Chatbot Using Trax

Submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Engineering

by

Leyton D'silva	**09**
Aaron Lobo	**33**
Valentine Miranda	**41**

Supervisor:

Prof. Uday Nayak

UNIVERSITY OF MUMBAI

Chatbot Using Trax

Submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Engineering

by

Leyton D'silva	**09**
Aaron Lobo	**33**
Valentine Miranda	**41**

Supervisor:

Prof. Uday Nayak

Department of Information Technology

Don Bosco Institute of Technology
Vidyavihar Station Road, Mumbai - 400070

2020-2021

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled **"Chatbot Using Trax"** is a bonafide work of:

Leyton D'silva	**09**
Aaron Lobo	**33**
Valentine Miranda	**41**

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Undergraduate** in **Bachelor of Information Technology.**

Date: 30/04/2021

(Prof. Uday Nayak)
Supervisor

(Prof. Janhavi B.)
HOD, IT Department

(Dr. Prasanna Nambiar)
Principal

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Project Report Approval for B.E.

This project report entitled **"Chatbot Using Trax"** by **Leyton D'silva (09), Aaron Lobo (33), Valentine Miranda (41)** is approved for the degree of **Bachelor of Engineering in Information Technology**

(Examiner's Name)

1. **Mrs Shweta Triphati**

(Supervisor's Name)

1. **Mr Nilesh**

Date: 30/04/2021

Place: Mumbai

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Leyton D'silva	**09**
Aaron Lobo	**33**
Valentine Miranda	**41**

Date: 30/04/2021

Abstract

In certain situations when human interaction is not possible, having something that can substitute for human interaction is not only complementary but also vital and therefore our project aims to develop a multi-domain chat bot that will answer queries in a more human-like manner while maintaining longer conversations. The technologies used in this project are trax, fast math, attention and a reformer model for the implementation of this system.

Keywords- Artificial Intelligence, Natural language processing, chatbot, Neural network, Deep Learning, Trax

Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

Chatbots can be classified using different parameters: the knowledge domain, the service provided, the goals, the input processing and response generation method, the human-aid, and the build method.

The expectations of the users are rising, and they want everything available at the moment, therefore chatbot technology is helping the users in almost all ways from satisfying them to answering their queries, full assistance is being offered to them. Chatbots are no longer seen as mere assistants, and their way of interacting brings them closer to users as friendly companions.

A chatbot is a program that communicates with you. It is a layer on top of, or a gateway to, a service. Sometimes it is powered by machine learning. Or, more commonly, it is driven using intelligent rules (i.e. if the person says this, respond with that). The services that a chatbot can deliver are diverse. From important life-saving health messages, to checking the weather forecast or to purchasing a new pair of shoes, and anything else in between. The term chatbot is synonymous with text conversation but is growing quickly through voice communication… Eg: “Alexa, what time is it?” The chatbot can talk to you through different channels; such as Facebook Messenger, Siri, WeChat, Telegram, SMS, Slack, Skype and many others. Consumers spend lots of time using messaging applications (more than they spend on social media). Therefore, messaging applications are currently the most popular way companies deliver chatbot experiences to consumers.

1.1 Problem Statement

This project aims to create a Chatbot using Trax that would assist us in multiple domains as well as be more human-like while maintaining longer conversations.

1.2 Scope of the Project

The current unprecedented pandemic has forced us to be distant and dependent on various technologies for communication. In times like these having something that can substitute for human interaction is not complementary but vital. We are focusing on creating a multi domain chatbot that can maintain conversations longer and is human-like.

1.3 Current Scenario

As of recent times, chatbot and qna systems are trending. User friendly customizable chatbots still are in the development phase and are mostly for the tech-savvy. Operating a chatbot is difficult for the average user. Having an accurate transformer based chatbot introduces latency which can make the experience less human-like. Having multi-lingual support is more of a luxury.

1.3.1 Rule Based Chatbots

Decision-tree bots are another name for rule-based chatbots. They follow a set of rules, as the name suggests. These rules serve as the foundation for the types of problems that the chatbot is familiar with and can solve.
Rule-based chatbots plot out conversations like a flowchart. They do this in anticipation of a customer's question and how the chatbot can react. Easy or complex rules may be used in rule-based chatbots. They can't, however, answer any questions that aren't in line with the defined rules. Interactions do not teach these chatbots anything. Furthermore, they only act and function in the situations for which you have prepared them.

1.3.2 Intent Classification based Chatbots

Chatbots can understand user requests thanks to Natural Language Processing (NLP). However, the conversation engine unit in NLP is critical for making the chatbot more contextual and providing users with customised conversation experiences. Intent classification is a key feature of this chatbot communication engine. It is a complex operation, despite how simple it can seem.
Text input is recognized by a software function known as a "classifier," which associates the data with a particular "purpose," resulting in a concise description of the terms for the machine to comprehend.

A classifier is a tool for categorizing data - in this case, a sentence - into multiple categories. Chatbots can classify each part of a sentence into broken down categories to understand the intention behind the information it has received, similar to how humans classify items into sets, such as a violin is an instrument, a shirt is a form of clothing, and happy is an emotion.

1.4 Need for the Proposed System

Chatbots have potential. Even though they are increasingly used, the modern chatbot is still a young technology. With the continuing development of AI, the potential for bots in business and personal lives is unlimited.

Chatbots emphasize the company's brand and image. The chatbot represents the company when it’s communicating with the customer, so, from a marketing point of view, it is a perfect embodiment of brand building.

They offer straightforward services. A well-optimized chatbot communicates only the essentials and does not overwhelm the user.

Chatbots automate processes. Bots are able to take on human work for, generally speaking, mundane or basic analytic task.

Chapter 2

2. Review of Literature

2.1 Summary of the investigation in the published papers

TOWARDS UNIVERSAL DIALOGUE STATE TRACKING, BY LILIANG REN, KAIGE XIE, LU CHEN, KAI YU

In this paper the dialogue system gathers all the information, and thus, keeps track of the user's goal at each dialogue turn. However, the current dialogue state trackers usually have a number of limitations, like not being able to handle the situation, where slot values are dynamically changing, or having a higher number of model parameters with every new slot being added. This limits the ability of these systems to scale to large dialogue domains. Thus, the authors suggest a universal dialogue state tracker StateNet, which not only overcomes these limitations but also significantly outperforms the state-of-the-art models but it lacks multi-language support and it cannot be customizable.

SEMANTIC PARSING FOR TASK ORIENTED DIALOG USING HIERARCHICAL REPRESENTATIONS, BY SONAL GUPTA, RUSHIN SHAH, MRINAL MOHIT, ANUJ KUMAR, MIKE LEWIS

In this paper they proposed a hierarchical annotation scheme for semantic parsing that allows the representation of compositional queries, and can be efficiently and accurately parsed by standard constituency parsing models. They released a dataset of 44k annotated queries, and show that parsing models outperform sequence-to-sequence approaches on this dataset but it lacks multi-lingual support and has higher latency.

2.2 Comparison between the tools / methods / algorithms

Name of the Model	BERT	Reformer Model
Library used	TensorFlow	Trax
Token limit	512	Up to 500,000
Latency	Less compared to Reformer model	More compared to BERT model
Customizability	Easier to customize the dataset	Difficult to customize the dataset
Training time	10 – 12 minutes	48 – 72 hours depending on the dataset

Table 2.1 Comparison between Algorithms

Chapter 3

3. Analysis and Design

3.1 Methodology / Procedure adopted

Throughout the project's software development life cycle, the development process used involves constant iterations of development and testing. Both development and testing take place at the same time. This demonstrates how our approach is similar to agile methodology. Figure 3.1 depicts it. Different modules are discussed at weekly meetings to ensure that all of the modules are complete and accurate. The project leader assigns a task for the following week at each meeting, and the progress on that task is reviewed the following week. On a weekly basis, meetings with the project guide are held.

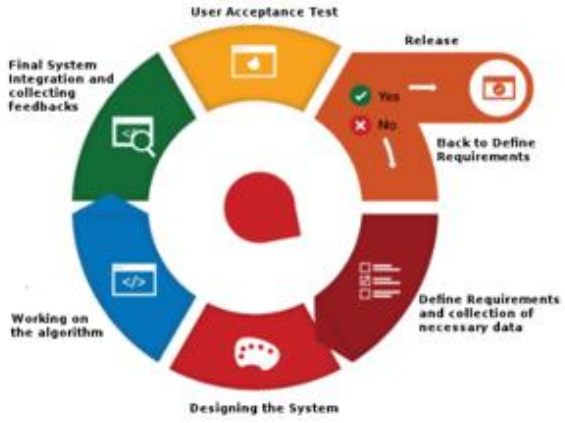


Figure 3.1 Methodology Adopted

3.2 Analysis

We initially created a classifier-based model for lower latency and it being easily customizable, since it is a very shallow model it would have had inconsistent behavior, therefore had to replace it. We then used a bert model which is based on the transformer architecture but it had a major limitation i.e., it couldn't be trained on larger datasets let alone multi-domain ones. Finally, we chose a reformer model using trax which is a library built for streamlined and efficient code. This generative model although having higher latencies could be trained on larger multi-domain datasets.

3.3 System Architecture / Design

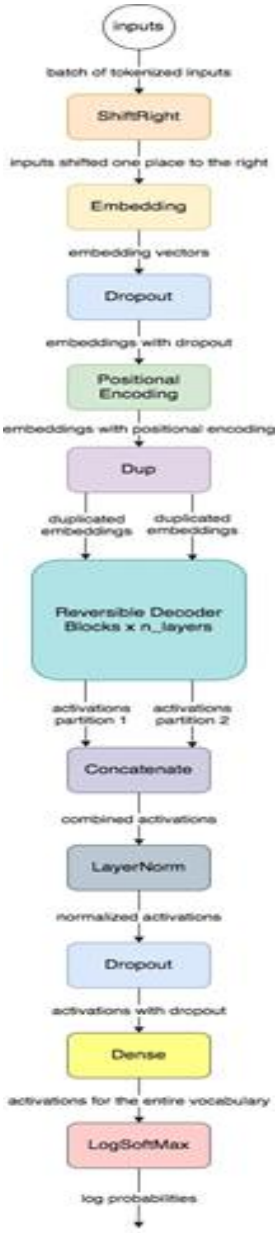


Figure 3.3 System Architecture

3.4 Modules and their description

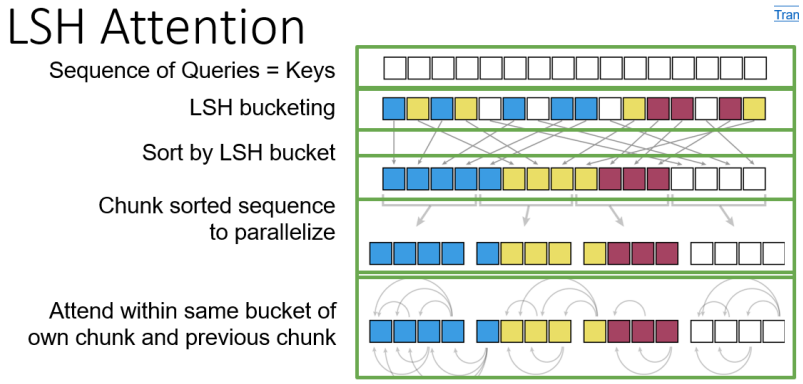


Figure 3.4.1 LSH Attention Module

When making the next decision, the larger the matrix multiply in the previous section is, the more context can be taken into account. The self-attention dot product, on the other hand, grows in proportion to the size of the input squared. For eg, if one desired a 1024 input size, each head will produce 1024^2 or more than a million dot products. Locality Sensitive Hashing (LSH) Self Attention is one such process.

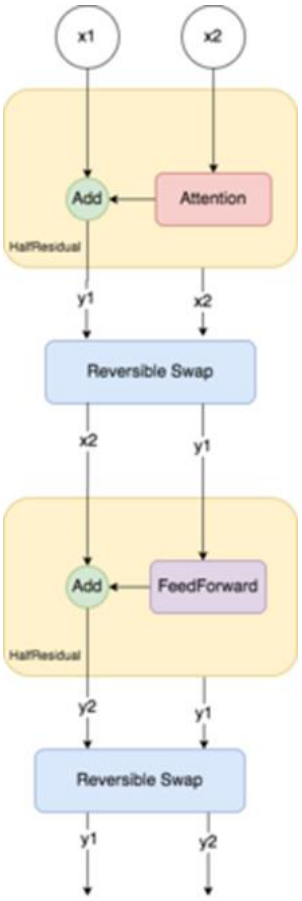


Figure 3.4.2 RevNet Module

A Reversible Residual Network, or RevNet, is a variant of a ResNet where each layer's activations can be reconstructed exactly from the next layers. Therefore, the activations for most layers need not be stored in memory during backpropagation. The result is a network architecture whose activation storage requirements are independent of depth, and typically at least an order of magnitude smaller compared with equally sized ResNets.

Chapter 4

4. Implementation

4.1 Implementation Plan *for Sem – 8*

Tasks	Weeks	1	2	3	4	5	6	7	8	9	10	11
Research on trax.												
Setup wsl to work with trax												
Setup google collab and try to setup trax												
Implementation of trax												
fixing version compatabilities												
Using various test cases to check for any possible errors												

Figure 4.1 Implementation plan for sem 8

4.2 Coding Standard

```
!pip install --upgrade tensorflow-probability==0.8
!pip install trax
!pip install t5==0.8.1
!pip list | grep trax
```

Figure 4.2.1 Installing the necessary dependencies

```
data_pipeline = trax.data.Serial(
    trax.data.Shuffle(),

    trax.data.Tokenize(vocab_dir=VOCAB_DIR,
                       vocab_file=VOCAB_FILE),

    trax.data.FilterByLength(2048),

    trax.data.BucketByLength(boundaries=[128, 256,  512, 1024],
                             batch_sizes=[16,    8,    4,   2, 1]),

    trax.data.AddLossWeights(id_to_mask=0)
)

train_stream = data_pipeline(stream(train_data))
eval_stream = data_pipeline(stream(eval_data))
```

Figure 4.2.2 Tokenizing, Batching with Bucketing

```
def training_loop(ReformerLM, train_gen, eval_gen, output_dir = "./model/"):

    lr_schedule = trax.lr.warmup_and_rsqrt_decay(
        n_warmup_steps=1000, max_value=0.01)


    train_task = training.TrainTask(

        labeled_data=train_gen,

        loss_layer=tl.CrossEntropyLoss(),

        optimizer=trax.optimizers.Adam(0.01),

        lr_schedule=lr_schedule,

        n_steps_per_checkpoint=10
    )


    eval_task = training.EvalTask(

        labeled_data=eval_gen,

        metrics=[tl.CrossEntropyLoss(), tl.Accuracy()]
    )

    loop = training.Loop(ReformerLM(mode='train'),
                         train_task,
                         eval_tasks=[eval_task],
                         output_dir=output_dir)

    return loop
```

Figure 4.2.3 Training the dataset

```
[ ]  def reversible_layer_forward(x, f, g):
         x1, x2 = np.split(x, 2, axis=-1)

         y1 = x1 + f(x2)
         y2 = x2 + g(y1)

         y = np.concatenate([y1, y2], axis=-1)

         return y
```

```
[ ]  def reversible_layer_reverse(y, f, g):

         y1, y2 = np.split(y, 2, axis=-1)

         x2 = y2 - g(y1)
         x1 = y1 - f(x2)

         x = np.concatenate([x1, x2], axis=-1)

         return x
```

Figure 4.2.4 Using Reversible Layers

```
[ ]  shape11 = trax.shapes.ShapeDtype((1, 1), dtype=np.int32)

     def attention(*args, **kwargs):
         kwargs['predict_mem_len'] = 120
         kwargs['predict_drop_len'] = 120
         return tl.SelfAttention(*args, **kwargs)

     model = ReformerLM(
         vocab_size=33000,
         n_layers=6,
         mode='predict',
         attention_type=attention,
     )
```

Figure 4.2.5 Decoding from our batch trained Dataset

Chapter 5

5. Results and Discussion

```
sample_sentence = ' Person 1: Can you book a taxi? Person 2: '
```

```
Person 1: Can you book a taxi?
Person 2: : I sure can. When would you like to leave?
Person 1: I want to leave after 13:00.
Person 2: I've booked you a grey ford, contact number is 07750772.
Person 1: Thank you for your help.
Person 2: You're welcome. Have a great day!!my pleasure to help. Goodbye.
Person 1: Thank you for your help.
Person 1: You're welcome. Have a good night! Bye!
```

Figure 5.1 Results

In this first result, we have given input as “Can you book a taxi?” and the chatbot replies with another question like what time it would like to leave based on the dataset trained. On receiving gratitude inputs in replies back.

```
sample_sentence = ' Person 1: '
sample_sentence += input()
sample_sentence += ' Person 2: '
```

```
Person 1: is there a restaurant nearby
Person 2: : There are 110 restaurants in Cambridge. What type of food would you like?
Person 1: I would like to eat Chinese food.
Person 2: I have several options for you to choose from. What price range would you prefer?
Person 1: I would like to eat Chinese food.
Person 2: There are 4 Chinese restaurants in the centre of town. Do you have a preference?
Person 1: I would like to book a table for 1 person at 15:00 on Wednesday. Person:  i i i want
```

In this second result, we have given custom input that is “is there a restaurant nearby” and it generates a conversation which is similar to another conversation in the dataset. On receiving the inputs this chatbot can maintain a conversation for longer time.

Chapter 6

6. Conclusion & Future Work

Trax is a deep-learning library focused on clear code and speed and is actively used and maintained by the Google Brain team. It includes basic models (like ResNet, LSTM, Transformer, and R.L. algorithms (like REINFORCE, A2C, P.P.O.). It is actively used for research and includes new models like the Reformer, and new R.L. algorithms like A.W.R. Trax has bindings to many deep learning datasets, including Tensor2Tensor TensorFlow datasets. On training the chatbot using the multiwoz Cambridge dataset, we obtained steady results.

With good hardware upgrades this project can be set up and used on a particular system. With a specific dataset this project can be used in single or multiple domains.

References

[1] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. CoRR, abs/1808.04444, 2018. URL http: //arxiv.org/abs/1808.04444.

[2] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya P. Razenshteyn, and Ludwig Schmidt. Practical and optimal LSH for angular distance. CoRR, abs/1509.02897, 2015. URL http://arxiv. org/abs/1509.02897.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016. URL http://arxiv.org/abs/1607.06450

[4] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. CoRR, abs/1506.02075, 2015. URL http://arxiv.org/ abs/1506.02075.

[5] Sarath Chandar, Sungjin Ahn, Hugo Larochelle, Pascal Vincent, Gerald Tesauro, and Yoshua Bengio. Hierarchical memory networks. arXiv preprint arXiv:1605.07427, 2016.

[6] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. URL https://openai.com/blog/sparse-transformers, 2019.

[7] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. CoRR, abs/1511.02301, 2015. URL http://arxiv.org/abs/1511.02301

[8] Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herve´ Jegou. Large memory layers with product keys. ´ CoRR, abs/1907.05242, 2019. URL http: //arxiv.org/abs/1907.05242.

[9] Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating wikipedia by summarizing long sequences. CoRR, abs/1801.10198, 2018. URL http://arxiv.org/abs/1801.10198.

[10] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In Proceedings of the Third Conference on Machine Translation: Research Papers, pp. 1–9, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6301. URL https://www.aclweb.org/anthology/W18-6301.

[11] Matt Post. A call for clarity in reporting BLEU scores. In Proceedings of the Third Conference on Machine Translation: Research Papers, pp. 186–191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/W18-6319.

Acknowledgements

Apart from the efforts and significant contributions of each and every team member, our guide Prof. Uday Nayak played a significant role in this project. We owe him a huge debt of gratitude for his unwavering encouragement and support during the thesis and its directive conduct. We were greatly motivated by his wise wisdom during our study. We'd like to express our gratitude to Don Bosco Institute of Technology, IT department for providing us with numerous opportunities to develop and explore our skills. We'd also like to express our gratitude to Prof. Sunantha Krishnan, our project coordinator, for her unwavering support and motivation.

Leyton D'silva	**09**
Aaron Lobo	**33**
Valentine Miranda	**41**

Date: 30/04/2021

Chatbot Using Trax

Uday Nayak, Leyton D'silva, Valentine Miranda and, Aaron Lobo

Department of Information Technology, Don Bosco Institute of Technology, Mumbai, India

Abstract- In certain situations when human interaction is not possible, having something that can substitute for human interaction is also complementary and vital. Therefore, our project aims to develop a multi-domain voice-enabled chatbot to answer queries in a more human-like manner while maintaining extended conversations. The technologies used in this project are trax, fast math, attention, and a reformer model to implement this system.

Index Terms- Artificial Intelligence, Natural language processing, chatbot, Neural network, Deep Learning, Trax.

I. INTRODUCTION

Chatbots can be classified using different parameters: the knowledge domain, the service provided, the goals, the input processing and response generation method, the human-aid, and the build method.

The users' expectations are rising, and they want everything available at the moment. Therefore, chatbot technology helps the users in almost all ways, from satisfying them to answering their queries and offering full assistance. Chatbots are no longer mere assistants, and their way of interacting brings them closer to users as friendly companions.

A chatbot is a program capable of communicating with users. It acts as a gateway to a service. Sometimes it is powered by machine learning. More commonly, driven using intelligent rules (i.e., if the person says this, respond with that). The services that a chatbot can deliver are diverse. From important life-saving health messages, checking the weather forecast or purchasing a new product, and anything else. The term chatbot is synonymous in text conversation but is proliferating through voice communication. E.g., "Alexa, what time is it?" The chatbot can talk to the user with the help of various channels, like Facebook Messenger, Siri, Slack, Skype, and many more. Consumers use messaging applications (more than social media applications). Therefore, there are popular ways for companies to deliver chatbot experiences to consumers.

II. AIM

The aim is to design a multi-domain voice-enabled chatbot to answer queries in a more human-like manner while maintaining extended conversations.

III. RELATED WORKS

A. TOWARDS UNIVERSAL DIALOGUE STATE TRACKING, BY LILIANG REN, KAIGE XIE, LU CHEN, KAI YU

In this paper, the dialogue system gathers all the information and keeps track of the user's goal at each dialogue turn. However, the current dialogue state trackers usually have several limitations, like not handling the situation where slot values are dynamically changing or having a higher number of model parameters with every new slot added.

It limits the ability of these systems to scale to large dialogue domains. Thus, the authors suggest a universal dialogue state tracker StateNet, which overcomes these limitations and significantly outperforms the state-of-the-art models but lacks multi-language support and cannot be customizable.

B. SEMANTIC PARSING FOR TASK ORIENTED DIALOG USING HIERARCHICAL REPRESENTATIONS, BY SONAL GUPTA, RUSHIN SHAH, MRINAL MOHIT, ANUJ KUMAR, MIKE LEWIS

This paper proposed a hierarchical annotation scheme for semantic parsing that allows the representation of compositional queries efficiently and accurately parsed by standard constituency parsing models. They released a dataset of 44k annotated queries and show that parsing models outperform sequence-to-sequence

approaches on this dataset, but it lacks multi-lingual support and has higher latency.

IV. NEED FOR THE PROPOSED SYSTEM

Chatbots have potential. Even though increasingly used, the modern chatbot is still a young technology. With the continuing development of A.I., the potential for bots in business and personal lives is unlimited.

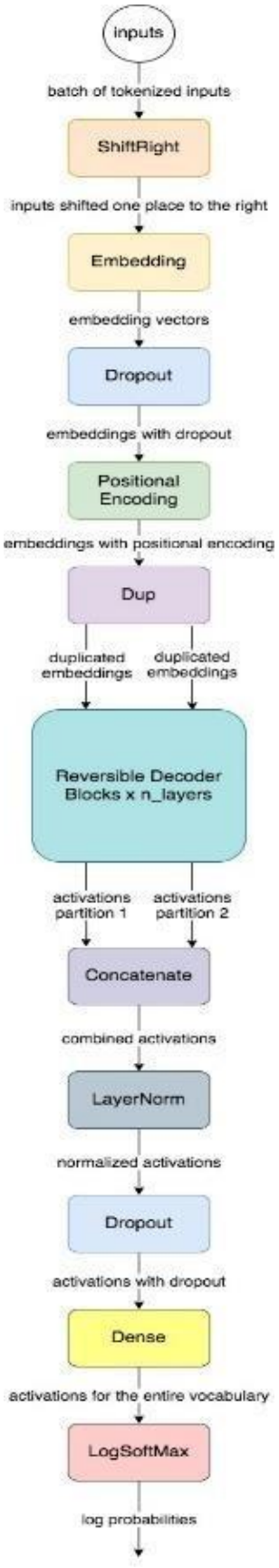
Chatbots emphasize the company's brand and image. The chatbot represents the company when it communicates with the customer, so it is a perfect embodiment of brand building from a marketing point of view.

They offer specific services. A well-optimized chatbot communicates only the essentials and does not overwhelm the user.

Chatbots automate processes. Bots can take on human work for, generally speaking, mundane or basic analytic tasks.

V. SYSTEM ARCHITECTURE

The Reformer architecture is a Transformer model designed to process longer data sequences efficiently (up to 1 million words in a language processing). The execution of Reformer requires lower memory consumption for achieving impressive performance even when running on a single GPU. It addresses three primary sources of memory consumption in the Transformer while improving how the Reformer model can handle up to one million words of context windows efficiently, all on a single accelerator and using only 16 Gigabytes of memory. In a nutshell, the model combines two techniques to solve attention and memory allocation problems: locality-sensitive-hashing (LSH) to reduce the complexity of attending over long sequences and reversible residual layers to more efficiently use the memory available.



5.1 Dot-product Attention

Scaled dot-product attention is the standard attention used in the Transformer, where the input consists of keys and queries of dimension dk, and values of dimension dv. The dot products are computed, scaled by √ dk, and a softmax function is applied to get the values' weights. The attention function is computed simultaneously on queries, packed together into a matrix Q in practice. Assuming the keys and values are packed together into matrices K and V, the matrix of outputs is defined as:

$$Attention(Q,V,K) = (\frac{QK^T}{\sqrt{dk}})V$$

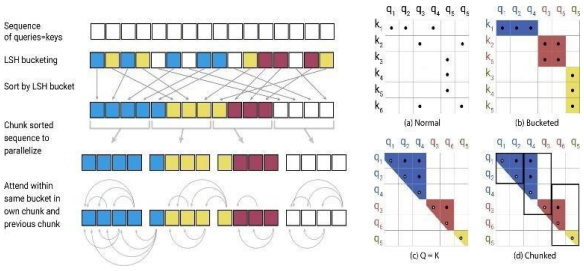
The self-attention dot product grows as the size of the input is squared. For example, if one wished to have an input size of 1024, that would result in 1024^2or over a million dot products for each head! Therefore, it has resulted in significant research related to reducing the compute requirements. One such approach is Locality Sensitive Hashing (L.S.H.) Self Attention.

5.2 Locality-Sensitive Hashing Attention

L.S.H. Self-attention uses Queries only, no Keys. Attention then generates a metric of the similarity of each value of Q relative to all the other values in Q. Further. Multiple random hashes can improve the chances of finding similar entries. It is the approach taken here, though the hash is implemented a bit differently. The values of Q are hashed into buckets using a randomly generated set of hash vectors. Multiple sets of hash vectors are used, generating multiple hash tables. In the figure above, we have three hash tables with four buckets in each table. Notionally, following the hash, Q's values have been replicated three times and distributed to their appropriate bucket in each of the three tables. To find similarity, then, one generates dot-products only between members of the buckets. The result of this operation provides information on which entries are similar. As the operation is distributed over multiple hash tables, the results need to be combined to form a complete picture used to generate a reduced dot-product attention array. It is clear that because we do not compare every value vs. every other value, Dots' size is reduced.

The challenge in this approach is getting it to operate efficiently. It will operate poorly on a vector processing machine such as a GPU or T.P.U. Ideally, operations are done in large blocks with uniform sizes.

While it is straightforward to implement the hash algorithm this way, it is challenging to manage buckets and variable-sized dot- products.



For a single query position i at a time:

$$o_i = \sum_{j \in P_i} \exp\left(q_i \cdot k_j - z(i, P_i)\right) v_j$$

where $P_i = \{j : i \geq j\}$

We introduce the notation Pi to represent the set that the query at position i attends to, and z to denote the partition function (i.e., the normalizing term in the softmax). For clarity, we also omit to scale $\sqrt{dk}$.

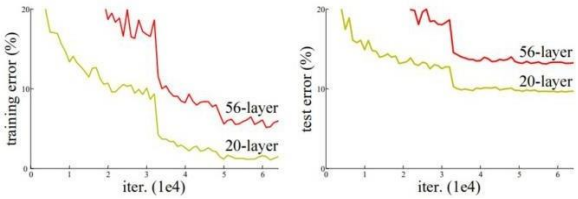
For batching purposes, we typically perform attention over a larger set $\tilde{P}i = \{0, 1, \ldots, l\} \supseteq$ Pi while masking out elements, not in Pi:

$$o_i = \sum_{j \in \tilde{P}_i} \exp\left(q_i \cdot k_j - m(j, P_i) - z(i, P_i)\right) v_j$$

$$\text{where } m(j, P_i) = \begin{cases} \infty \; if \; j \notin Pi \\ 0 \; otherwise \end{cases}$$

5.3 ResNet

For solving a complex problem, we stack some additional layers in Deep Neural Networks. It results in improved accuracy and performance. The intuition behind adding them is that they progressively learn more complex features. For example, for recognizing images, the first layer learns to detect edges. The second learn to identify textures. Finally, the third layer will learn to detect objects. Nevertheless, there exists a maximum threshold for depth with the traditional Convolutional neural network model. Let us look at a plot that describes error% on training and testing data for a 20-layer Network and 56 layers Network.



We can see that the error percentage for 56-layer is more than a 20-layer network in both cases of training data and testing data. It suggests that with adding more layers on top of a network, its performance degrades. It could be blamed on the optimization function, initialization of the Network, and vanishing gradient problem. One might be thinking that it could be a result of overfitting too. However, the error percentage for a 56-layer network is worst on both training and testing data which is not the case for an overfitting model.

The problem of training deep networks has been alleviated with ResNet or residual networks' introduction, and these ResNets are made up of Residual Blocks.

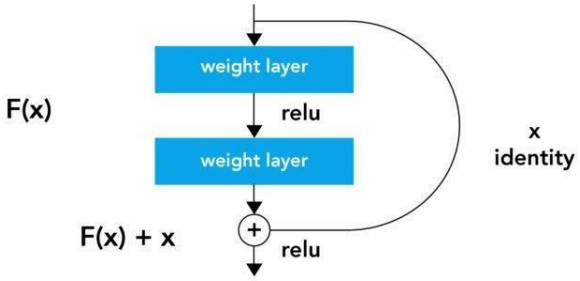


Figure 2. Residual learning: a building block.

The first thing to notice is a direct connection that skips some layers (may vary in different models) in between. Such a connection is called 'skip connection' and is the core of residual blocks, due to which the output of the layer is different. Without using the skip connection, the input 'x' is multiplied by the layer's weights, followed by a bias term.

Next, it goes through the activation function, f(), and we get our output as H(x).

$$H(x) = f(wx + b)$$

Or

$$H(x) = f(x)$$

Now, after introducing skip connection, the output is changed to

$$H(x) = f(x) + x$$

A slight problem with this approach is when the input dimensions vary from that of the output, resulting in convolutional and pooling layers. However, when dimensions of f(x) are different from x, two approaches can be considered:

A skip connection is padded with extra zero entries, resulting in the dimensions increase.

The projection method used to match the dimension is done by adding a 1×1 convolutional layer to input.

The output is:

$$H(x) = f(x) + w1.\ x$$

Here, an additional parameter w1 is added with no additional parameter for the first approach.

Here we add parameter w1, whereas no additional parameter is required when using the first approach.

Skip connections in ResNet help solve vanishing gradient in deep neural networks, allowing the gradient's alternate shortcut path to flow through. These connections help by allowing the model to learn the identity functions, ensuring that the higher layer will perform similarly to the lower layer.

Let us explain this further.

Say we have an external network and a deep network that maps an input 'x' to output 'y' using H(x). The deep Network must perform as well as the shallow Network without degrading the performance observed in plain neural networks (without residual blocks). It can be achieved if the additional layers in a deep network learn the identity function. Thus, their output equals inputs which eliminates them to degrade the performance even with extra layers.

Moreover, the residual blocks make it exceptionally easy for layers to learn identity functions. From the formulas above, it is evident that in plain networks, the output is

$$H(x) = f(x) + x$$

So, in order to learn an identity function, f(x) must be equal to x, which is grader to attain, whereas in-case of ResNet, which has output:

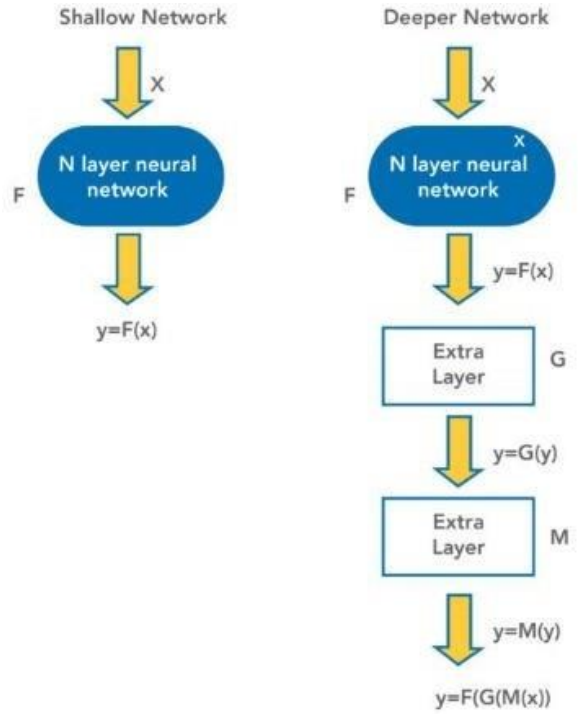
$$H(x) = f(x) + x,$$

$$f(x) = 0$$

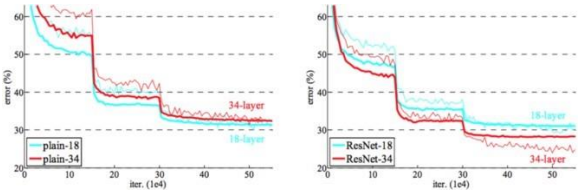
$$H(x) = x$$

All that is needed is to make f(x)=0, which is more manageable, which leads to x as output which is also the input.

In a best-case scenario, additional layers of the deep neural Network can better approximate the mapping of 'x' to output 'y' than the shallower counterpart and reduce the error by a significant margin. Furthermore, ResNet is expected to perform equally or better than plain deep neural networks. Using ResNet has resulted in a significant enhancement in the performance of neural networks with more layers.



G and M act as Identity Functions. Both the Network Give same output



The difference is vast in the networks with 34 layers where ResNet-34 has a much lower error percentage when compared to plain-34. Also, on further observation, the error percentage for plain-18 and ResNet-18 is almost the same.

5.4 RevNet

Reversible Residual Network (RevNet) is a ResNet variant where each layer's activations can be reconstructed exactly from the subsequent layers. Therefore, activations for most layers need not be stored in memory during backpropagation. It results in a network architecture whose activation storage requirements are independent of depth and typically an order of magnitude smaller than equally sized ResNets

RevNet consists of a series of reversible blocks. Units in each layer are divided into two groups, denoted $x1$ and $x2$; the authors find what works best is partitioning the channels. Each reversible block takes inputs ($x1$, $x2$) and produces outputs ($y1$, $y2$) according to the following additive coupling rules – inspired the transformation in NICE (nonlinear independent components estimation) – and residual functions F and G analogous to those in standard ResNets:

$$y1 = x1 + F(x2)$$

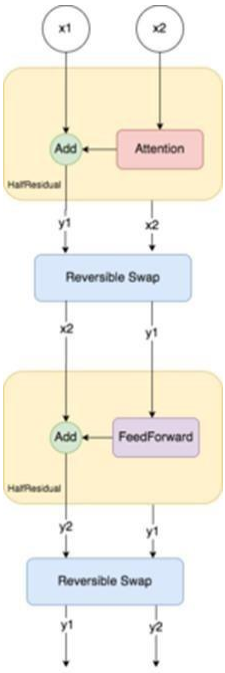
$$y2 = x2 + G(x1)$$

It is possible to reconstruct each layer's activations from the next layer's activations as follows:

$$x2 = y2 - G(y1)$$

$$x1 = y1 - F(y2)$$

Unlike residual blocks, reversible blocks must have a stride of 1; otherwise, the layer discards information and cannot be reversible. Standard ResNet architectures have a handful of layers with more significant stride. When defining a RevNet architecture analogously, the activations must be stored explicitly for all non-reversible layers.



The above image refers to the reversible decoder blocks. These reversible decoder blocks are used in the Reformer to improve memory efficiency.

VI. CONCLUSION

Trax is a deep-learning library focused on clear code and speed and is actively used and maintained by the Google Brain team. It includes basic models (like ResNet, LSTM, Transformer, and R.L. algorithms (like REINFORCE, A2C, P.P.O.). It is actively used for research and includes new models like the Reformer, and new R.L. algorithms like A.W.R. Trax have bindings to many deep learning datasets, including Tensor2Tensor TensorFlow datasets. On training the chatbot using the multiwoz Cambridge dataset, we obtained steady results.

VII. ACKNOWLEDGEMENT

We want to thank our Principal, Dr. Prasanna Nambiar, and the H.O.D Prof. Janhavi Baikerikar of Don Bosco Institute of Technology for letting us carry out this project.

VIII. REFERENCES

[1]. Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. CoRR, abs/1808.04444, 2018. URL http://arxiv.org/abs/1808.04444

[2]. Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya P. Razenshteyn, and Ludwig Schmidt. Practical and optimal L.S.H. for angular distance. CoRR, abs/1509.02897, 2015. URL http://arxiv.org/abs/1509.02897

[3]. Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. arXiv preprint arXiv:1607.06450, 2016. URL http://arxiv.org/abs/1607.06450

[4]. Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. CoRR, abs/1506.02075, 2015. URL http://arxiv.org/abs/1506.02075.

[5]. Sarath Chandar, Sungjin Ahn, Hugo Larochelle, Pascal Vincent, Gerald Tesauro, and Yoshua Bengio. Hierarchical memory networks. arXiv preprint arXiv:1605.07427, 2016.

[6]. Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. URL https://openai.com/blog/sparse-transformers, 2019.

[7]. Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. CoRR, abs/1511.02301, 2015. URL http://arxiv.org/abs/1511.02301 for Computational Linguistics. URL https://www.aclweb.org/anthology/ W18-6319.

[8]. Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Herve´ Jegou. Large memory layers with product keys. ´ CoRR, abs/1907.05242, 2019. URL http: //arxiv.org/abs/1907.05242.

[9]. Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. Generating Wikipedia by summarizing long sequences. CoRR, abs/1801.10198, 2018. URL http://arxiv.org/abs/1801.10198.

[10]. Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. Scaling neural machine translation. In Proceedings of the Third Conference on Machine Translation: Research Papers, pp. 1–9, Brussels, Belgium, October 2018. Association for Computational Linguistics. DOI: 10.18653/v1/W18-6301. URL https://www.aclweb.org/anthology/W18-6301.

[11]. Matt Post. A call for clarity in reporting BLEU scores. In Proceedings of the Third Conference on Machine Translation: Research Papers, pp. 186– 191, Belgium, Brussels, October 2018. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/ W18-6319.

[12]. Matt Post. A call for clarity in reporting BLEU scores. In Proceedings of the Third Conference on Machine Translation: Research Papers, pp. 186–191, Belgium, Brussels, October 2018. Association

PLAGIARISM REPORT