding*. In *Multimedia Tools and Applications*, vol. 76: 9, pp. 11377 – 11390. 2017.

[2] XU, P. and R. SARIKAYA: *Exploiting shared information for multi-intent natural language sentence classification*. In *INTERSPEECH*, pp. 3785 – 3789. 2013.

[3] WANG, Y. and A. ACERO: *Combination of cfg and n-gram modeling in semantic grammar learning*. In *Proceedings of Eurospeech*, pp. 1229 – 1232. 2003.

[4] CHUNG, J., Ç. GÜLÇEHRE, K. CHO, and Y. BENGIO: *Empirical evaluation of gated recurrent neural networks on sequence modeling*. *CoRR*, abs/1412.3555, 2014. URL `http://arxiv.org/abs/1412.3555`. `1412.3555`.

[5] HOCHREITER, S. and J. SCHMIDHUBER: *Long short-term memory*. *Neural computation*, 9(8), pp. 1735–1780, 1997.

[6] PETERS, M. E., M. NEUMANN, M. IYYER, M. GARDNER, C. CLARK, K. LEE, and L. ZETTLEMOYER: *Deep contextualized word representations*. In *Proc. of NAACL*. 2018.

[7] *Natural language understanding benchmark*. `https://github.com/snipsco/nlu-benchmark/tree/master/2017-06-custom-intent-engines/`, 2017. Accessed: 2019-01-24.

[8] HEMPHILL, C. T., J. J. GODFREY, and G. R. DODDINGTON: *The atis spoken language systems pilot corpus*. In *Speech and Natural Language*. 1990.