# Unified Multi Intent Order and Slot Prediction using Selective Learning Propagation

**Bharatram Natarajan, Priyank Chhipa, Kritika Yadav** and **Divya Verma Gogoi**
research.samsung.com
{bharatram.n, p.chhipa, k.yadav, divya.g}@samsung.com

## Abstract

Natural Language Understanding (NLU) involves two important task namely Intent Determination (ID) and Slot Filling (SF). With recent advancements in Intent Determination and Slot Filling tasks, explorations on handling of multiple intent information in a single utterance is increasing to make the NLU more conversation based rather than command execution based. Many has proven this task with huge multi-intent training data. In addition, lots of research have addressed multi intent problem only. The problem of multi intent also pose the challenge of addressing the order of execution of intents found. Hence, we are proposing a unified architecture to address multi intent detection, associated slots detection and order of execution of found intents using low proportion multi-intent corpus in the training data. This architecture consists of Multi Word Importance relation propagator using Multi Head GRU and Importance learner propagator module using self-attention. This architecture has beaten state of the art by 2.58% on MultiIntentData dataset.

## 1 Introduction

Many voice assistants like Samsung Bixby, Amazon Alexa, Microsoft Cortana, Google Assistant has provided voice solution to ease the phone usage for the users. To make user experience more conversational rather than command oriented, exploration on handling multi intent by NLU is increasing. NLU currently handles three important task for identification. Domain Detector (DD) is the task of identifying which domain or application should execute the utterance. ID is the task of identifying what is the intent of the user from the utterance. SF is the task of identifying the objects of interest (named entities) on which we execute the intent operation. For multi-intent, ID task involves identification of one or more intents in the utterance told by the user. Hence, ID must be able to identify the dynamic number of intents present in the utterance along with identification of the boundaries or segments for each intents. In addition, the order of execution of the identified intents matters as one intent execution might be dependent on the other intent execution. Lastly, we would like to have low proportion of training data for multi–intent so that we reduce the dependency on data generation and maintenance. Hence, we propose a unified architecture that address the following problems: multi intent identification, slot identification, multi-intent boundary detection and execution order of intents.

Lots of work has happened in the area of single intent and slot. Chen et al. (2019) proposes the exploration of BERT architecture for NLU task where pre-trained bi-directional representation on unlabeled corpus, with simple fine tuning, aided in the task of combined single intent prediction and slot detection. E et al. (2019) offer bi-directional interrelated information sharing between intent learnings and slot learnings. In addition, they use new iteration mechanism to enhance the sharing of the learnings. Chen and Yu (2019) project the usage of word attention, calculated using word embedding, in addition to semantic level attention at each decoding step of Bi-LSTM. They also use fusion gate for fusing the intent and slot learnings for enhancing the relationships between intent and slot. Bhasin et al. (2020) recommend the use of Multi modal Bi-Linear Pooling technique for fusing the learnings of intent and slot. Tingting et al. (2019) outlines the usage of Bi-LSTM along with attention for jointly learning the learnings of intent and slot.Xu and Sarikaya (2013a) recommends the usage of convolutional neural network [CNN] along with tri-crf for the joint task of intent detection and slot learning. Liu and Lane (2016a) recommends the usage of attention information along with recur-



*Proceedings of the ICON 2020 Workshop on Joint NLP Modelling for Conversational AI*, pages 10–18
Patna, India, December 18 - 21, 2020. 

rent neural networks in encoder-decoder stage that enhances the learnings of intent and slot. Wang et al. (2018b) recommend the usage of encoder for encoding the sentence representation using CNN , for local feature and high level phrase representation, and Bi-LSTM for capturing contextual semantic information and decoding using attention information, calculated from encoder, in each decoding step of Long Short term Memory [LSTM] decoder. Liu and Lane (2016b) offers the usage of recurrent neural network[RNN] for solving the problem of joint intent and slot by updating the intent detection as and when words are coming from the utterance. Wang et al. (2018a) suggest the usage of bi-model network where two parallel Bi-LSTM are used and they use the hidden information of one Bi-LSTM to another in each network. Then they use the learnings to predict intent and slot from each network. Yu et al. (2018) offer to use cross-attentive information propagation for enhancing the meaning of the word at word level as well as tagging level to aid the task of joint intent and slot prediction. The above networks has explored many ways of capturing important word level information and fusing the learnings for predicting single intent and slot. Inspired on the information captured, explorations on multi intent detection and slots have gathered steam recently to make the task finding generic.

Gangadharaiah and Narayanaswamy (2019) suggest Bi-LSTM encoder for encoding the sentence information and uses sentence level attention information as well as word-level attention information for each time step in both the decoders. One decoder predicts word-level intent detection, another decoder predicts word-level slot detection and one feed forward neural network predicts sentence-level intent prediction. This network suffers from contiguous boundary utterance detection. Xu and Sarikaya (2013b) suggests the usage of share information between multiple intents to identify segments belonging to each intent by using hidden layer to map the learning of word importance to intent prediction. The network is very shallow and is unable to capture the long distance word relationship. Kim et al. (2017) suggest usage of two-stage system to detect multiple intents in a single utterance when the model is trained with single utterance by first breaking the utterance into two chunks in the first stage and processing each chunk sequentially by the model. This method suffers from pipeline approach where error in first stage results in propagative error in model stage. The exploration, in this area, is very less comapred to single intent and slot, dur to the absence of proper open source dataset for multi intent data.Hence, we are proposing a novel architecture, which will address the following: multiple intent detection, intent segmentation or boundaries and execution order of the found multi intents and slot prediction, where execution order determines the relationship between intents to derive sequence of execution on the voice assistant system.All the experiments are run on MultiIntentData dataset, a newly deveoped dataset.

We first address the problem statement in detail in Dataset Section, and then followed by Dataset Pre-Processing for extracting the required information, architecture explanations, results discussion and finally Conclusion.

## 2 Dataset

We solve four types of problem in this paper namely word-level intent prediction, sentence-level intent prediction, word-level order prediction, and word-level slot prediction. We use word-level intent prediction for finding the boundaries or segments of multiple intents present in the sentence as shown in Figure 1

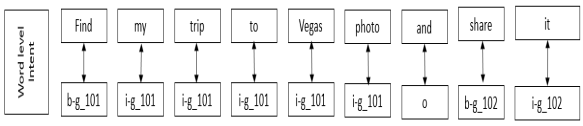


Figure 1: Word level Intent Prediction

We use sentence-level intent prediction for identifying all the intents present in the sentence as shown in Figure 2

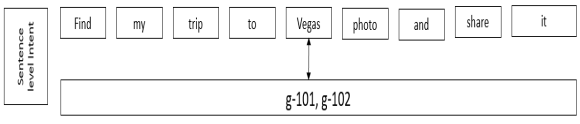


Figure 2: Sentence level Intent Prediction

We use word-level order prediction for finding out the order in which the intent segments or intent boundaries must execute as shown in Figure 3

We use word-level slot prediction for finding all the slots in the sentence as shown in Figure 4.

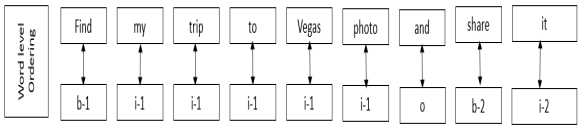


Figure 3: Word level Order Prediction

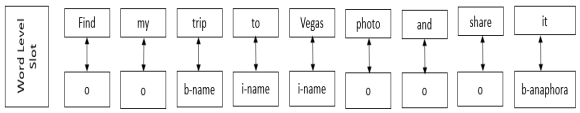


Figure 4: Word level slot prediction

For handling very less multi-intent training data, we have created new training and test dataset with the help of linguist namely MultiIntentData. MultiIntentData dataset was created using single intent information from two domains namely Gallery and Camera.

Gallery domain contains the information shown in Table 1.

| Intent ID | Description |
|---|---|
| g-101 | Open gallery with or without using picture name, album name and folder name |
| g-102 | Share the pictures found using picture name, album name and folder name |
| g-219 | Add the pictures found using picture name, album name and folder name to wallpaper |

Table 1: Gallery domain information.

Camera domain contains the information shown in Table 2.

| Intent ID | Description |
|---|---|
| c-1 | Open camera |
| c-176 | Turn on flash feature in camera |
| c-23 | Change the picture size of front or rear camera and take picture |
| c-3 | Change the modes of camera and take picture |
| c-408 | Create emoji using the taken picture |

Table 2: Camera domain information.

The data is created for natural forms of multi-intent voice queries while also ensuring the intent order and dependencies are ensured. For example, we created continuous sentences without any separators between individual intents. Example is "Take the shot in pro mode with the flash". Here the user is requesting to turn on flash in camera and then set the mode to pro before taking picture. In this utterance, there are no separators. Considering the constraints, linguist has created data using four types of combination from single intent data of two domains as shown in Table 3. The created

| Combination Type | Example Utterance |
|---|---|
| Independent intents within domain | share Malibu pictures and add the latest paper to wallpaper |
| Independent intents across domain | open camera after launching gallery |
| Dependent intents within domain | find latest Malibu pictures and share it |
| Dependent intents across domain | take a pic using selfie mode and share it |

Table 3: Intent Combinations with example.

data[1] contains 1896 training utterances and 350 test utterances. The training data contains 8% multi-intent training data and 92% single-intent training data using intents from two domains mentioned in Table 1 and Table 2. The test data contains 92% multi-intent data and 8% single intent data.

## 3 Proposed Method

This section explains Data Preprocessing followed by Model 1, Model 2 and Model 3.

### 3.1 Data Preprocessing

The training data and test data are present in the format as shown in Figure 5.

<[view my {farewell}(name) album](G_101)>(1) before <[launching the camera](C_1)>(2)

Figure 5: Word level Order Prediction

We write each utterance inside the square brackets followed by intent id inside parenthesis. This represents intent information.

In addition, we write each slot phrase in the utterance by curly brackets followed by slot id inside parenthesis. This represents slot information.

Finally each intent information is written inside the "<" and ">" symbols followed by order id, a number inside parenthesis. This represents order information.

[1]https://github.com/MultiIntentData/MultiIntentData

To get the utterance with intent, slot and order details do the following. First, we extract the order information by using regular expression to search for first encountered "<" and ">" followed by parenthesis. The order information contains intent information inside the "<" and ">" and order id. We extract one or more slot information from intent information by using regular expression to find all the left curly braces and matching right curly braces along with parenthesis. This will give list of (slot phrase, slot id) tuples. Using this list, we generate the original utterance by replacing all the slot information with corresponding slot phrases. Then we use another regular expression to search for left square bracket with matching right square bracket along with parenthesis. This provides (utterance, Intent ID) tuple for intent information. Hence the order tuple becomes ((utterance, Intent ID), Order ID) where the utterance has the intent as Intent ID and order as Order ID. Now we assign Intent Id and order ID for each word in the utterance in IBO format to generate word-level intent information and word-level order information. From the list of (slot phrase, slot ID) tuples , we create IBO format for slots where the phrases, from the utterance, not in the slot phrase are assigned "o" and phrases in slot phrase are assigned Slot ID with first word as "b-Slot ID" and rest of the slot phrases as "i-Slot ID" to generate word-level slot information. We repeat the above steps for another order information within the utterance (If present).

If more than one order is present then there might be phrases not belonging to any order. In such cases, we assign those phrases as "o" for word-level intent, word-level slot and word-level order information. We concatenate the utterances generated in the process. The list of intent ID(s), generated by parsing multiple order information, is assigned as label for the final utterance generated for sentence level intent information.

The next section explains model architecture evolutions.

## 3.2 Model 1: GRU learner enhancer with Self Attention

Figure 6 shows the architecture of the proposed model. The model is explained in the following subsections.

### 3.2.1 Utterance Pre-processing

First, we count the number of words in the utterance (W1). If the count is less than max length

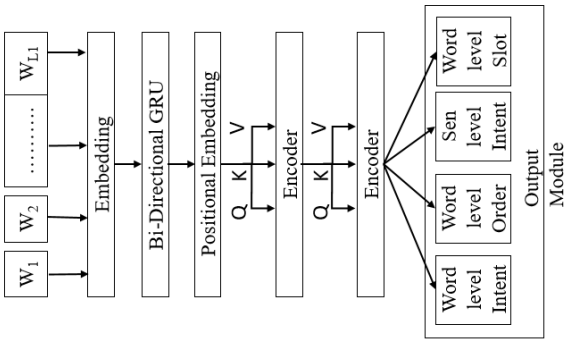


Figure 6: Bi-directional GRU with two Encoders

(L1), then we append the utterance with (W1 –L1) padding words. We use "¡pad¿" symbol as the padding word. Then we index individual word in the utterance using training dictionary. In training dictionary, we assign ¡pad¿ symbol with index 0 and other words (including "unk" word) are indexed one to "N - 1" (N -¿ max number of words in the dictionary). If we do not find the word in the training dictionary, then we assign index of "unk" word. We pass indexed words of the utterance to embedding layer. We use L1 as 23.

### 3.2.2 Embedding Layer

Embedding layer contains the weight matrix of each index to vector of L2 dimensions. Its dimension is N * L2. We pass each indexed word through the weight matrix to get its vector of L2 dimensions. Since there are L1 words present in the utterance, we get L1 * L2 matrix. Then we pass this matrix to single GRU unit. We use glove embedding of size 300 dimension to map word index to its glove-embedding vector of 300 dimension. We assign "unk" word with random initialization of 300-dimension vector. Hence L2 is 300.

### 3.2.3 Bi-directional GRU Layer

Gated Recurrent Unit (GRU) layer is a gated mechanism, where it uses reset gate and update gate for information propagation at each time step.

The update gate decides what information needs to propagate by passing previous hidden state and current input through sigmoid function.

Sigmoid function squashes the value between zero and one. If the value is closer to zero then we do not propagate the info. If the value is closer to one we propagate the info.

The reset gate decides what information from the past needs to propagate by passing the previous hidden state and current input through sigmoid

function and multiplying the output with previous hidden state. The output, calculated using sigmoid function, contains what value needs to stay and what value needs to forget. Multiplying the output with previous hidden state updates the past information propagation as shown in Equation 1.

$$\begin{aligned} z_t &= \sigma_g(W_z * x_t + U_z * h_{t-1} + b_z) \\ r_t &= \sigma_g(W_r * x_t + U_r * h_{t-1} + b_r) \\ h_t &= z_t * h_{t-1} + (1-z_t)* \\ &\quad \phi_h(W_h X_t + U_h(r_t * h_{t-1}) + b_h) \end{aligned} \tag{1}$$

where $z_t$ represent update gate and $r_t$ represent reset gate. We use bi-directional GRU where we concatenate the outputs of forward GRU and backward GRU.

We pass the concatenated output of Bi-directional GRU after adding with trigonometric position embedding. Trigonometric positional embedding generates alternate sine and cosine embedding taking position to generate embedding. We pass the combined embedding to transformer encoder module. We use 256 as hidden dimension of Bi-directional GRU.

#### 3.2.4 Transformer Encoder

We use transformer encoder module (Devlin et al., 2018) using multi-head attention layer, positional feed forward layer, and residual connection layer followed by normalization. We use two encoder modules.

Multi Head attention network splits the input embedding into "n" equal chunks and we provide each chunk as input to self-attention. Self-attention layer aides in enhancement of word importance over the entire sentence. We achieve this by providing the encoded input representation as a set of key-value pair. Then we provide query as same encoded input sentence. We use scalar dot product attention where we apply dot product between query and all the keys to provide weighted sum and then we multiply with value to provide weighted sum of the value. We concatenate the "n" self-attention outputs. This output contains the information importance from different subspaces from different positions. We pass this output to residual connection module. We use "n" as 16. This module produces 256 as hidden dimension output.

Residual connection module takes the input and output of multi-head attention module as input to this module and apply element wise addition on this module. We give the output to Normalization layer. This module produces 256 as hidden dimension output.

Normalization layer apply normalization on the input layers. We provide the output to position wise feed forward neural network. This module produces 256 as hidden dimension output.

Position wise feed forward neural network enhance the learning of the word level importance. This module produces 256 as hidden dimension output.

We pass the output of Positional Feed Forward neural network as input to another encoder and repeat above steps. The output of the second encoder contains 256 as hidden dimension.

We finally pass the output of second Positional Feed forward neural network to output module.

#### 3.2.5 Output module

The module consist of four fully connected feed forward neural networks. Each feed forward network module predicts word level slot, word level intent, word level order and sentence level intent using softmax on the first three networks and sigmoid on the last one. The first three provides multi class classification and last one provides multi label classification. There are 17 word-level intent, 5 word-level order, 20 word-level slot and 8 sentence-level intents to identify.

#### 3.2.6 Analysis

The proposed Model-1 able to handle simple relationship between the intents and orders.

However, it is not able to capture the relationship between intent and order boundaries properly. Increase in hidden dimension leads to poor performance of the model due to less training data. In addition, the model is not able to identify the boundaries of slot properly. Finally, the model is not able to differentiate between the presence of word as part of separator and presence of word as part of open title type slot.

Consider the example "display camera app to make me a crazy emoji". In this, the word-level intent prediction segments the sentence properly.

Whereas the word-level order prediction is not segmenting the sentence properly. It predicts only one order when the intent is clearly presenting two intents as shown in Figure 7 and Figure 8.

This clearly shows that common module is not sufficient to use both intent as well as order prediction. The fact that the intent and order predictions

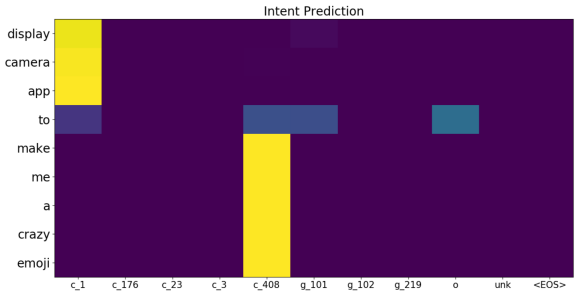


Figure 7: Intent Prediction. Brighter color indicates stronger relationship

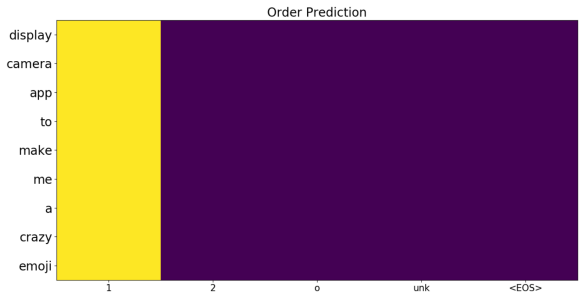


Figure 8: Order Prediction. Brighter color indicates stronger relationship

are inter-related propagated the idea of Model 2 explained in the next section.

### 3.3 Model 2 : Multi Head GRU with Selective Learning Propagation block

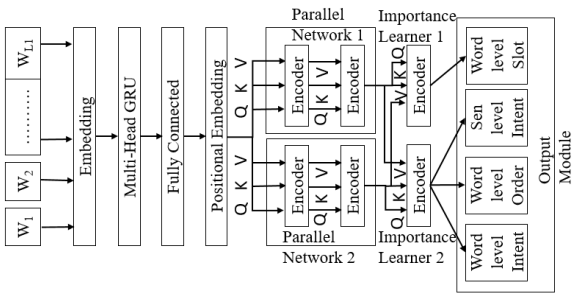


Figure 9: Multi Head GRU with Selective Learning Propagation

Figure 9 shows the architecture of the proposed model.

First, we process the utterance as explained in 3.2.1 section. Then we pass the output, indexed words of length L1, to embedding layer. Embedding layer process the indexed utterance and generates the embedding vector (L2) for each word as explained in 3.2.2 section. Now we pass this to Multi-Head GRU module, which we explain in the next section

#### 3.3.1 Multi Head GRU

We split the input embedding into four equal chunks. We give each chunk to one bi-directional GRU. Bi-directional GRU provides contextual information of the current word with respect to future data and past data. We explained this in 3.2.3 section. We concatenate the output of each parallel Bi-directional GRU. The hypothesis behind Multi Head GRU is we capture different phrase importance from different positions. Then we pass through fully connected network to pick the important phrases from concatenated output. We sent the fully connected network output to parallel network, which we explain in the next section. We use 256 as hidden dimensional information.

#### 3.3.2 Parallel Network modules

The module has two parallel transformer encoder block. We have explained the working of Transformer Encoder block in 3.2.4 section. The reason for two parallel networks is different learning representation is available for the same input. In addition, we use one network for predicting the intent related predictions and other for slot related predictions. This makes each network learnings to concentrate on related task only thereby dividing the learnings of the task between the two networks. Now we selectively propagate the learnings of each network for each task using Learner module, which we explain, in the next section.

#### 3.3.3 Importance Learner Module

This module is the most important module. It takes the output of two parallel network modules and selects the information from one parallel network module to enhance the learning of another parallel network module. We achieve this using the self-attention block in transformer encoder module where we use query and key as the output of one parallel network module and value as other parallel network module instead of passing the same input as query, key and value. We have explained the working of self-attention block in 3.2.4 section. Since there are two networks that requires learning enhancement we use two learning module. Importance Learner 1 takes query and key as output of parallel network module 2 and value as output of parallel network module 1. Importance Learner 2 takes query and key as output of parallel network module 1 and value as output of parallel network module 2. We provide the output of Importance Learner 2 to Output module to predict word-level

intent, word-level order and sentence-level intent. Similarly, we provide the output of Importance Learner 1 to output module to predict word-level slot.

Output module takes the output of Importance Learner 1 and Importance Learner 2 and does final prediction. We have explained the working of Output module in 3.2.5 section.

#### 3.3.4 Analysis

The model is able to differentiate the boundaries of open title kind of slots properly. In addition, the model is able to understand the difference between the words being part of open title slot and the words acting as separator.

There is improvement in the relationship understanding between intent and order boundaries. However, the model is suffering from order mix-up within the boundary. However, the model is suffering from order mix-up within the boundary. In addition, confusion point exist between open title slots. Consider the example "click photo in food mode after turning on flash". In this, the model is able to find the intent boundaries properly but the order boundaries is not proper due to relationship misunderstanding between "click photo" and "after" as shown in Figure 10 and Figure 11.

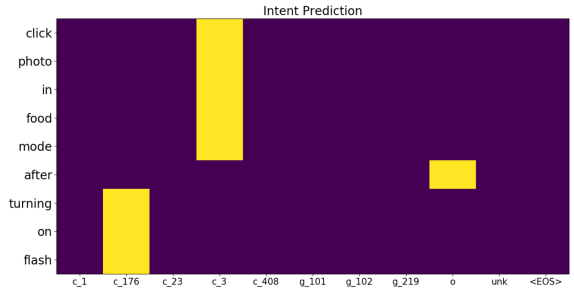


Figure 10: Intent Prediction. Brighter color indicates stronger relationship

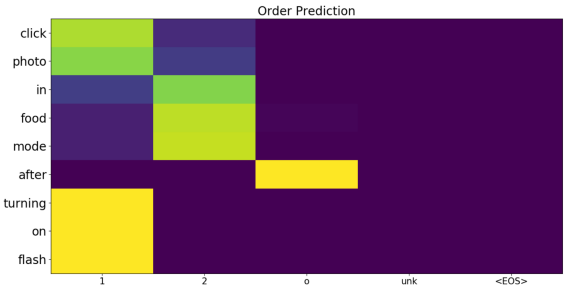


Figure 11: Order Prediction. Brighter color indicates stronger relationship

The limitations clearly shows the need for new way of addressing the problem of intent and order boundary relationship as well as improvement in open title slot detection.

### 3.4 Model 3: Multi Head GRU with Importance Learner Module and Order processing

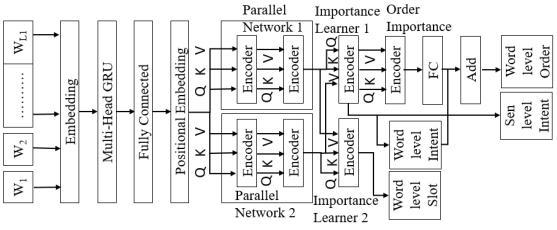


Figure 12: Multi Head GRU with Importance Learner module and Order Importance Module

Figure 12 shows the proposed architecture of Model 3.

We pre-process the input utterance as mentioned in 3.2.1 section. We get output as indexed words for the utterance whose count will be L1.

We pass the utterance containing indexed words of length L1 to embedding layer. Embedding layer assigns vector to each word as explained in 3.2.2 section. The output will be a matrix of L1 words with each word having embedding vector of length L2. Hence its dimension will be L1 * L2. We pass the matrix to Parallel network modules.

Parallel network modules generate different representational information for the same input as well as information learning is divided between the networks as explained in 3.3.2 section. Both the outputs are provided as input to Importance Learner module.

Importance Learner module selectively chooses the learning of one network to influence the learning of other network as explained in 3.3.3 module. Since there are two networks, we use two Importance Learner modules separately to enhance each network learnings. The output of Importance Learner 1 is given to Order Importance learner module as explained in the next section. We also use the output to predict word level intent and sentence-level intent.

#### 3.4.1 Order Importance learner module

This module enhances the learning by passing through another transformer encoder layer where we use the same input as query, key and value. The working of transformer encoder is explained

| Architecture | Slot Sentence level Acc | Word Intent Sentence level Acc | Sen Intent Sentence level Acc | Word Order Sentence level Acc | Overall Sentence Level Acc |
|---|---|---|---|---|---|
| Model 1 | 89.14 | 88 | 88.57 | 86.57 | 86.57 |
| Model 2 | 91.43 | 88.29 | 88.86 | 89.14 | 88.29 |
| Model 3 | **94.86** | **90.29** | **90.29** | **90.29** | **90.29** |
| Gangadharaiah and Narayanaswamy (2019) | 90.86 | 89.14 | 89.71 | 87.71 | 87.71 |

Table 4: Comparison of state of the art models.

in 3.2.4 section.We add the module output with word-level intent output. For this, we reduce the hidden dimension of the network to intent number by passing through intent fully connected network with "relu" activation.

The Order Importance Learner module output predict word-level order by passing through fully connected layer with softmax as activation.

The Importance Learner 2 predicts word-level slot.

### 3.4.2 Analysis

The model, when ran on MultiIntentData dataset, is able to differentiate the open title slots well. In addition, the slot boundaries have improved. In addition, the word differentiation as part of open title or part of separator is able to identify properly. In addition, we are able to see huge improvement in intent and order boundary understanding. Finally, the model has reduced the confusion within order boundaries.

There is a need of improvement in better intent detection and slot detection for few cases.

## 4 Result and Discussion

We ran all the three models using MultiIntentData dataset. We modified Rashmi and Narayanaswamy (2019) architecture to predict order as well, similar to prediction of word-level intent by adding new decoder and providing same attention information for each decoder step to predict word-level order, and ran on MultiIntentData dataset to compare with state of the art. All the result are captured in Table 4.

From the table we are able to beat the state of the art architecture by 2.58%. This is attributed to the fact that parallel network along with importance learner module is able to enhance the learning of intent, slot and order when compared to unified architecture proposed in Gangadharaiah and Narayanaswamy (2019). In addition, the word differentiation between part of the catchall and part of the separator is handled well by Model 3.

From the result in Table 4, we are able to understand that Model 3 is the best performing model over Model 1 and Model 2 by 3.72% and 2% respectively. From the result, we are able to see that Model 3 is able to perform open slot distinction i.e. distinction between the slots, open slot boundary detection and word boundary detection between part of open slot and part of separator over Model 1. For more details please have a look at 3.2.6 and 3.4.2. Model 3 is able to improve the order and intent boundaries well as well as confusion points between open title slots. For more details, please see 3.3.4 and 3.4.2.

## 5 Conclusion

This work showed the importance of multi intent detection with associated slots and its order of execution over single intent and slot. This also showed the importance of multi intent learning using low corpus data. We are able to derive that Multi Head GRU aide in better contextual understanding of the input embedding representation. In addition, the presence of parallel network for intent and slot learning, along with two-importance learner module has shown better understanding on the differentiation between boundaries of the slot and start of the next intent. In addition, the word level intent aided in influencing the overall sentence level intent. The word level intent as well as the separator also influence the intent ordering execution. This has resulted in improvement to the tune of 3.72%. Multi head self-attention learning on context aided in improving from state of the art model by 2.58%.

Future scope is to resolve anaphora resolution of slots within intent and across intent.

## References


Anmol Bhasin, Bharatram Natarajan, Gaurav Mathur, and Himanshu Mangla. 2020. Parallel intent and slot prediction using mlb fusion. In *2020 IEEE 14th International Conference on Semantic Computing (ICSC)*, pages 217–220. IEEE.

Qian Chen, Zhu Zhuo, and Wen Wang. 2019. Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Sixuan Chen and Shuai Yu. 2019. Wais: Word attention for joint intent detection and slot filling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 9927–9928.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Haihong E, Peiqing Niu, Zhongfu Chen, and Meina Song. 2019. A novel bi-directional interrelated model for joint intent detection and slot filling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5467–5471.

Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. 2019. Joint multiple intent detection and slot labeling for goal-oriented dialog. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 564–569.

Byeongchang Kim, Seonghan Ryu, and Gary Geunbae Lee. 2017. Two-stage multi-intent detection for spoken language understanding. *Multimedia Tools and Applications*, 76(9):11377–11390.

Bing Liu and Ian Lane. 2016a. Attention-based recurrent neural network models for joint intent detection and slot filling. *arXiv preprint arXiv:1609.01454*.

Bing Liu and Ian Lane. 2016b. Joint online spoken language understanding and language modeling with recurrent neural networks. *arXiv preprint arXiv:1609.01462*.

Chen Tingting, Lin Min, and Li Yanling. 2019. Joint intention detection and semantic slot filling based on blstm and attention. In *2019 IEEE 4th international conference on cloud computing and big data analysis (ICCCBDA)*, pages 690–694. IEEE.

Yu Wang, Yilin Shen, and Hongxia Jin. 2018a. A bi-model based rnn semantic frame parsing model for intent detection and slot filling. *arXiv preprint arXiv:1812.10235*.

Yufan Wang, Li Tang, and Tingting He. 2018b. Attention-based cnn-blstm networks for joint intent detection and slot filling. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, pages 250–261. Springer.

Puyang Xu and Ruhi Sarikaya. 2013a. Convolutional neural network based triangular crf for joint intent detection and slot filling. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 78–83.

Puyang Xu and Ruhi Sarikaya. 2013b. Exploiting shared information for multi-intent natural language sentence classification. In *INTERSPEECH*, pages 3785–3789.

Shuai Yu, Lei Shen, Pengcheng Zhu, and Jiansong Chen. 2018. Acjis: A novel attentive cross approach for joint intent detection and slot filling. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE.