Intent Classification Using BERT

Kirti Desai
UFID# 0454-1017
University of Florida

***Abstract*— The ultimate destination of Natural language processing (NLP) is to ability of read, classify and be able to understand and extract meaningful information which can be used to in wide array of applications. At the center of this objective is to extract language features and inter-dependencies of different words in individual sentences, to be able to learn contextually. Those features can be used for a number of downstream NLP tasks. Under supervised learning, one such task is to be able to classify sentences or documents into different types or labels. Based on problem statement those can be classified into binary classification or multi-label classification.**

The real-world application such as spoken dialog systems has more of multi-label intent classification problems where it can be pretty complex to understand intent based on spontaneous speech. This paper aims to present the Intent Classification (IC) with near state of the art results over airline travel information system dataset (ATIS) using a novel pre-trained model called BERT (Bidirectional Encoder Representations from Transformers.

I. Introduction

Natural Language Processing (NLP) is at the heart of human-computer interaction. NLP is the way of making a machine understand the human languages and able to communicate with them. It has brought reforms in every industry like diagnosis in healthcare, semantic analysis and cognitive assistant. Most of the modern query documents can be classified into more than one class label and classification tasks has always been center of array of Natural Language Processing tasks. In the classification domain until recently, all the architectural models were limited by the locating the dependencies of words in one direction only and suffered from short-term memories for example sequence to sequence [1] models like LSTM (Long Short Term) and GRU (Gated recurrent units). They were also very expensive computationally as a lot of time and efforts was put into training the models. They were all sequential models taking a toll on time. Models like Extended Neural GPU were developed focused on parallel computations by calculating hidden states in parallel. They improved the training time but it made hard for these models to find contextual dependencies.

Attention [6] was a major break-through in the language representation model. There is an additional Attention layer in decoder which calculates an attention vector based on all the hidden states of the encoder rather than just the last one. Using self-attention, Transformer model [6] calculates relative position of words in the sequence which proved to perform well in comprehension related tasks. It also boosted the training speed significantly using parallel computation. As shown in 1 It uses a stack of encoders, each with self-attention layer and feed forward network layer.

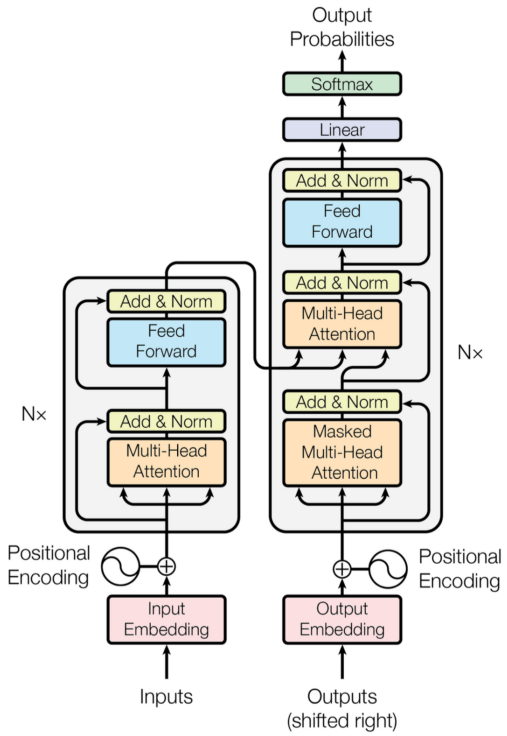

Figure 1: The Transformer - model architecture.

Fig. 1: Transformer Encoder Architecture (source: Transformer original paper)

BERT [4] acted as swiss army knife for modern NLP tasks. Previous all models were able to find the contextual dependencies only in one direction. But BERT is fully bi-directional.It is developed constructively using previously known intelligent ideas of language representational models. It is a pre-trained transformer encoder stack which is available in different sizes with varying size of encoder layers and dimensions of hidden states. Using BERT is inexpensive and time saving as it just needs an additional layer and fine-tuning which can be trained in less than two hours. In real-world scenarios, there can be very tough challenges to classify an utterance under one label as each sentence can contain multiple intents. Among a number of downstream NLP tasks, Intent classification and slot filling 7 is essential and being used extensively. We will show near SOTA (state-of-the-art)

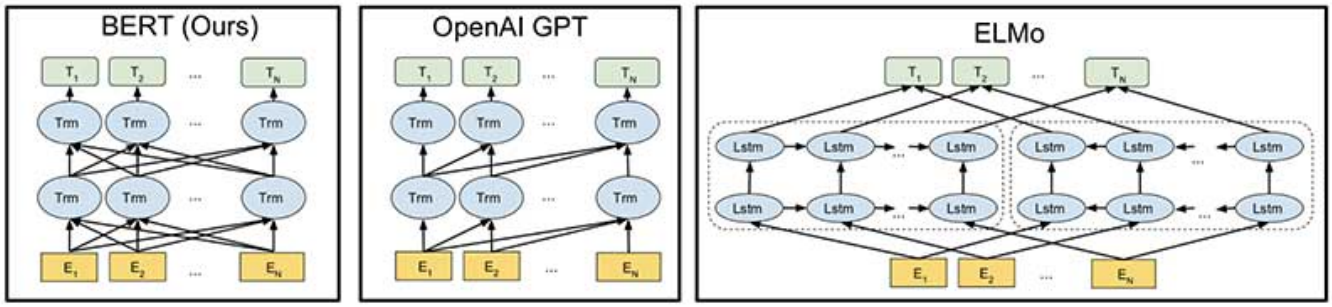

Fig. 2: Architecture of BERT, OpenAI GPT and ELMo

results in this paper while classifying multiple intents in single utterance. We will use the benchmark dataset ATIS 5, and will classify intent of each query. We achieved accuracy of 96% over test set.

II. Related work and previous achievements

By the time of start of AI assistants and automating the support channels, intent classification has been a central task. There are multiple work that has been done on it. We can divide building these model into mainly two blocks:

i Word Embeddings
ii Neural Network Models

All data that to be feeded into neural net needs to be converted into vectors of token ids. Every model has word embedding layer as the input layer to take input data (queries). These embeddings greatly influence the results, so a lot of efforts are put into preprocessing the data. There are some popular pre-trained word embeddings present like Word2vec 10, Global vectors for word representations (Glove) 11 and fastText 12. Word2vec uses neural net models like continuous bag-of-words (CBOW) and skip-gram to train and find the analogy between the words. These are log-linear models. Glove is built on a large corpus of word to word dataset and established relations between different words based on contexts. It was trained with unsupervised algorithm and finds probability of word to word co-ocurrences. Both of these approaches gives similar results. fastText is an extension to Word2vec. It also uses skim-gram model but improves it by breaking word into characters which in turn helped to learn prefixes and suffixes. All these embedding models provide a better language representation and contextual dependencies which can be used in different applications to save time and improve accuracy.

There have been multiple experiments with neural networks to implement the downstream tasks, some of them are following:

i Bi-directional Long Short Term Memory (BLSTM) : In a paper by Sreelakshmi et al., 13, they propsed a bi-directional sequential model to learn the semantics and features. Sequence to Sequence models are known for remembering the long distances dependencies. A single direction LSTM model is based on RNN, which is able to learn the word contexts only in one direction. This novel approach improved the accuracy results to 98.221%. They developed their own symmentical reach embeddings using Glove and BLSTM abled it to detect non-consectuive word relations. Architecture of BLSTM is shown in fig: 3

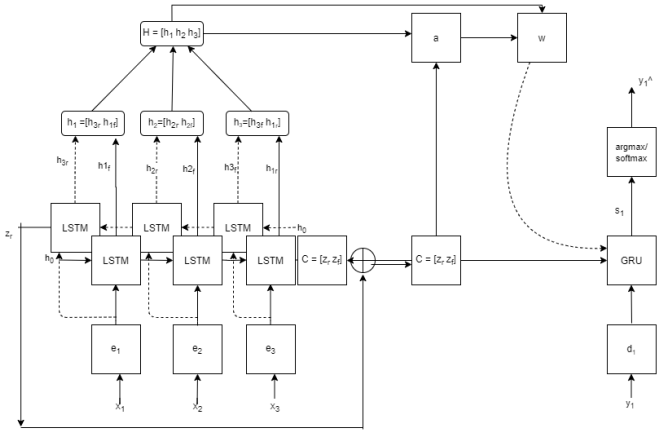

Fig. 3: Bi-directional LSTM Network

ii Attention based Bi-directional RNN (BiRNN) : In a paper by Bing Liu et al., 14, they proposed a attention based bi-direction RNN model to improve the understanding on speech and dialogue systems and incorporating the alignment information embedded in the queries through slots.They used LSTM cells as the RNN encoder and embeddded the attention so that while decoding they predicted the sentence by adding the weighted average of all the hidden states of sequence which are combination of forward and backward direction. They experimented their model on ATIS dataset to achieve accuracy score of 95.75.

OpenAI GPT language model is also a variation in language representational models which is solely based on predicting the upcoming word. This is an transformer based encoder model which computes the probability of all other words and predict one output. It is very useful in predicting long sentence comprehension based on any length of input sentence. BERT is direct descendent of GPT, which also used bi-directional feature.

Elmo AI is another deep contextualized language representation model using Bi-directional LSTM.This is also built following the objective of language modeling. To predict next

word, It concatenate the hidden states both left and right, and it uses character based contextual meaning makes it stronger semantically. Though this uses a Bi-directional LSTM, it is not able to learn from both left and right simultaneously. Cmpared to this, BERT uses masked language modeling, which takes account of both left and right words corresponding to the masked token.

III. Dataset

The Airline Travel Information System (ATIS) dataset is built on spontaneous speech corpus has been proven to be immensely valuable in spoken language understanding (SLU). In the training set of ATIS, there are 4978 samples with vocab size of 943 as well as there are 129 different slots and 26 intents. Test set is consisted of 893 samples, 129 slots and 26 intents. The intent distribution over the sentences is presented in Figure 4 and Figure 5.

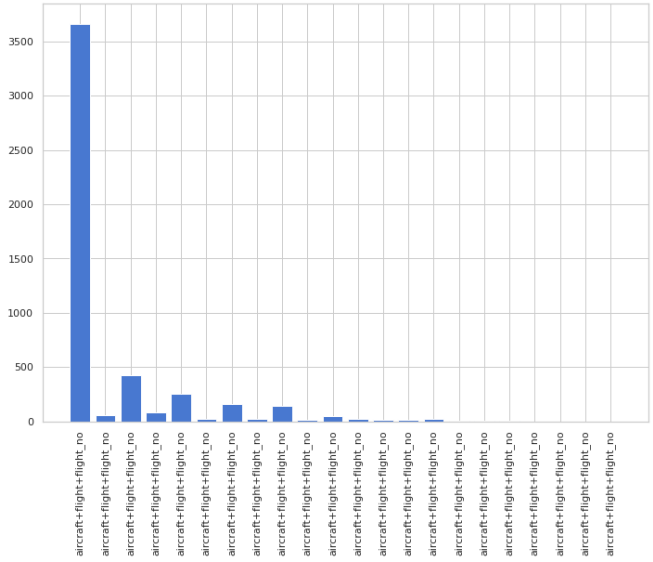

Fig. 4: Intent Distribution over Training Sentences

IV. Description

A. Data Preprocessing

The first phase towards training the model is to preprocess the data which includes tokenization of all the sentences and padding all the sentences to make the data compatible to BERT input. We will use tokenizer provided by BERT to transform the data into acceptable format as BERT expects tokenized data with start of sentence ([CLS]) and end of senetence ([SEP]). We convert the words into tokens as BERT's vocabulary.

Next Step is to vectorize the data according to IDs in the BERT tokenizer vocabulary. All sentence are padded with '[PAD]' token which is at index 0. Then we split the data into train, validation and test set in the ration of 90 to 10.

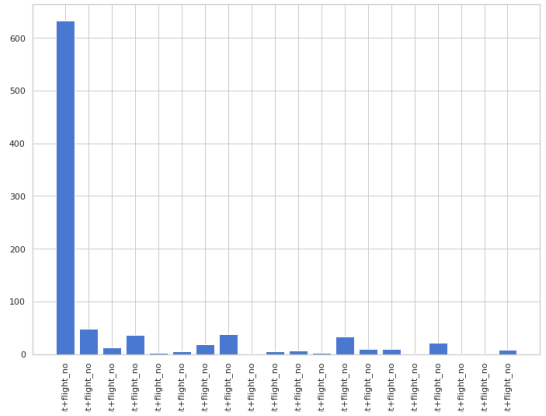

Fig. 5: Intent Distribution over Test Sentences

```
Input sentence:
 [CLS]  what is the arrival time in san francisco for the 755 am flight
leaving washington  [SEP]
Tokenized Sentence:
 ['[CLS]', 'what', 'is', 'the', 'arrival', 'time', 'in', 'san', 'francisco',
'for', 'the', '75', '##5', 'am', 'flight', 'leaving', 'washington', '[SEP]']
```

Fig. 6: Tokenization example of one input sentence

```
--------------------------------------------------------------------------------
Query text: BOS flights from dc to denver EOS
Query vector:  [178 429 444 344 851 351 179]
Intent label:  flight
Slot text:  O O O B-fromloc.state_code O B-toloc.city_name O
Slot vector:  [128, 128, 128, 49, 128, 78, 128]
********************************************************************************
Query text: BOS i would like to fly from pittsburgh to san francisco EOS
Query vector:  [178 479 932 545 851 431 444 682 851 739 440 179]
Intent label:  flight
Slot text:  O O O O O O O B-fromloc.city_name O B-toloc.city_name I-toloc.city_name O
Slot vector:  [128, 128, 128, 128, 128, 128, 128, 48, 128, 78, 125, 128]
--------------------------------------------------------------------------------
```

Fig. 7: Tokenization example of one input sentence

B. Model Specifications

BERT 4 is a open source pre-trained transformer Encoder stack model. It is successor of OpenAI GT and alleviate the limitation of sequence to sequence of models. BERT acted as swiss army knife for modern NLP. Previous all models were able to find the contextual dependencies only in one direction. But BERT is fully bi-directional.It is developed constructively using previously known clever tricks of language representation models. It is available in different sizes with varying size of encoder layers and dimensions of hidden states. Though there are multiple versions, two being the standard one are following:

1) BERT-Large, Cased (Whole Word Masking): 24-layer, 1024-hidden, 16-heads, 340M parameters
2) BERT-Base, Uncased: 12-layer, 768-hidden, 12-heads, 110M parameters

For this intent-classification task we have used the 2nd one, BERT-Base. To develop the BERT, it was given two fundamental tasks.

1) Mask Language Model: While training, for every input sentence BERT randomly masks the 15% of the tokens with [MASK] token and then predict these tokens by passing these vectors through softmax layer. Now these

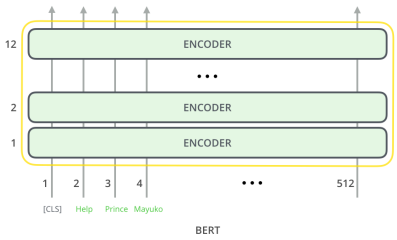

Fig. 8: Bert Encoder Input

masked tokens are predicted based on the other words in the sentence. BERT used a variation of MLM as by replacing a particular chosen token only 80% of the time and 10% some random word and 10% the same token.

2) Next Sentence Prediction: Second auxiliary task to train BERT is a binary classifier for a sentence being next to another. Recording the sentence relation is very essential for a Question-Answering Systems and Natural Language Interfaces. BERT was pre-trained with sentence corpus such that 50% of time Q sentence was next to P and other times there was random sentence.

Input embedding layer for BERT consist of three layers which are token embedding layer, segment embedding layer, Positional embedding layer. Token embedding layer is built upon Wordpiece model such that words are broken down into sub-words to handle them better.

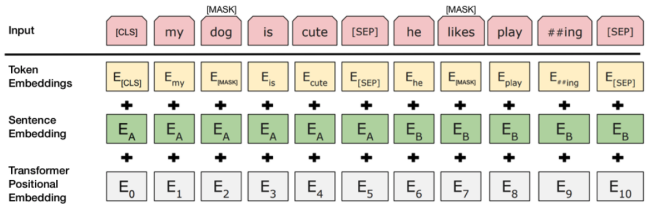

Fig. 9: BERT Embedding Layer

BERT can be used for wide array of NLP tasks with one fine-tuned additional layer on top of actual model.

1) Sequence Classification Tasks
2) Sentence Pair Classification Tasks
3) Question-Answering Tasks
4) Single Sentence Tagging Tasks

For our project, we have to apply fine-tuning minimally for one of the tasks named sequence classification task. We will use attention masking with all the words are candidate to be predicted, so we mask all the tokens. We add a classification layer using BertForSequenceClassification [10]. In our model, there are total of 13 layers with 12 layers of bert-attention, bert-output and bert-intermediate and one classification output layer.

For our loss optimizer, we "Adam" built by Adam Kingma et al, 15 combining the features of RMSProp algorithm and Adagrad algorithms, whcih computes stochastic gradient descent (SGD) with first order gradients. To Optimize different parameters, it uses first two moments.

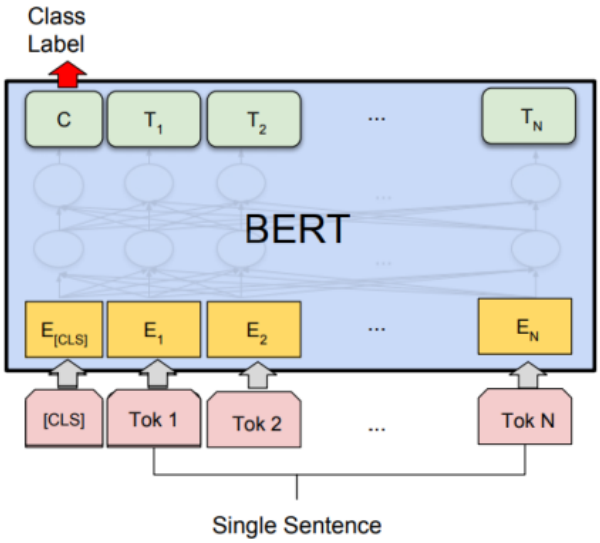

Fig. 10: Sentence Classification Task Model

V. Evaluation

The project was implemented in Jupyter Notebook through Google-Colab. The ATIS data is fed from the python's pickle file. After making the data ready, all the data was divided in the batches of 32 and converted into tensor iterators. We feed the data into the model and train the model for 5 epochs. Graph for training loss for each epoch is shown in fig: **??**

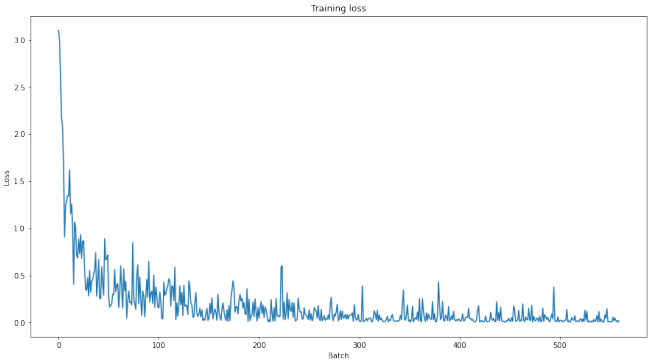

Fig. 11: Training loss on all the batches

We then run on validation set for a single epoch which is 10% of the training set and plot the accuracy in the fig 12 . We can see that It performs well on validation set.

After running the model on test set, we calculated the confusion matrix with the true labels and predicted labels. Plotted this confusion matrix through heatmap, that is showed in fig: 13

I have also made a classification table which gives compares metric scores, please refer 14

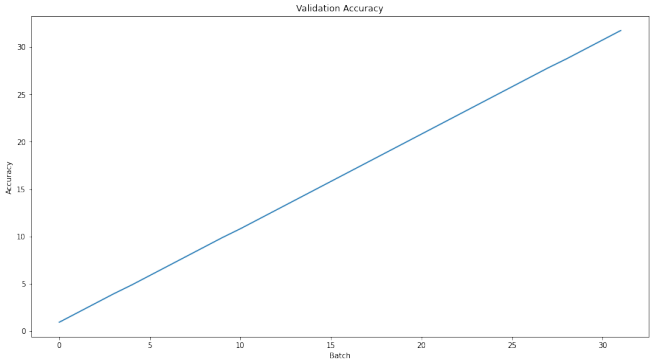

Fig. 12: Validation accuracy

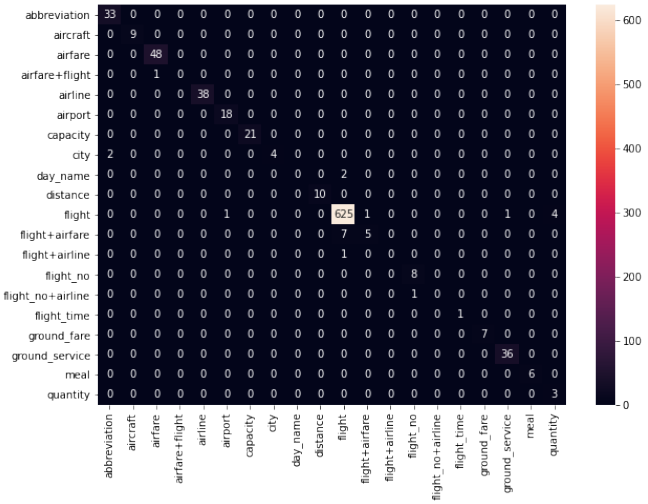

Fig. 13: Confusion matrix

	precision	recall	f1-score	support
0	1.00	0.94	0.97	35
1	0.89	1.00	0.94	8
3	1.00	0.96	0.98	50
4	0.00	0.00	0.00	0
6	1.00	1.00	1.00	38
8	1.00	0.95	0.97	19
9	1.00	1.00	1.00	21
11	0.67	1.00	0.80	4
12	0.00	0.00	0.00	0
13	1.00	1.00	1.00	10
14	0.99	0.99	0.99	635
15	0.50	0.86	0.63	7
16	0.00	0.00	0.00	0
17	1.00	0.89	0.94	9
18	0.00	0.00	0.00	0
19	1.00	1.00	1.00	1
20	0.86	1.00	0.92	6
21	1.00	1.00	1.00	36
23	1.00	1.00	1.00	6
24	1.00	0.38	0.55	8
accuracy			0.98	893
macro avg	0.75	0.75	0.73	893
weighted avg	0.99	0.98	0.98	893

Fig. 14: Classfication report

VI. Summary and Conclusions

This paper explored various techniques used to tackle the intent classification task and sequential improvements and differences over them. We accomplished the near SOTA result for intent classification task on ATIS dataset. We achieved this result by training over very small dataset consisting of less than 5000 samples. We studied different approach to same task and established BERT outperforms them. All thanks to almighty BERT, we used the pre-trained model and added a classification layer to achieve a result with % accuracy. In the result, we saw , our model was able to generate correct intent almost all the time. It saved a lot of time as training over such a small dataset took no time and abstracted away a lot of complex details of neural net. Though BERT has been able to break a lot of records on downstream tasks, there are other models like XLNet which outperformed BERT on over 20 tasks, which is built over autoregressive language model. So there is more work needed in the future.

References

[1] I. Sutskever, O. Vinyals, and Q. Le. (2014) Sequence to Sequence Learning with Neural Networks.In *Advances in Neural Information Processing Systems 27 (NIPS 2014)*

[2] Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733, (2016)*

[3] K. Cho, B.Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio. (2014) Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation.In *EMNLP, page 1724-1734. ACL, (2014)*

[4] Jacob Devlin, Ming-Wei Chang, Kanton Lee, Kristina Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *https://arxiv.org/pdf/1810.04805.pdf.*

[5] Charles T. Hemphill, John J. Godfrey, George R. Doddington, "The ATIS Spoken Language Systems Pilot Corpus", https://www.aclweb.org/anthology/H90-1021.pdf

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aiden N Gomez, Lukasz Kaiser, Illio Polosukhin "Attention Is All You Need", https://arxiv.org/pdf/1706.03762.pdf

[7] Qian Chen, Zhu Zhuo, Wen Wang, "BERT for Joint Intent Classification and Slot Filling", https://arxiv.org/pdf/1902.10909.pdf

[8] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi,Hananneh Hajishirzi1 (2017)," Bi-Directional attention flow for machine comprehension", ICLR 2017.

[9] Bing Liu, Ian Lane, "Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling"

[10] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, "Distributed Representations of Words and Phrases and their Compositionality"

[11] Jeffrey Pennington, Richard Socher, Christopher D. Manning, "GloVe: Global Vectors for Word Representation"

[12] Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, "Bag of Tricks for Efficient Text Classification"

[13] Sreelakshmi Ka, Rafeeque P Ca, Sreetha Sb, Gayathri E Sb, "Deep Bi-Directional LSTM Network for Query Intent Detection"

[14] Bing Liu1, Ian Lane, "Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling"

[15] Diederik P. Kingma, Jimmy Lei Ba (2015) Adam: A Method For Stochastic Optimization.*ICLR 2015*