Exploring Encoder-Decoder and Transformer Models for Conversational Agents

Ednah Olubunmi Aliyu[1] and Eduan Kotzé[2]
University of the Free State [1,2]
Department of Computer Science and Informatics
Bloemfontein, South Africa

Abstract - This paper investigates neural encoder-decoder and transformer models for open-domain dialogue agents to understand how they work with abstraction layers in deep neural language models. The encoder-decoder encodes and decodes the input/output vectors, while the learned embedding layer reduces the dimensionality of the vocabulary. The transformer model computes internal multi-head attention between two vectors using cosine similarity and similarly masked multi-head attention to computes attention vectors for the current and previously generated word embeddings from the source sentence. The models trained using ChatterBot Dialogue Corpus, the Cornell Movie Corpus and the Ubuntu Dialogue Corpus. It differs from previous work in that we compared three datasets to three models (Long Short-Term Memory (LSTM), Gated Recurrent Units (GRU) and Multi-Head Attention Transformer) and performance were determined using three human raters and A/B testing process. Human ratings are based on four key traits of conversational agents in the open-domain space, namely repetition, interestingness, making sense and fluency. Using the Fleiss-kappa inter-rater, the level of inter-rater agreement was measured as 100% for the ChatterBot Dialogue Corpus while the other two never agreed. A randomised testing process called A/B showed that GRUs scored 84%, LSTM scored 76% and Multi-Head Attention Transformer scored 78%. This result demonstrates that A/B testing can improve the quality and user experience of open-domain conversational agents.

Index Terms - Autoregression, Chatbots, Encoder-Decoder (ED), Multi-head Attention (MA), Transformer and Subjective Human Evaluation (SHE)

I. INTRODUCTION

Being able to interact with a computer in natural language (that is, sequences of words) and creating machines that think have long been desired [1]. This technology is called conversational agents or chatbots and is categorised into three types, namely a task-oriented dialogue agent, a non-task-oriented chatbot and a question-answering dialogue system. Firstly, a task- oriented dialogue agent is designed to accomplish certain tasks such as making a reservation at a restaurant [2]. Secondly, a non-task-oriented chatbot is an automated system that communicates with humans through text messages. Thirdly, a question-answering (QA) dialogue system is developed to answer the user's questions. These models are developed through rule-based or corpus-based methods. The rule-based method uses if-else conditional logic (rules) to generate responses while the majority of corpus-based chatbots produce their responses either by retrieval methods (taking a response from a corpus), or generation methods (using a language model or encoder- decoder to generate the response) given the dialogue context [2]. This generative method application is the focus of this research work.

In [3] dealing with language processing is sequential since language is a temporal phenomenon in nature. In the context of neural language models, this temporal nature that employs fixed-size input vectors with associated weights to capture all relevant aspects makes it difficult to deal with sequences of varying length and fails to capture important temporal aspects of language. [4] stated that one solution to this problem is to involve deep learning in training the neural network. The most widely used deep neural network for natural language processing is recurrent neural networks (RNN). [3], defined a RNN as any network that contains a cycle within its network connections. That is, any network where the value of a unit is directly, or indirectly, dependent on earlier outputs as an input. Also, it is trained via backpropagation through time (BPTT) and shares trainable weights over the entire sequence. RNNs are rarely used because of their gradient vanishing problem [5]; [6]. Long short-term memory (LSTM), gated recurrent unit (GRU), bidirectional RNN, encoder-decoder (sequence-to-sequence) architecture and convolutional recurrent neural networks were introduced to solve this problem. Hence, the state-of-the-art approach in sequence modeling is encoder-decoder models [7].

In literature, the encoder-decoder model has been used in different contexts such as machine translation, text summarisation, image captioning, next word generation, and conversational agents. [5] proposed a sequence-to-sequence framework that extracts knowledge on an IT helpdesk dataset to solve a technical problem via conversations and an open-domain movie transcript dataset to perform common sense reasoning. The work of [8] demonstrated an RNN language model generating responses that are sensitive to the context of the conversation trained on large quantities of unstructured Twitter media.

One major challenge of the encoder-decoder baseline model, according to [9], is its inefficiency to model long-range dependencies with ease. [10] addressed this issue by introducing the self-attention mechanism. The self-attention mechanism has the ability to connect any two positions of the input sequence to compute a representation of that sequence. According to [11], Transformer is a de facto model for many state-of-the-art sequence processing tasks. For instance, adding attribute descriptions to the input text was carried out by [12] in TransferTransfo model using Generative Pre-trained Transformer (GPT) architecture for generating personal responses through a persona-chat dataset.

This paper examines the application of encoder- decoder models (LSTM and GRU) [13];[14] and Transformer models [10] using the Chatterbot Dialogue Corpus [15], the Cornell Movie Corpus [16] and the Ubuntu Dialogue Corpus [17]. Our goal is to have a better understanding in this area of research by experimenting with a word-based tokenisation algorithm for the encoder-decoder models (LSTM and GRU), a subword- based algorithm for the transformer model, and building an input pipeline with the distributed training strategy provided by Tensorflow's Functional API [18]. We also perform human evaluation for the conversational agents to measure the responses produced by the different models.

In summary, the contributions of this paper are: First, we examine the application of encoder-decoder and transformer models for conversational agents. To the best of our knowledge, this is the first work that jointly show the differences in performances between the models response diversity using the three datasets. Also, we evaluate the model using human evaluation because from the literature, we found that human evaluation on chatbots needs more exploration (Section 4). Second, experimental results show that GRU model generated meaningful short text responses compared to the LSTM while transformer manage long term dependencies better than encoder-decoder models (Section 4).

The remainder of this paper is organised as follows. Section 2 provides background and related works. Section 3 describes the experimental setup and proposed conversational agents components. Section 4 presents the results and discussion while Section 5 concludes, providing directions for future research.

II. BACKGROUND AND RELATED WORK

A conversational agent is an example of conditional language tasks. That is, the task of predicting the next word, given the words so far, and some other input x as defined in "(1)," [19]; [20].

$$P(y|x) = \prod_t P(y_t|x, y_1, \dots, y_{t-1}) \quad (1)$$

where x = dialogue history (context), y = model's next utterance, each word is given a timestamp t,

$t-1$ which describes the position of an individual word.

In [21], they identified six different neural generation systems that produce this probability distribution for non-task-oriented dialogue tasks, namely Encoder-Decoder-based methods, Hierarchical Recurrent Encoder-Decoder (HRED)-based methods, Variational Auto-Encoder (VAE)-based methods, Reinforcement Learning (RL)-based methods, Generative Adversarial Network (GAN)-based methods and Pre-training-model- based methods. However, this research focuses on encoder-decoder neural generation in an open-domain dialogue system. Fig. 1 is an example of an encoder- decoder framework where the encoder (LSTM / GRU) encodes the source sequence X, each unit generates the T tokens and, together with the input and contexts, sentences are concatenated into a semantic context vector and serve as input to the decoder (LSTM / GRU). [21] explained further that, given the context and a response sequence Y of length T, the decoder maximizes the generation probability of Y conditioned on context: $P(Y|context)$.

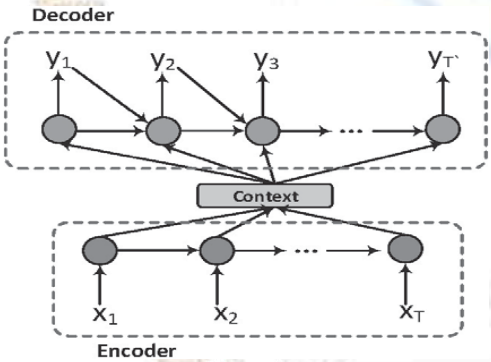

Fig.1 A typical Encoder-Decoder framework. [21]

Next, we will provide state-of-the-art conversational agents models and techniques that have been applied in designing such an interactive system.

(1) Background: Recurrent Neural Network

From literature, we define a recurrent neural network (RNN) as an extension of a conventional feedforward neural network, which is able to handle a variable-length sequence input [14]. In this paper, we are interested in evaluating two recurrent units in RNNs: long short-term memory (LSTM) unit and a gated recurrent unit (GRU).

(2) Long Short-Term Memory Units

According to [17], the Long Short-Term Memory (LSTM) unit was proposed to model long-range dependencies. This was made possible using several gates. The *forget gate* allows information to pass through the *sigmoid* activation function (return value from 0 to 1) and forgets any value near zero. The *input gate* passes information through the *sigmoid* and *tanh* (map value between -1 and 1) activation function. The results from these two functions are multiplied and passed to the *cell state*. The *output gate* decides what the next hidden state should be while the *cell state* stores information.

(3) Gated Recurrent Units

A gated recurrent unit (GRU) is similar to LSTM unit and was first proposed for machine translation by [22]. Similar to LSTM, the GRU has gating units to model the flow of information inside a unit, without having separate memory cells [14]. Each GRU cell comprises two gates, namely the *reset gate*, that decides how much past information to forget while the *update gate* acts similar to the forget gate and input gate in LSTM [23]. A detailed description of the models can be found in [23] and [14].

(4) Existing Work on LSTM-RNN and GRU-RNN

An LSTM-RNN model with 1024 memory cells using stochastic gradient descent with gradient clipping was presented by [5]. In a sequence-to-sequence framework, they extracted knowledge on an IT helpdesk dataset to provide solutions to a technical problem via conversations and open-domain movie transcript dataset to perform common sense reasoning.

Also, [17] adopted this model to analyse the Ubuntu dataset, choosing the number of neurons as the hyper-parameter of the model. Their results showed that the LSTM-RNN outperforms the GRU-RNN on F1, accuracy, a recall and precision.

In addition, [13] developed a chatbot using high-level neural network libraries like Tensorflow [24] and Keras [18]. An LSTM model was evaluated on the Chatterbot Dialogue Corpus. The encoder comprises four layers, namely the *input layer* that encodes the input vector of length (40), the *embedding layer* of size (200), the *LSTM layer* of units (200) and the *dense layer* of dimension (141, 535), where the vocabulary size is (535). The decoder decodes the output vector of length (141) using the Greedy algorithm decoding strategy [25].

Furthermore, [26] compared a rule-based chatbot and LSTM Seq2Seq chatbot on a small training dataset of 300 turns of Information Technology (IT) service queries and responses. Their analysis has shown that the LSTM-Seq2Seq model generalised well to new input while the rule-based chatbot ensured better task completion rates. They measured how well the dialogue systems are performing using ROUGE automated evaluation metrics. The rule-based chatbot had a higher ROUGE score.

In [14], they compared LSTM-RNN, GRU- RNN and a tanh-RNN on polyphonic music modelling tasks and speech signal modelling tasks. Their experiment demonstrated the quality of LSTM-RNN and GRU-RNN over the traditional tanh-RNN, where the GRU-RNN outperformed the other two models

We observed that LSTM-RNN has been used extensively for open-domain conversational agent tasks compared to GRU-RNN. Therefore, this study will adopt a LSTM- RNN and GRU-RNN for our experiments.

(5) Transformer

Transformer, as presented in Fig. 2 is an encoder-decoder model with attention mechanism and feedforward neural network [10]; [2]. Transformer does not require sequential data to be processed in order. As a result, Transformer allows much more parallelisation [27] than a regular RNN.

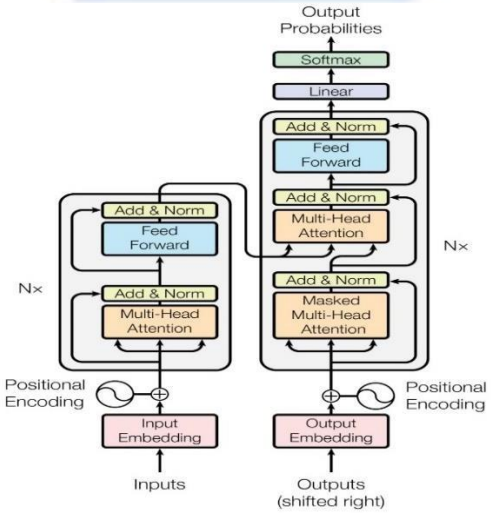

Fig. 2: Structure of a Typical Transformer Model [10]

In the encoder part, a vocabulary dictionary is created for the training data and a numeric index is assigned to each word say $x_0, \dots, x_n$. The embedding layer encodes the input and a positional embedding $e_p^0, \dots, e_p^n$ is introduced to obtain the model of the word order (context) in each sentence using the weight frequency formula in [10] as defined in "(2)"

$$PE_{pos,2i} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

$$PE_{pos,2i} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right) \quad (2)$$

where the cosweight frequency is for odd time-step, sin weight frequency refers to even time-step, pos indicates the order in the sentence, i is the indices of the position embedding dimension and d_{model} is the embedding vector.

First, the input embedding goes through a multi-head self-attention block which indicates how the word vectors are related to each other. To compute the attention scores, according to [3], the computation of the similarity between two vectors using cosine similarity is required. They further stated that similarity can be seen as a proxy for attention. Cosine similarity can be obtained by taking the dot-product between two vectors and dividing it by the magnitude for scaling purposes in "(3)."

$$Similarity(Q;K) = \frac{Q.K^T}{Scaling} \quad (3)$$

where Q is the target token called the query, K is all the tokens in the input known as the key. Since it is a similarity between matrices, it is necessary to transpose key vectors during matrices multiplication. The output is called the attention filter. The dot-product attention is scaled with the dimension of the key vector. The vector corresponding to each input token is converted to a value vector V, weighted by the corresponding attention weight and normalised. The final output of the multi-headed attention layer is given in "(4),"

$$Attention(Q,K,V) = Softmax_K\left(\frac{Q.K^T}{\sqrt{d_K}}\right).V \quad (4)$$

As the *Softmax* normalisation is done on the key, its values decide the amount of importance given to the query. The outputs from this block serve as input to a feedforward neural network to convert attention score to a form that can be propagated more efficiently across time steps. It consists of two linear transformations with a *ReLU* activation in between [28] as defined in "(5)."

$$FFN(x) = \max(0, xW_1 + b_1)W_2 + b_2 \quad (5)$$

where x is the input vector, W_1 and W_2 are the weight matrices while b_1 and b_2 are the bias vectors. Both blocks are augmented with residual connections and layer normalization. The residual connection helps the encoder not to forget the earlier information while neural normalisation performs the addition of the two endpoints that get to the feedforward layer. That is, the output of multi-headed attention (Layer (X)) and the original input X to that layer as given in "(6),"

$$LayerNormalization\ (X + Layer(X)) \quad (6)$$

In the decoder, the special token <start> is passed through the output embedding and converted into n-embedding vectors. The positional information is then added and passed to the multi-head attention layer together with the encoder output. The decoder has extra masked multi-head attention which computes the attention vector for current and prior words for learning purposes. The final linear layer in the decoder possesses a bunch of fully connected neurons where the size depends on the number classes. In the case of dialogue generation, the size of the final layer is the total vocabulary size. To get a scale of probability score for every word, the output matrix from the Add and Norm layer are flattened into a single row, concatenated and passed to the linear layer, which is another feedforward connection network. To get a single score for each word called Logits, these scores are passed into *Softmax* layer and converted into probabilities. Hence, the words with the highest probabilities for generation must be chosen. This process continues until the decoder generates a special <end> token.

(6) Existing Work on Transformer

For the task of language modelling, [29] stated that, only the decoder architecture is utilised. However, we found out from the work carried out in [12]; [30]; [31]; [32], adopted the internalisation of transfer learning-based transformer model for language generation tasks while the original Transformer model was used for machine translation [14]. [33] presented a Transformer chatbot tutorial with Tensorflow 2.0. The work utilised Cornell Movie Corpus and implemented a Transformer with Functional API and model subclassing in Keras [24]. However, the work only show few results without any evaluation.
This study will adopt a Transformer for our experiments

(7) Evaluation Techniques for Conversational Agents

The evaluation of conversational agents still remains a challenging research problem [34]; [35]. This paper highlights two of the main evaluation approaches to conversational agents:

a. Objective Evaluation: It is an automated method based on Bilingual Evaluation Study (BLEU) usually for machine translation and ROUGE score for text summarisation [35]. However, [36] pointed out the general metrics under these categories are coarse-grained (BLEU, ROUGE) and fine-grained evaluations. The former focuses on the adequacy of the responses generated (appropriateness) and the human likeness thereof (ability to imitate human behaviour), while the latter focuses on specific behaviours that a dialogue system should manifest. The two approaches are interwoven as the coarse-grained concept encapsulates many fine-grained concepts such as coherence, relevance and correctness. Despite this approach, according to [36], they do not correlate well with human expectations.
b. Subjective Evaluation: It is a human-based evaluation. According to [34], there are different approaches to human evaluation:
(i) Lab experiments: This involves evaluating dialogue systems in a lab environment. That is, users would be invited to fill out a questionnaire. For instance, [37] invited two business analysts to go through the answer to each testing question based on the following evaluation rules: grammatically correct, semantically related, well-spoken language, context-independent and not overly generalised. An answer will be labelled as suitable only if it satisfies all the rules, neutral if it satisfies the first three and breaks either of the latter two, and unsuitable otherwise. [36] referred to this approach as static context. Also, it is referred to as single-turn pairwise evaluation in [38].

(ii) In-field experiments/Dynamic Context: Here, the user is provided with an interactive interface to communicate with the dialogue systems. The Alexa Prize, that lets real users interact with operational systems and gathers user feedback over several months, is an analogy of this scenario. [38] referred to this as multi-turn Likert evaluation where performance is evaluated on a Likert (1-5) scale.

(iii) Crowdsourcing: This is done by using platforms such as Amazon Mechanical Turk (AMT). These platforms provide large numbers of recruited users. LEGOEval [39] is an open-source toolkit that enables researchers to easily build and deploy their human evaluation tasks on AMT. LEGOEval supports representative human evaluation tasks, such as static evaluation, where crowd workers are asked to rate sampled dialogues, and interactive evaluation, where crowd workers interact with two systems and evaluate their responses.

In this study, a subjective evaluation approach with a lab experiment will be adopted, following the method used in [37] and [40]. The models will be examined using four criteria: Repetitive, Interestingness, Making Sense and Fluency [20]. Three analysts were recruited to administer the generated response questions. This study differs from other work showing the differences in performances of LSTM-RNN, GRUs-RNN and transformer using three corpora.

III. EXPERIMENTAL SETUP

This section describes our experimental results with the three datasets, showing some examples of the interactions with the applications that we trained. Also, we compare the performance of the system using human evaluation on a set of 60 questions.

(1) Datasets

We experiment with two single-turn open-domain dialogue datasets and one multi-turn task-specific domain.

(2) Chatterbot Dialogue Corpus

The Chatterbot Dialogue Corpus is used by [15] as part of a Python library to generate automated responses to user inputs. ChatterBot Dialogue Corpus is a piece of user-contributed questions and answers from various categories such as computers, emotion, food, greetings, health, movies, politics, sports and more. The dataset is formatted using yet another markup language (YAML); a human-friendly data serialisation language for all programming languages. We considered greetings, computers, artificial intelligence, sports, science and politics ChatterBot Dialogue Corpus as leverage to build a chatbot that converses with humans. It contains 2k dialogues with an average of 15 questions-answers per category. It was downloaded from https://github.com/gunthercox/chatterbot-corpus

(3) Cornell Movie Corpus

The Cornell Movie Corpus by [16] is a collection of fictional conversations extracted from raw movie scripts. It contains 220000 conversational exchanges between 1000 0 pairs of movie characters with 304000 utterances. Nine thousand characters across 617 movies were involved w here *movie_conversation.txt* has the following format: ID of the first character, ID of the second character, ID of the movie in which this is conversation occurred, and a list of line IDs and *movie_lines.txt* containing the following format: ID of the conversation line, ID of the character who uttered this phase, ID of the movie, name of the character and the text of the line [33].

(4) Ubuntu Dialogue Corpus

The Ubuntu Dialogue Corpus by [17] is unstructured multi-turn goal-oriented dialogue systems where Ubuntu chat logs are used for obtaining technical support with various Ubuntu issues and in some cases, users discuss issues not related to Ubuntu. It contains 930,000 dialogues spanning 100 billion words. To make it a dyadic conversation, we converted Ubuntu Dialogue Corpus in CSV file into text file, stored in empty lists, iterated through the lists and parsed the conversations into questions and answers sets through Python codes.

(5) Text Preprocessing

Since text data is known to be noisy, several data preprocessing steps were performed. Special characters and contractions in each

sentence were removed. A tokeniser was used to tokenise each sentence, adding *START_TOKEN* and *END_TOKEN* indicating the start and end of each sentence, filtering out sentences with more than the maximum length tokens and padding tokenised sentences to maximum length. Each dataset split into a training and test set is 80% by 20%. The cleaned dataset is used to train an end-to-end conversational agent for 70 epochs including the vocabulary sizes in thousands of words after initialising the tokeniser, as shown in Table I. However, the dataset sizes considered is to make the experiment small and fast in terms of computational complexity.

Table 1 Vocabulary Sizes

Corpus	**Dataset sizes (Dialogues)**	**Vocab sizes**
ChatterBot	2×10^3	2k
Cornell Movie	2.2×10^5	10k
Ubuntu	9.3×10^5	20k

(6) Model Implementation and Optimisation

Following the same direction of work on generative conversation systems as [13], we used the Keras Functional API to create encoder-decoder and transformer models for the chatbots.

(7) LSTM-RNN and GRU-RNN Models

We implemented the encoder-decoder model by passing all the required configuration settings such as the dimension of the encoder input, embeddings dimension, and one layer of LSTM and GRU. Both embeddings and hidden layers had a size of 200. In the decoder part, it follows a similar structure with the encoder except for the addition of the dense layer that maps the output from the LSTM or GRU to the dimension of the vocabulary size. The models were trained for 70 epochs with a batch size of 48, Categorical Cross Entropy was adopted as a loss function while *Adam* optimiser [41] was used to minimise the loss.

(8) Transformer Model

We used the original transformer model that consists of an encoder part and a decoder part as presented by [33]. It is a two-layer model that has eight attention heads, 512 hidden vector units, embedding vectors size of 256, a feedforward network with *ReLu* activation and a dropout rate of 0.1. This choice of hyperparameter is to keep the model small and relatively fast. We used the Adam optimiser with a custom learning rate scheduler according to [10]. Unlike the sequence-to-sequence models where the batch size is specified as the number of sentence pairs, in the case of transformer model, the batch size which is a copy of model parameters, specifies the approximate number of tokens in one batch to be 64 multiplied by the training devices (that is, Tensorflow Processing Units (TPU)) using *tf.distribute.strategy* class in Tensorflow).

Furthermore, we implemented the self-attention along with the multi-head attention taking three inputs, namely Q (query), K (key) and V (value). These were put through linear (Dense) layers and split up into multiple heads. Positional encoding was added to give the model some information about the relative position of the words in the sentence. Each encoder layer consisted of sublayers: Multi-head attention (with padding mask) and two dense layers followed by dropout. Each of these sublayers has a residual connection around it which help in avoiding the vanishing gradient problem followed by a layer normalisation *tf.keras.layers.LayerNormalisation*, which makes it easier to train the model. Likewise, the decoder layer consists of sublayers: Masked multi-head attention (with a look-ahead mask and padding mask), Multi-head attention (with a padding mask) and two dense layers followed by dropout. Each of these sublayers has a residual connection around it followed by a layer normalisation. We trained our transformer for 100 epochs by simply calling *model,fit()*. Fig. 3 presents the components of our conversational agents.

Datasets

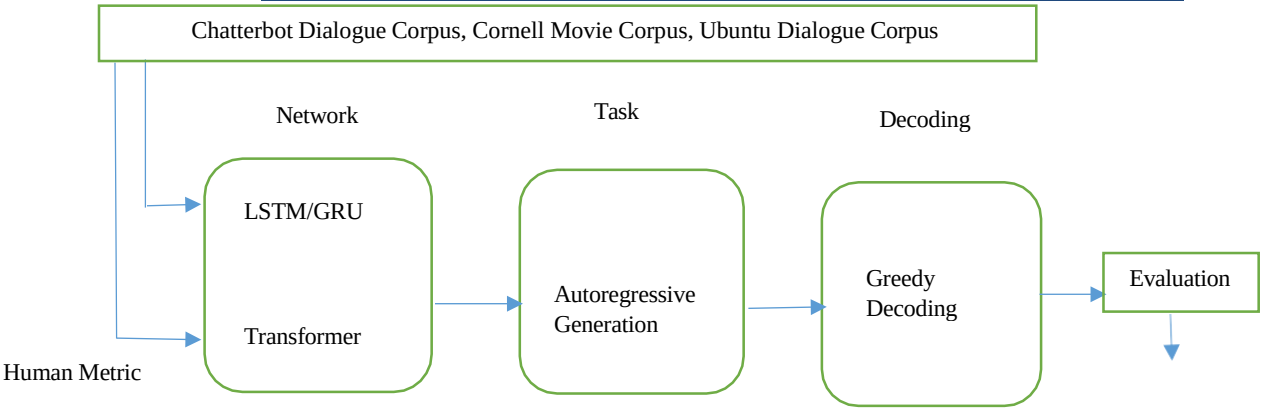

Fig. 3: Proposed Conversational Agent Components

The decoder is a conditional language model (autoregressive generation) that employs the Greedy decoding algorithm [42] and [25]. To generate the highest scoring item at each time steps, the argmax mathematical function provided in NumPy was used based on the sequence generated so far and the next utterance.

IV. RESULTS AND DISCUSSION

This section presents our results and observations identified during the empirical study.

(1) Evaluation Method

This research work adopted human evaluators to measure the responses produced by the different models. We pre- tested the survey with two target groups to see whether they understand the evaluation criteria and whether there is any confusion or misinterpretation of the questions. Having responded to the comments and suggestions of the pilot target, we effected those changes and administered the questionnaire to three analysts. One is a linguistic practitioner and the other two are postgraduate students. Responses obtained from the three evaluated models were administered to the human evaluators with the following options, namely unsuitable, neutral and suitable. The evaluation set-up required the human judgement to rate a model according to specific criteria [20]:

(a) **Repetitive:** A response is considered not repetitive if there is no (1) Internal repetition – No repeating words or phrase within response.
(2) Repetition across responses – Different response narration is the same. (3) Partner repetition – Repeating what the user says.
(b) **Interestingness:** A response is somewhat interesting if it presents information in an engaging manner. That is, an interesting response that can be followed up and holds one's attention.
(c) **Making Sense:** A conversation is considered to make sense if it is coherent. If the system's responses are relevant and semantically appropriate throughout a conversation. Also, maintains good flow and does not change topics or lose track of the conversation.
(d) **Fluency:** A response is considered fluently written if it is grammatically correct and well- spoken.

(2) ChatterBot Dialogue Corpus Evaluation Result

In hindsight, it is important to note that different responses were generated by the system. However, we have presented a small trial to demonstrate the evaluation process due to space constraint.

Table 2 presents the summary rating responses of the human evaluators for ChatterBot Dialogue Corpus for each evaluated model after considering the majority voting. The evaluators marked generated response with the following interaction set-up along with the rules defined above:

(a) **Suitable:** means response is appropriate and satisfies all the rules.
(b) **Neutral:** indicates response is less appropriate in certain contexts.
(c) **Unsuitable:** means the response does not satisfy the rules at all.

Table 2: Summary of evaluation results for the chatterbot dialogue corpus

User Question	*"what language are you written in"*

Golden Standard Answer	*"i am written in python"*
System Response (LSTM)	*"python"*
System Response (GRU)	*"I am written in python."*
System Response (Transformer)	*"python."*

	Repetitive	**Interestingness**	**Making Sense**	**Fluency**
Human Evaluator 1	Suitable	Suitable	Suitable	Neutral
Human Evaluator 2	Suitable	Suitable	Suitable	Suitable
Human Evaluator 3	Suitable	Suitable	Suitable	Neutral

From Table 2 on each criterion, the three human evaluators rated the context to be 100% not repetitive, interestingness and making sense while 99% fluency. To establish these facts, we count how many times the system response is rated as suitable or neutral for each criterion that make a good conversation. In total, we have 12 ratings. To know this, we graded the nominal variables to be "0" for unsuitable, "1" for neutral and "2" represent suitable. 10 rating with suitable response and 2 rating with neutral response. However, we measured the degree of agreement among the evaluators using the Fleiss kappa inter-rater reliability [43]; [44]. For this, we divided the number of rating of suitable response and neutral response by the total number of rating, with that we have 83% of system response rated as suitable response by the raters and 16% of system response rated as neutral. Thereafter, we calculated the kappa; a measure of rater agreement using the formula $P_o - P_e / 1 - P_e$ where P_o is the observed agreement and P_e is the random expected agreement judgement. For this purpose, the Fleiss kappa k is calculated to be 1.52.

(3) Cornell Movie Corpus Evaluation Result

We applied the same approach in Table 3. On each criterion, the human evaluators rated the context to be 100% not repetitive, no agreement between the human evaluators in terms of interestingness while making sense and fluency scores on each category is 66%. To ascertain these facts, we computed the inter-rater reliability which gives−1, indicating there is disagreement between the raters

Table 3: Summary of evaluation results for the cornell movie corpus

User Question	*"not the hacking and gagging and spitting part. please"*
Golden Standard Answer	*"okay then how about we try out some french cuisine . saturday night ?"*
System Response (LSTM)	*"okay. how?"*
System Response (GRU)	*"i do not want it either."*

System Response (Transformer)	*"i do not know, you are talking like you are confused but you are a surprise."*			
	Repetitive	**Interestingness**	**Making Sense**	**Fluency**
Human Evaluator 1	Suitable	Not applicable	Unsuitable	Unsuitable
Human Evaluator 2	Suitable	Not applicable	Suitable	Suitable
Human Evaluator 3	Suitable	Suitable	Suitable	Suitable

(4) Ubuntu Dialogue Corpus Evaluation Result

The Ubuntu Dialogue Corpus evaluation result is presented in Table 4 with 100% not repetitive, 99% interestingness and 66% fluency and making sense. For this, we computed the inter-rater agreement to be -0.40, also indicating slight disagreement between the human evaluators.

Table 4: Summary of evaluation results for the ubuntu dialogue corpus

User Question	*"did i choose a bad channel? i ask because you seem to be dumb like windows user."*			
Golden Standard Answer	*"the second sentence is better english and we are not dumb"*			
System Response (LSTM)	*"nope not sure we are not dumb"*			
System Response (GRU)	*"what are you trying to do."*			
System Response (Transformer)	*"oh , well that is what i am."*			
	Repetitive	**Interestingness**	**Making Sense**	**Fluency**
Human Evaluator 1	Suitable	Neutral	Unsuitable	Unsuitable
Human Evaluator 2	Suitable	Not Suitable	Suitable	Suitable
Human Evaluator 3	Suitable	Suitable	Neutral	Suitable

Having obtained the judgements given by the three human evaluators, Table 5 gives the consensus in rating by the human evaluators indicating that ChatterBot Dialogue Corpus context is interestingness, fluency and making sense compared to the Cornell Movie Corpus and the Ubuntu Dialogue Corpus. Another perspective is traced to the corpus used, especially the Cornell Movie Corpus and the Ubuntu Dialogue Corpus, as they are not well curated in terms of grammatical errors hence, affected the seq2seq models response fluency.

Model Evaluated (LSTM. GRU and Transformer)	**K Value**

Table 5: Human evaluator agreement

Context with ChatterBot Dialogue Corpus	1.52
Context with Cornell Movie Corpus	-1
Context with Ubuntu Dialogue Corpus	-0.40

(5) A/B Testing for Interestingness

To further justify the performance of the models, and ascertain the fact that each time the same question is asked, the system gives a different response, we conducted an A/B test, for a simple randomised controlled experiment in which two samples (A and B) of a single vector variable are compared. As in [37], we employed top-1 accuracy (P_{top1}) as the criterion. This formula is given as $P_{top1} = (G_{suitable} + G_{neutral})/G_{total}$ and measures whether the generated responses are suitable or neutral as depicted in Table 6.

Table 6: Model comparison using a/b test

Model	G_{total}	$G_{unsuitabl}$	$G_{neutral}$	$G_{suitable}$	P_{top1}
LSTM	50	12	18	20	76%
GRU	50	8	20	22	84%
Transfor mer	51	11	21	19	78.43 %

As shown in Table 6, GRU-RNN has a P_{top1} of 84% which is much higher than LSTM-RNN of 76% and Transformer of 78%. This finding suggests that A/B testing can successfully determine interestingness responses for open domain conversational agents.

(6) Results and Discussion

The examining of the encoder-decoder and transformer models for conversational agents is based on the need to gain more knowledge in order to solve complex problems. In the previous sections we have shown the stages involved in actualizing this empirical study.

Our results are presented in Table 5 and 6. Table 5 reports the level of agreement between the human evaluators known as *k* on the second column, where the context with Chatterbot Corpus measured 100% agreements. This shows the importance of the evaluator's reliability in the sense that, Chatterbot Corpus is interesting, making sense and fluent compared to the Cornell Movie Corpus and the Ubuntu Dialogue Corpus. Table 6 presents the model comparison using the A/B test, and the top_1 accuracy of the GRU-RNN model measured 84%. This shows that the context generated by the GRU-RNN model is meaningful short text responses compared to the LSTM- RNN and transformer models. However, the study also shows clearly that the transformer model can handle long- range dependencies with ease when compared to LSTM- RNN and GRU-RNN.

From the experiment carried out and what we discovered from the existing works, we make the following observations:

1. The decoder is the language model that generates the target sentence conditioned with the encoding created by the encoder. However, we realise that neural conversational agents do not take into account the context of the conversation as the encoder only read the current input and ignored previously stated input.
2. We realise that the transformer model is unable to capture dependencies longer than 512 words. This issue shaped our understanding of how the subword-based tokenisation algorithm splits long texts into shorter chunks and feeds them to the model separately.
3. A self-attention mechanism is computationally expensive. We noticed that the process of producing a summary vector (that is the attention weight) is repeated for every token in the input.
4. We noticed that the generative bots, especially the transformer, gave meaningful responses that were not from a set of answers, which demonstrates transformer model is able to generate words in context to the conversation.

The bar chart distribution of the most common attributes between the evaluators mark is presented in Fig. 4 where it shows there was a consistent agreement between the human judgement in terms of avoiding repetition, interestingness and making sense. Fig. 5 depicts there is a total disagreement between the evaluators in terms of interestingness. The same was disagreement between one of the evaluators in making sense and fluency. In Fig. 6, a slight disagreement occurs in fluency and making sense. These outcomes indicated that the corpus utilised for the chatbot must be well-curated to engage the users. Also, the distribution of the A/B testing is shown in Fig. 7, indicating that GRU-RNN outperformed LSTM-RNN and Transformer in terms of generating interesting short text responses.

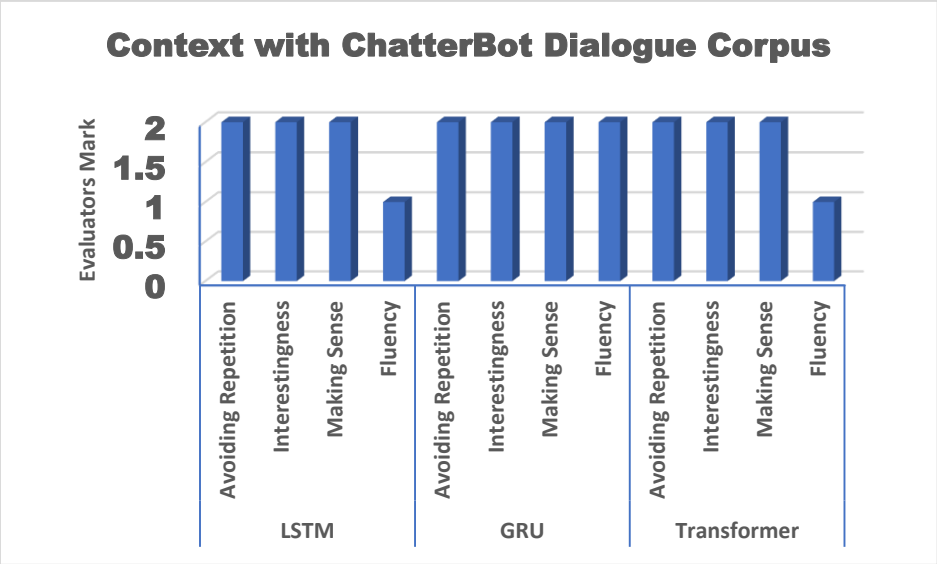

Fig. 4: Context with ChatterBot Dialogue Corpus

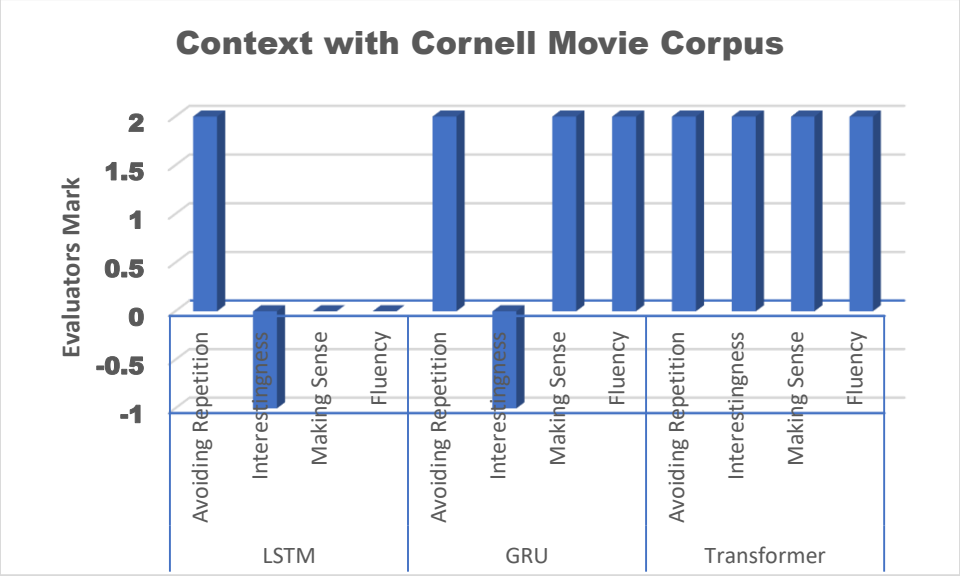

Fig. 5: Context with Cornell Movie Corpus

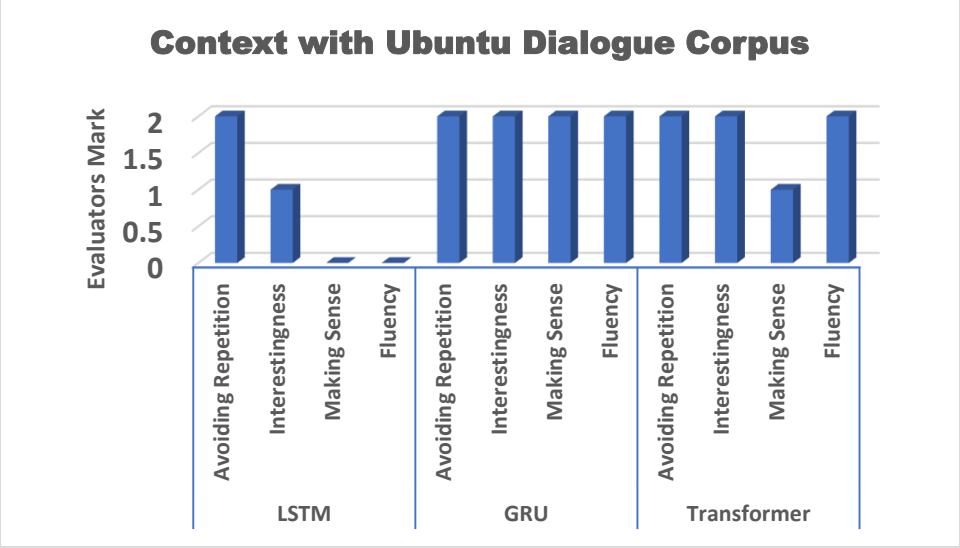

Fig. 6: Context with Ubuntu Dialogue Corpus

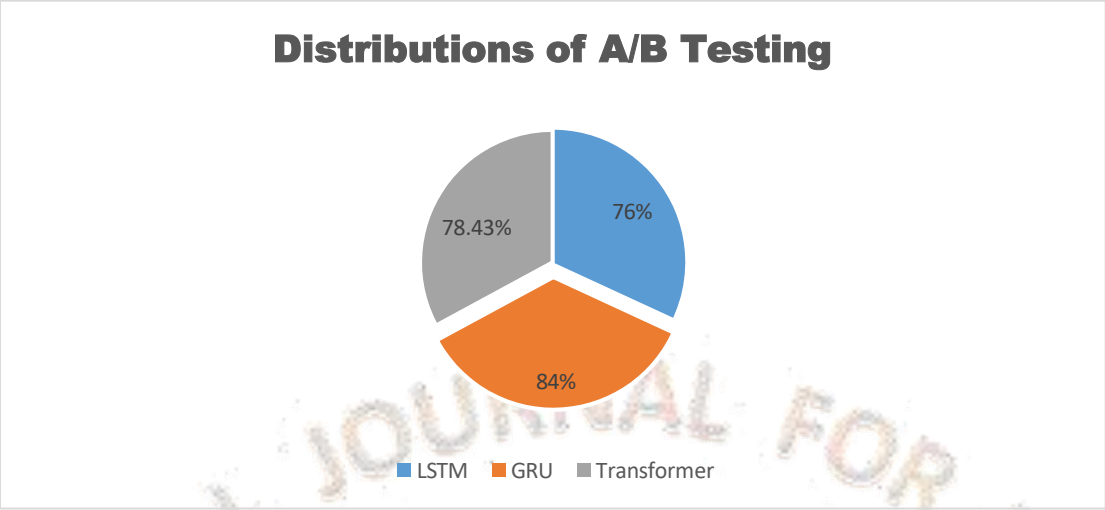

Fig. 7: Distributions of A/B Testing

V. CONCLUSION

In this paper, we examined the application of encoder- decoder and transformer models for conversational agents. The models operate in an autoregressive manner to generate natural utterances based on the current state of the conversation. This was achieved with the help of decoder after feeding the sentence representation produced by the encoder, as well as a special token <START>, which indicates the start of a sentence. These two inputs got processed and produces the first hidden state, passed to the linear layer and *Softmax* layer to produce the largest probability distribution over the vocabulary. However, this is repeated until the special token <END> is reached. This type of decoding process is called greedy algorithm.

As mentioned previously, human evaluation was used to evaluate the models where, repetitive, interestingness, making sense and fluency were the concepts considered by human to evaluate the conversational agents.

We have shown that the GRU-RNN models outperform the LSTM-RNN and transformer models in terms of generating interesting short text responses. The LSTM-RNN model, also had the capability of generating short text responses that were not quite meaningful, while the transformer model managed long-range dependencies with ease.

We intend to include Encoder-decoder model with attention mechanism in our future research work in order to improve their response generation quality.

Attention, is a mechanism in neural networks that focuses on a specific part of the input and computes its context-dependent summary.

In addition, future work includes curating personalised datasets and incorporating into response generation models examined, coupled with pre-trained models such as the Bidirectional Auto-Regressive Transformer (BART) [45] for complex tasks such as using our low resource languages to engage in conversation.

VI. REFERENCES

[1] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. MIT Press, 2016.

[2] M. Lundell Vinkler and P. Yu, “Conversational Chatbots with Memory-based Question and Answer Generation.” Department of Science and Technology, Linkoping University, Sweden, 2020.

[3] D. Jurafsky and J. H. Martin, "Speech and Language Processing, Chapter 15: Vector Semantics," Draft of August 7, 2017.

[4] J. Gao, M. Galley, and L. Li, “Neural approaches to conversational AI,” vol. 13, no. 2–3, pp. 127–298 2019.

[5] O. Vinyals and Q. Le, “A neural conversational model,” arXiv Prepr. arXiv1506.05869, 2015.

[6] R. Csaky, “Convolutional Sequence to Sequence Learning,” arXiv Prepr. arXiv1908.08835, 2019.

[7] C. Manning, R. Socher, G. G. Fang, and R. Mundra, “CS224n: Natural Language Processing with Deep Learning1,” 2017.

[8] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J-Y. Nie, J. Gao and B. Dolan: “A neural network approach to context-sensitive generation of conversational responses”; arXiv Prepr. arXiv1506.06714, 2015.

[9] A. Fan, M. Lewis, and Y. Dauphin, “Hierarchical neural story generation,” arXiv Prepr. arXiv1805.04833, 2018.

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin: "Attention is all you need"; 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA NIPS 2017.

[11] Q. Fournier, G. M. Caron, and D. Aloise, "A practical survey on faster and lighter transformers," arXiv Prepr. arXiv2103.14636, 2021.

[12] T. Wolf, V. Sanh, J. Chaumond, and C. Delangue, "Transfertransfo: A transfer learning approach for neural network-based conversational agents," arXiv Prepr. arXiv1901.08149, 2019.

[13] S. Panchal : "Creating a Chatbot from Scratch Using Keras and Tensorflow"; 2019.

[14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio: "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence modeling"; arXiv Prepr. arXiv1412.3555, 2014.

[15] G. Cox, "ChatterBot Documentation" Release 0.8.7 January 25, 2019.

[16] D-N-M. Cristian and L. Lee, "Chameleons in Imagined Conversations: A New Approach to Understanding Coordination of Linguistic Style in Dialogs" Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL, 2011.

[17] R. Lowe, N. Pow, I. Serban, and J. Pineau, "The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems," arXiv Prepr. arXiv1506.08909, 2015.

[18] F. Chollet. "Keras." Retrieved from https://keras.io/

[19] C. Manning and R. Socher, "Natural Language Processing with Deep Learning CS224N/Ling284." 2019.

[20] A. See, S. Roller, D. Kiela, and J. Weston, "What makes a good conversation? how controllable attributes affect human judgments," arXiv Prepr. arXiv1902.08654, 2019.

[21] B. Sun and K. Li, "Neural Dialogue Generation Methods in Open Domain: A Survey," Nat. Lang. Process. Res., vol. 1, no. 3–4, pp. 56–70, 2021.

[22] K. Cho, B. V. Merriënboer, D. Bahdanau, and Y. Bengio: "On the properties of neural machine translation: Encoder-decoder approaches" arXiv Prepr. arXiv1409.1259, 2014.

[23] R. Fu, Z. Zhang, and L. Li: "Using LSTM and GRU neural network methods for traffic flow prediction" 31st Youth Academic Annual Conference of Chinese Association of Automation, pp. 324–328, YAC 2016.

[24] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu and X. Zheng : "TensorFlow: A System for Large-Scale Machine Learning" Retrieved from tensorflow.org 2016.

[25] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy: "Deep Reinforcement Learning for Sequence-to-Sequence Models." https://arxiv.org/pdf/1805.09461.pdf 2019.

[26] M. M. H. Dihyat, and J. Hough: "Can rule-based chatbots outperform Neural models without pre-training in Small Data Situations? A Preliminary Comparison of AIML and Seq2Seq"; School of Electronic Engineering and Computer science, Queen mary University of London, UK, 2021.

[27] A. Pathak: "Comparative Analysis of Transformer based Language Models"; CS & IT Conference Proceedings, vol. 11, no.1, 2021.

[28] R. S. Csaky, "Deep Learning Based Chatbot Models," arXiv: 1908.08835v1, 23 August, 2019.

[29] S. Mesham, L. Hayward, J. Shapiro, and J. Buys, "Low- Resource Language Modelling of South African Languages," arXiv Prepr. arXiv2104.00772, 2021.

[30] Lewis M., Liu Y., Goyal N., Ghazvininejad M., Mohamed A., Levy O,, Stoyanov V., and Zettlemoyer L.:"Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension" arXiv Prepr. arXiv1910.13461, 2019.

[31] C. Raffel, N. Shazeer, A. Robert, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu.: "Exploring the limits of transfer learning with a unified text-to-text transformer"; arXiv Prepr. arXiv1910.10683, 2019.

[32] J. Zhang, Y. Zhao, M. Saleh, and P. Liu: "Pegasus: Pre- training with extracted gap-sentences for abstractive summarization"; arXiv Prepr. arXiv:1912.08777v3, 2020.

[33] B. M. Li: "A Transformer Chatbot Tutorial with Tensorflow 2.0"; May 23 2019.

[34] J. Deriu, A. Rodrigo, A. Otegi, G. Echegoyen, S. Rosset, E. Agirre, M. Cieliebak:"Survey on evaluation methods for dialogue systems"; arXiv Prepr. arXiv;1905.04071v2, 2021.

[35] A. Venkatesh, C. Khatri, A. Ram, F. Guo, R. Gabriel, A. Nagar, R. Prasad, M. Cheng, B. Hedayatnia, A. Metallinou, R. Goel, S. Yang, and A. Raju: "On evaluating and comparing conversational agents"; 31st Conference on Neural Information Processing Systems, Long Beach, CA, USA, NIPS 2017.

[36] J. M. Deriu: "Evaluation of Dialogue Systems"; University of Zurich, Zurich. 2021

[37] M. Qiu, F-L. Li, S. Wang, X. Gao, Y. Chen, W. Zhao, H. Chen, J. Huang, and W. Chu: "Alime chat: A sequence to sequence and rerank based chatbot engine"; Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, pp. 498–503. 2017

[38] S. Roller, Y-L. Boureau, J. Weston, A. Bordes, E. Dinan, A. Fan, D. Gunning, D. Ju, M. Li, S. Poff, P. Ringshia, K. Shuster, E. M. Smith, A. Szlam, J. Urbanek, and M. Williamson: "Open-Domain Conversational Agents: Current Progress, Open Problems, and Future Directions"; arXiv:2006.12442v2, 2020.

[39] Y. Li, J. Arnold, F. Yan, W. Shi, and Z. Yu : "LEGOEval: An Open-Source Toolkit for Dialogue System Evaluation via Crowdsourcing" ;arXiv Prepr. arXiv2105.01992, 2021.

[40] L. Shang, Z. Lu, and H. Li: "Neural Responding Machine for Short-Text Conversation"; Noah's Ark Lab, Huawei Technologies Co. Ltd. Sha Tin, Hong Kong. arXiv:1503.02364v2, 2015.

[41] D. P. Kingma, and J. Ba: "Adam: A method for stochastic optimization". arXiv Prepr. arXiv1412.6980, 2014.

[42] A. See: "Natural Language Processing with Deep Learning". Retrieved from https://web.stanford.edu/class/cs224n/slides/cs224n-2019- lecture15-nlg.pdf 2018.

[43] J. L. Fleiss:"Measuring Nominal Scale Agreement Among many Raters"; Psychological Bulletin, 76 (5): 378. 1971.

[44] J. R. Landis and G. G. Koch: "The Measurement of Observer Agreement for Categorical Data"; 1977.

[45] M. D. Bruyn, E. Lotfi, J. Buhmann, and W. Daelemans: "BART for Knowledge Grounded Conversations" ; CEUR- WS.org/Vo1-2666/KDD_Converse20_paper_7.pdf , 2020.