IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 16, 2020, accepted February 24, 2020, date of publication March 5, 2020, date of current version March 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2978551

Enhancements to the Sequence-to-Sequence-Based Natural Answer Generation Models

KULOTHUNKAN PALASUNDRAM, NURFADHLINA MOHD SHAREF, KHAIRUL AZHAR KASMIRAN, AND AZREEN AZMAN

Intelligent Computing Research Group, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Selangor 43400, Malaysia

Corresponding author: Nurfadhlina Mohd Sharef (nurfadhlina@upm.edu.my)

This work was supported in part by the Intelligent Computing Research Group, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Malaysia, and in part by the Asian Office of Airforce Research and Development (AOARD) on Multi-Objective Deep Recurrent Reinforcement Learning Research.

ABSTRACT There is a great interest shown by academic researchers to continuously improve the sequence-to-sequence (Seq2Seq) model for natural answer generation (NAG) in chatbots. The Seq2Seq model shows a weakness whereby the model tends to generate answers that are generic, meaningless and inconsistent with the questions. However, a comprehensive literature review on the factors contributing to the weakness and potential solutions are still missing. Therefore, this review article fills the gap by reviewing Seq2Seq based natural answer generation-based literature to identify those factors and proposed methods to address the weakness. This literature review identified several factors such as input question is not sufficient to determine a meaningful output, usage of cross-entropy function as the loss function during training, infrequent words in training data, language model influence which generates answers not relevant to the question, utilization of teacher forcing method during training which results in exposure bias, long sentences and inability to consider dialogue history as the factors. Additionally, this literature review also identified and reviewed the methods proposed to address the weakness such as utilizing additional embedding and encoders, using different loss functions and training approaches, as well as utilizing other mechanisms like copying source word(s) and paying attention to a certain portion of the input. For discussion, these methods are categorized into four broad categories which are *Structural Modifications*, *Augmented Learning*, *Beam Search* and *Complementary Mechanisms*. Additionally, the paper highlights unexplored areas in Seq2Seq modeling and proposes potential future works for natural answer generation.

INDEX TERMS Seq2Seq, natural answer generation, natural language processing, dialogue generation, chatbot.

I. INTRODUCTION

Chatbots are online computer systems that can mimic humans, and converse with humans using natural language. A chatbot should carry-out three (3) basic functions:- dialogical agent (able to comprehend the topic or problem and respond accordingly), rational agent (has the competence to analyze and handle the user input in natural language and respond logically) and embodied agent (can express feelings like a human and gain trust of the user to engage it) [3].

Usage of chatbots as the customer interface to provide answers to queries round the clock is fast gaining momentum. Chatbots equipped with reasoning and knowledge can scale much faster than human personnel. Additionally, chatbots help in keeping the costs lower than utilizing humans over several channels like phone, email, live chat, and message boards [1], [2].

The triggering point for chatbot development can be attributed to Alan Turing, who proposed a question ''Can machines think?'' in his article titled ''Computing Machinery And Intelligence'' which was published in 1950 [4]. In 1966, a chatbot called ''ELIZA'' was developed to study the natural language communication between man and machine [5]–[7]. Since then hundreds of thousands of chatbot have been developed especially with the help of chatbot platforms launched by technology companies

The associate editor coordinating the review of this manuscript and approving it for publication was Joey Tianyi Zhou.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

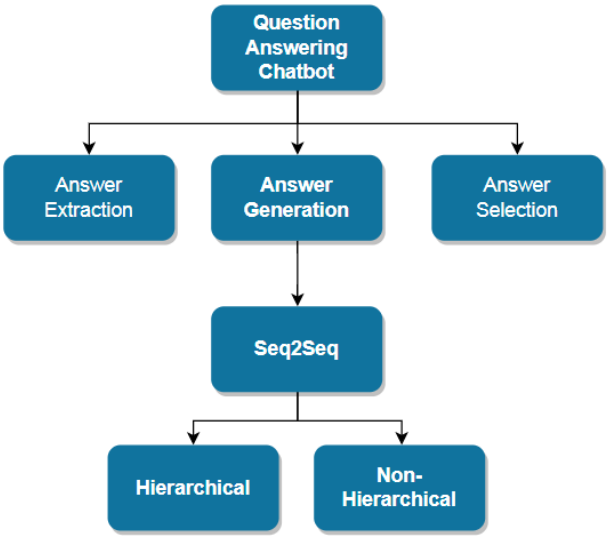

FIGURE 1. Question answering chatbot types. Seq2Seq based natural answer generation models are the focus of this review.

as Api.ai, Amazon, Chatfuel, Facebook, Google, IBM and Microsoft [8].

Question-answering chatbots aims at automatically providing natural answers from structured knowledge-base or unstructured documents to questions posed by users in natural language [9]–[11]. Question-answering chatbots can be modeled to provide a natural answer to a given question based on three methods which are the answer selection, answer extraction and answer generation. Existing natural language question-answering chatbots are typically based on answer selection or answer extraction, and this leaves much space for further research in natural answer generation methods. A natural answer generation (NAG) task can be defined as ''given a sequence of words (question sentence in natural language), generate an accurate answer in natural language (another sequence of words) for that question''. The generated answer must be correct, coherent and natural.

In recent years, many researchers have investigated and worked on Recurrent Neural Network (RNN) based Encoder-Decoder generative models (*commonly referred to as Seq2Seq models*), which was originally proposed for neural machine translation (NMT), to develop NAG chatbots. As shown in figure 1, Seq2Seq models can be further broken down into hierarchical (*models conversation history / long conversations*) and non-hierarchical (*models single question to answer*) models [2], [12]–[17].

There are several recent review papers written on natural language processing and generation. [18]–[20] covered broadly, on a high level, on the advances on NAG techniques and reviewed only a few aspects of the Seq2Seq model. References [21], [22] focused more on the trend without much emphasis on the issues with Seq2Seq learning. However, an in-depth review that focuses on issues of applying the Seq2Seq model to generate natural answers and proposals to address them has not been published. *This paper fills this gap by discussing ways to improve the Seq2Seq models in NAG settings*. Figure 2 provides a peek into the methods proposed by researchers and reviewed in this paper.

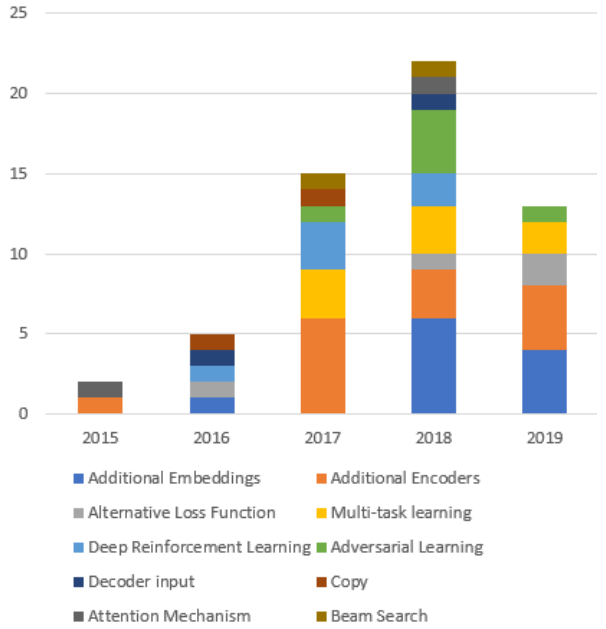

FIGURE 2. Enhancement methods proposed for Seq2Seq based natural answer generation by year.

In summary, there are two (2) key contributions to the literature on Seq2Seq model. First, weaknesses and the contributing factors found in Seq2Seq based NAG models were reviewed. Second, the various enhancement methods proposed by researchers to address those weaknesses were reviewed. With this, future researchers planning to work on the Seq2Seq model especially in the natural answer generation domain will be able to have a thorough understanding of any potential issues they may encounter and existing solutions to address them.

This paper has been structured in the following sections. First, in section 2, a description of the Seq2Seq model, its key components and algorithms are provided. Section 3 explains the weaknesses found in Seq2Seq based natural answer generation models and factors contributing to those weaknesses, followed by section 4 which reviews the enhancement methods proposed by researchers to address those weaknesses. Section 5 provides the conclusion and proposed future works.

II. SEQ2SEQ MODEL

Figure 3 shows a Seq2Seq model which consists of an encoder and a decoder. The Encoder aims to represent the meaning of the input sentence. It does it by encoding the input sentence into a dense vector.

The following activity happens during the encoding phase:-

1) The question string (''How are you ?'') is first tokenized into a list of tokens. A token can be a word or a sub-word of one (1) or more characters and symbols.
2) Each token is then associated with a vector which is a subset of input vocabulary. Out of vocabulary tokens

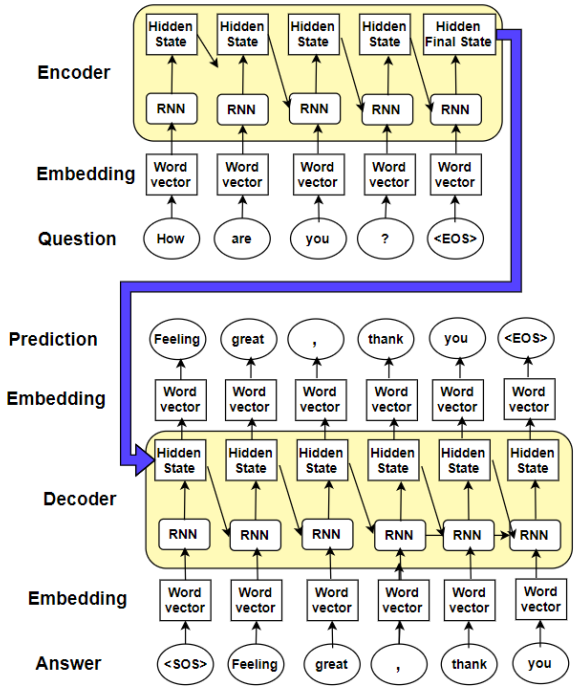

FIGURE 3. Illustration of a Seq2Seq Model with attention mechanism. The question is converted to embedding and then encoded to hidden states. Weighted hidden states are used by the decoder to generate the answer.

are associated with <UNK> token. An end of sequence (<EOS>) token is added at the end. This sequence of vectors (called the embedding) will then be passed to the Encoder. Embedding can be either pre-trained such as Word2Vec [23] and Glove [24] or jointly trained during model training.

3) The encoder network will process this embedding and produce intermediate states (fixed-length vectors) beginning with the first word and then updating the state for the next token encountered and eventually for the whole sentence. In other words, the encoder's final hidden state (also known as context vector) represents the sentence encoding (captures the meaning of this input sentence). Equation (1) and (2) show how intermediate hidden states and context vector are computed respectively. f_1 *and* f_2 can be any nonlinear functions such as tanh, Long Short-Term Memory (LSTM) [16], [25] or Gated Recurrent Unit (GRU) [15]. The encoder behaves the same way during training and testing.

Most of the question-answering chatbot NAG models made use of bidirectional encoders as originally proposed by [16]. In a bidirectional encoder, there will be one

$$h_t = f_1(x_t, h_{t-1}) \tag{1}$$

$$c = f_2(h_1, h_2, \ldots, h_T) \tag{2}$$

forward RNN and one backward RNN. An input sequence is processed in both directions (forward and backward). Then, the forward and backward hidden states are concatenated before passing to the decoder.

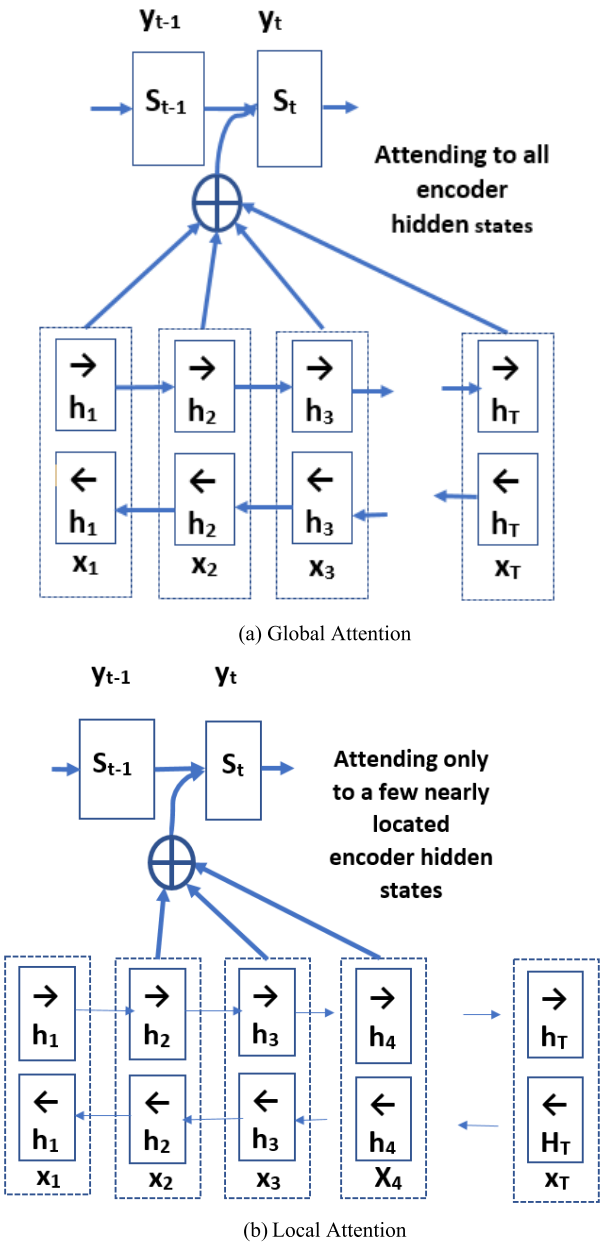

FIGURE 4. Illustration of attention mechanisms. (a) Global Attention–considers all the hidden states, (b) Local Attention Mechanisms - considers selected hidden states based on the size of a calculated window.

The decoder aims to generate an answer based on the encoder's hidden states. The following activity happens during the decoding phase:

1) As depicted in Figure 4, using either a global attention [26] or a local attention [27] mechanism, the decoder first scores the alignment between input tokens and output tokens to compute the context vector of time t (3) which is the weighted sum of all the encoder's intermediate hidden states.
2) Based on this computed context vector and the start of sequence token (<SOS>), the conditional probability of the target tokens (4) is computed to generate/predict

Algorithm 1 Training a Seq2Seq Model With Attention Mechanism for Answer Generation

Input: Question (X)-Answer(Y) pairs, Maximum Answer Tokens to Generate (T), Number of Epochs

Steps:

For epoch 1 to Number of Epochs

For the batch of question-answer pairs, X and Y Do

1. Encoder: Perform question encoding, generate the hidden states for each timestep
2. Decoder:
   - 2.1 Generate tokens (with the highest probability) one by one by feeding weighted hidden states from Encoder until maximum answer length is reached, or end of sequence token is generated
   - 2.2 Join all tokens to generate an answer (Y')
3. Calculate the cross-entropy loss (the difference between Y and Y')
4. Update the model parameters

End For

End For

Output: Trained Seq2Seq Model

Algorithm 2 Seq2Seq Model Prediction (Beam Search)

Input: Trained Seq2Seq Model, Question (Q), Beam Size (k)

Steps:

1. Encoder: Perform question encoding, generate the hidden states for each timestep
2. Decoder:
   - 2.1 Generate tokens (with probability scores) by feeding weighted hidden states from Encoder
   - 2.2 Sort and get the k top tokens based on probability score *(this is the hypothesis with* 1 *word)*
   - 2.3 Add to beam search list
   - 2.4 For each of the hypothesis in the beam search list
     - 2.4.1 Generate tokens (with probability scores) by feeding weighted hidden states from Encoder
     - 2.4.2 Sort and get the k top tokens
     - 2.4.3 Add to the hypothesis in the beam search list
   - 2.5 Calculate probabilities for each of the k^2 hypothesis, sort and keep only k top hypothesis
   - 2.6 Repeat steps 2.4 to 2.5 until maximum answer length is reached or end of sequence token is generated
   - 2.7 Sort beam search list and return the hypothesis with the highest probability

Output: Answer to Question (Q)

the first token.

$$s_t = g_1(y_{t-1}, s_{t-1}, c_t) \tag{3}$$

$$p(y_t) = g(y_{t-1}, s_t, c_t) \tag{4}$$

3) The decoder then takes in its previous hidden state and the supplied ground-truth token to compute the next hidden state and the new conditional probability of target tokens to predict the next token. This step continues until the decoder generates a <EOS> token or reaches the maximum number of tokens to be generated.

The encoder and decoder are jointly trained (Algorithm 1) to maximize the conditional log probability of the correct output given the source sentence for the parameters (5). θ denotes model parameters, (x^n, y^n) is the n-th training pair of sentences, and T_n is the length of the n-th target sentence (y_n). Equation (5) is commonly simplified in literature as (6). The negative log-likelihood loss is given as (7).

$$\theta = \arg\max \sum_{n=1}^{N} \sum_{t=1}^{Tn} \log p(y_t^n | y_{t-1}^n, x^n) \tag{5}$$

$$\theta = \arg\max(\log p(Y|X)) \tag{6}$$

$$L_{LL} = -\frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{Tn} \log p(y_t^n | y_{<t}^n, x^n) \tag{7}$$

The negative loss likelihood is commonly implemented as a cross-entropy loss function as shown in (8).

$$L_{CE}\theta = -\sum_{t=1}^{T} \log p(y_t | y_1, y_2, \ldots, y_{t-1}, X) \tag{8}$$

The objective of the Seq2Seq model training is to maximize the probability to predict the correct token by minimizing prediction error cross-entropy loss. Algorithm 1 outlines the steps for the training. However, there is a slight difference between decoding during training and decoding during testing. Typically beam search decoding [13] is used during testing. As shown in Algorithm 2, k number of tokens will be predicted, where k can be from 1 to n where n is the vocabulary size. Then at each subsequent prediction, k tokens will be predicted for each of the tokens predicted earlier. Thus, in beam search decoding of size k, a total of k^2 will be predicted at each time step. However, to avoid performance bottleneck and processing complexity, only the top k hypothesis (incomplete sentence) based on its score is maintained and used for the next tokens prediction. The remaining hypothesis is ignored when the algorithm proceeds to the next time step.

Several tweaks can applied to optimize the performance of a Seq2eq model such as implementing different RNN network type which are LSTM or GRU, using a different attention mechanism which are global or local, trying different dropout rates, training batch sizes, beam search sizes, using alternative embedding types such as character or other n-grams in place of word embedding, tweaking embedding dimension and the number of hidden units and layers of the encoder and decoder [26], [27].

III. WEAKNESSES IN SEQ2SEQ MODEL

Although the Seq2Seq model performed reasonably well in NAG with those tweaks, some of the answers generated can be meaningless as the model tends to generate

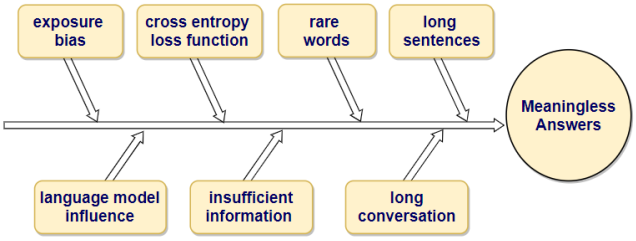

FIGURE 5. Factors causing Seq2Seq model to generate generic, meaningless and inconsistent answers.

generic or inconsistent answers. As shown in figure 5, these weaknesses are hypothesized to be caused by several factors such as:- the use of the cross-entropy loss function which encourages output generation with words which have high frequency in the training data [2], [28]–[31]; the use of teacher forcing technique during training which causes exposure bias [32]; insufficient information in the question to generate meaningful answer [30]; inability to deal with rare words such as named entity as decoder prediction is limited to the vocabulary that it was trained on [33]–[36]; language model influence in decoder compared to question information which results in answers are not relevant to the question [37]; difficulty to cope with long sentences (more than 20 words) or long conversations (conversation history) due to the need to compress all necessary information of a source sentence into a fixed-length vector [38]–[45]. The various solutions proposed by researchers to address these weaknesses will be reviewed in this paper.

In addition to quality issues highlighted above, recurrent neural network (RNN) based Seq2Seq models are slow to train. RNNs view input as a chain structure whereby the next output depends on the previous hidden state (sequential by design) which does not enable parallelization over elements of a sequence and thus require a linear number of $O(N)$ of operations where N is the sequence length [46], [47]. There are some proposals to replace RNN with Convolution Neural Network (CNN) such as by [48]–[50] or pure attention-based models like Transformer [47]. However, CNN based Seq2Seq models proposed were not for conversation modeling. Additionally, [1] performed an empirical study to compare the performance of Transformer with Seq2Seq and found that the Seq2Seq model (based on bidirectional LSTM encoder and unidirectional LSTM decoder) performed better than the Transformer model in terms of answer quality. Thus, this issue will not be elaborated in this review.

IV. SEQ2SEQ ENHANCEMENTS FOR NATURAL ANSWER GENERATION

There are several enhancements proposed by researches to address the weaknesses in Seq2Seq model-based natural answer generation. These enhancements can be broadly categorized into (A) structural modifications; (B) augmented learning; (C) beam search; and (D) complementary mechanisms.

A. STRUCTURAL MODIFICATIONS

Structural modifications refer to adding additional components to a standard Seq2Seq model. This allows for additional input to skew the model to generate a more meaningful or consistent answer. The following subsections provide a more detail explanation of how an answer can be skewed to meet a researcher's criteria.

1) ADDITIONAL EMBEDDINGS

Instead of passing only the question embedding to the encoder, an additional input embedding is passed to skew the encoding and decoding process to generate a more meaningful answer. As depicted in Figure 3, word embedding is a key component in a Seq2Seq model for NAG problems. Word embeddings are the distributed representations of words in a vector space used by learning algorithms to achieve better performance in natural language processing tasks by grouping similar words together [23]. Additional embedding can be introduced during decoding to skew the output to achieve a desirable outcome which means the answer is now conditioned to additional input.

For example, additional embedding called speaker embedding was proposed by [51] to generate a more consistent answer. The speaker embedding represents the persona (such as gender, location and age) of the person interacting with the chatbot. In this model, the speaker embedding, and word embedding were jointly trained, and it was reported by the authors that the answers generated were more consistent compared to a model without the additional embedding. There are several other proposals made by researchers to make the generated answer more meaningful[52]–[57] added emotion embedding. References [36], [58] added facts embedding. Reference [59] utilized topic embedding to ensure answers are related to the current topic. References [59] and [60] incorporated user profile embedding. In these scenarios, (3) and (4) are updated as (9) and (10) respectively where z represents the additional input and g_1 and g_2 is a non-linear function such as tanh, LSTM or GRU.

$$s_t = g_1(y_{t-1}, s_{t-1}, c_t, z) \tag{9}$$

$$p(y_t) = g_2(y_{t-1}, s_t, c_t, z) \tag{10}$$

2) ADDITIONAL ENCODERS

Adding additional parallel (figures 11 and 12) or hierarchical encoders (figures 6,7,8,9 and 10) were also proposed by several authors to provide additional input to the model and improve answer quality.

[2] proposed joint learning of input sentence and input topic to skew the output sentence generation to be relevant to the topic identified in the input sentence. They utilized another encoder (topic encoder) to generate topic embeddings. The decoder then uses the hidden states on this topic encoder to compute topics attention $\boldsymbol{o_t}$ for every timestep t. The original message attention ($\boldsymbol{c_t}$) and the newly introduced topic attention is then jointly learned during decoding. Reference [2] reported that their topic aware model can

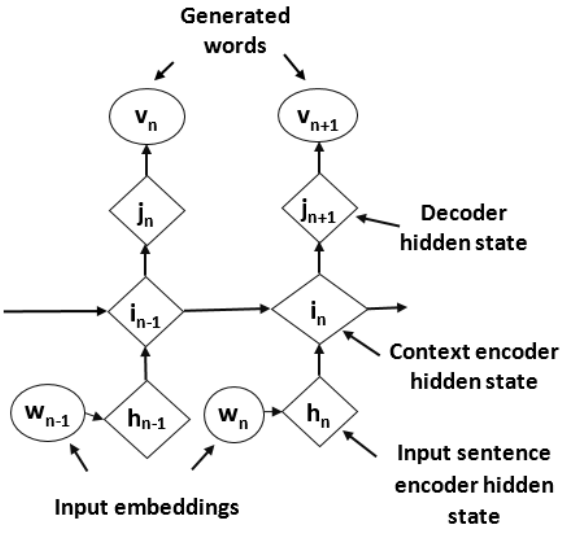

FIGURE 6. Hierarchical Encoder-Decoder (HRED). Generation of answer word is conditioned upon the decoder and context encoder hidden states.

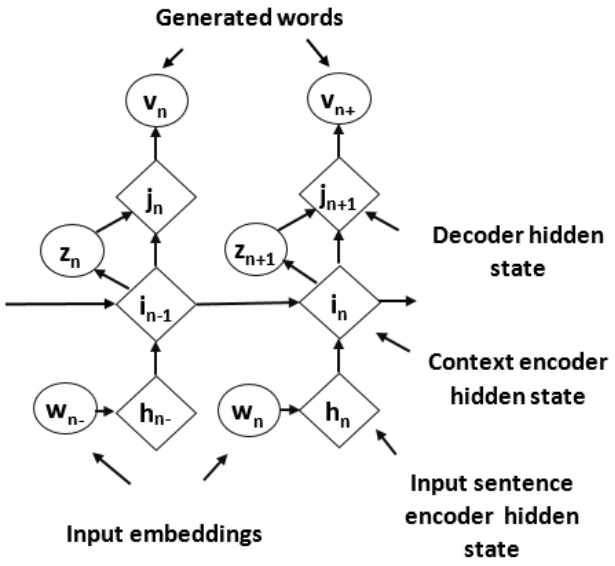

FIGURE 7. Variational (Latent Variable) Hierarchical Encoder-Decoder (VHRED). Generation of answer word is conditioned upon the decoder and context encoder hidden states and additionally a latent variable.

generate much more informative and interesting answers and less generic answers compared to the standard models. In this scenario, (3) is updated as (11) and g_1 is a non-linear function such as tanh, LSTM or GRU. For this scenario, the generation probability is (12), which is a combination of the conditional probability of sentence encoder and topic encoder.

$$s_t = g_1(y_{t-1}, s_{t-1}, c_t, o_t) \tag{11}$$

$$p(y_t) = p_{enc1}(y_t) + p_{enc2}(y_t) \tag{12}$$

[2]'s idea of using additional topics encoder was later adopted and enhanced in multi-task learning set up by [61] to generate a more topic coherent and diverse sentences.

[34], [35], [62], [63] proposed additional facts encoder to encode and inject facts retrieved based on keywords in the input sentence to the decoder.

Reference [64] used multiple encoders and a single decoder as part of an ensemble of models. One (1) encoder will encode the question as in a typical Seq2Seq model. The rest of the encoders (k in total) are used to encode k number of replies that are retrieved by a retrieval model for the query. The decoder takes in the encodings from all the $k+1$ encoders to generate an answer. In other words, the decoding process is conditioned to the query and an additional number of input sentences (replies from a retrieval model). On a slightly different approach, [61], [65] used multiple encoders, each trained to encode input of specific topics as proposed in [66]. During testing, an encoder selection module will choose the specific encoder based on the input topic.

Another approach proposed to improve the answer quality is by enabling the Seq2Seq model to have the ability to keep track of dialog history. This is achieved with hierarchical based Seq2Seq models. The original hierarchical model (HRED, Figure 6) was proposed by [67] for query suggestion. The first proposal to use HRED for answer generation was by [17] with a slight modification to the encoders. Their HRED model consists of two (2) bidirectional encoders and one (1) decoder. The first encoder (sentence encoder) encodes the current sentence/question. The last hidden state of the sentence encoder is then passed as input to the second encoder (context encoder). The context encoder updates its internal state based on the sentence encoder and its hidden states (which encodes the previous sentences/questions). This hidden state of the context encoder is then passed to the decoder to generate the answer. In a long conversation, the current context or topic of chatting can change to another context/topic. To equip the HRED model to handle this change, [42] proposed to use measure context-query relevance. In particular, they used cosine similarity to measure relevance. If the answer has high relevance, then the answer is conditioned to both the context and query. Otherwise, the answer is generated based on the query only.

Since the decoder has to summarize all the past information to generate the next word and due to the vanishing gradient effect, the HRED model faces difficulties to maintain the context and generate meaningful answers [41], [44], [45]. To alleviate this limitation, several authors proposed various improvements to the model. Reference [44] proposed Latent Variable Encoder-Decoder Model (VHRED, Figure 7) which is an improvement upon the original HRED architecture by introducing random latent variable (z) which is inferred from the context and sentence encoders and added to the decoder. The latent variable represents the current topic or sentiment and is conditioned on all the previous words and is trained by maximizing a variational lower bound on the log-likelihood. This latent variable is then concatenated with the context vector of the context encoder and passed to the decoder. In other words, the latent variable augments the HRED model at the context level to help the model to better handle and maintain the conversation state by adding one latent variable for each sentence.

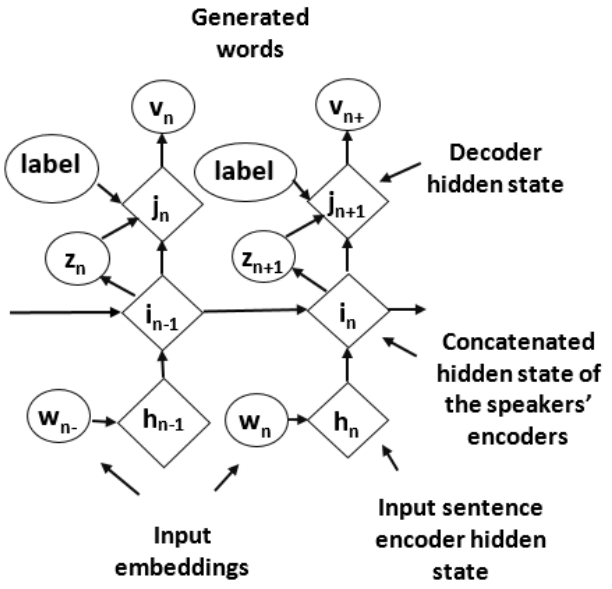

FIGURE 8. Separated Hierarchical Encoder-Decoder (SPHRED). Generation of answer word is conditioned upon the decoder and context encoder hidden states and additionally a latent variable and a label.

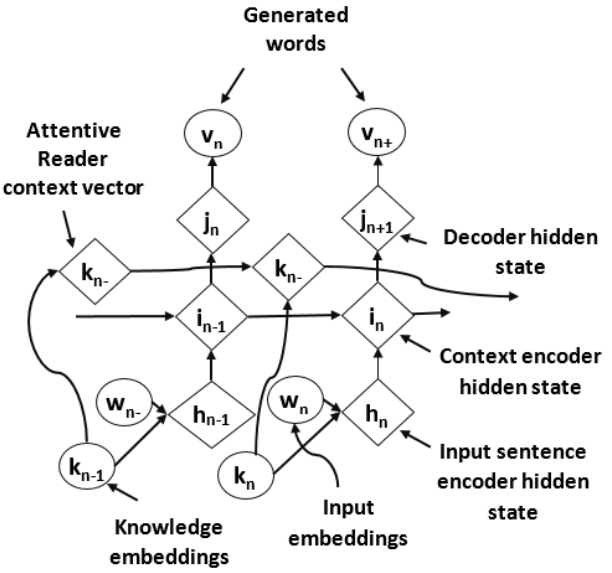

FIGURE 9. Attentive Reader: generation of answer word is conditioned upon the decoder, context encoder, and attentive reader hidden states.

Using a similar approach, [45] proposed Separated HRED (SPHRED, Figure 8) which conditions the answer based on the external labels in addition to the dialog context and a latent variable for each sentence. But in their model, the latent variable is not randomly generated but conditioned to a certain attribute (label). The label was assigned values that indicate whether an answer can be generic and whether the model should meet certain sentiment when generating an answer. Additionally, two (2) context encoders were applied; one for each speaker as compared to only one (1) in HRED and VHRED.

Reference [36] leveraged on HRED and introduced a third encoder called Attentive Reader (Figure 9) which encodes the

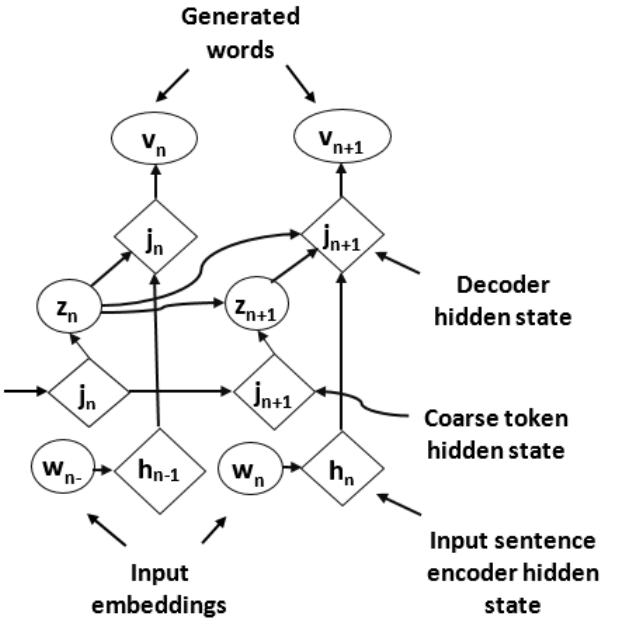

FIGURE 10. Multi-resolution recurrent neural networks (MrRNN). Generation of answer word is conditioned upon the decoder, coarse token and context encoder hidden states and additionally a latent variable.

previous dialog turns as well as knowledge embeddings gathered throughout the conversation to condition the decoder to generate a knowledgeable answer.

Reference [41] addressed the limitation of HRED by proposing another model called Multiresolution Recurrent Neural Networks (MrRNN, Figure 10) which conditions the answer generation to a sequence of latent variables instead of just one during each word generation. Instead of using a context encoder, MrRNN uses a coarse prediction encoder which generates the sequence of latent variables ($z_1, z_2, \ldots z_n$) which is conditioned only to the previously generated latent variables.

B. AUGMENTED LEARNING

A Seq2Seq model is trained using supervised learning whereby pairs of question-answer are fed into the model so that the model learns to predict the answer for a given question. It learns by repeating the process of predicting an answer for a given question, compares prediction with the gold standard, gets the difference/error using cross-entropy loss function, adjusts the weights of its neural networks to find the most probable answer by reducing the error. Unfortunately, the most probable answer may be a generic or inconsistent answer. To address this limitation, various proposals have been put forward by researchers to improve answer quality by replacing the cross-entropy loss function with an alternative loss function as reviewed in the following subsections.

1) ALTERNATIVE LOSS FUNCTION LEARNING

The standard loss function for Seq2Seq models which is the negative log-likelihood of output Y given input X, tends to result in generic answers. One method to address this issue is by augmenting the standard loss function with additional

terms to improve answer quality.

$$\theta = \arg\max((1-\lambda)\log p(Y|X) + \lambda \log p(X|Y)) \quad (13)$$

Reference [28] introduced Maximum Mutual Information (MMI) function (23) which skews towards those answers that are specific to the given input and avoids favoring answers that unconditionally enjoy high probability by penalizing the language model. This weighted MMI objective function represents a tradeoff between sources given targets and targets given sources with the λ controlling the influence of one another. However, this function was only used during testing and not training. It was also reported that this method which depends on an inverse model that is trained by swapping inputs and outputs may lead to ungrammatical outputs [28], [31]. Building upon the MMI idea, [53] proposed 'affective loss function' to support their 'affective embedding' to train their model to generate answers that are emotionally aligned by augmenting the standard loss function with an additional term that represents 'affective dissimilarity' between the input and output embeddings.

To discourage the generation of frequently occurring words, [68] proposed a frequency aware cross-entropy function (24) whereby they incorporated a weighting mechanism conditioned on the token frequency whereby frequent words will get lower weights and vice-versa. w_i is the weight for y_t which is calculated based on frequency in the training set.

$$L_{FACE}\theta = -\sum_{t=1}^{T} w_i \log p(y_t|y_1, y_2, \ldots, y_{t-1}, X) \quad (14)$$

As mentioned earlier in section III, there can be multiple answers that can be generated for a given question. With this in mind and the premise of not all answers are equal, [69] proposed a bag level loss function (25) instead of the individual question-answer pair loss function (17). Each question and all the relevant answers are grouped during training and each answer is scored according to how good is the answer to the question in a process called selective attention. N_β indicates the number of bags and L^k_{LLi} is the negative log-likelihood loss of the i-th instance in the k-th bag and α^k_i is the attention weight of the i-th answer's loss in this bag.

$$L = -\frac{1}{N\beta}\sum_{k=1}^{N\beta}\sum_i \alpha^k_i L^k_{LLi} \quad (15)$$

2) MULTI-TASK LEARNING (MTL)

Multi-task learning (MTL) is a machine learning approach where multiple tasks are learned to improve answer generation quality. MTL has shown success in many applications of machine learning including natural language processing [70], [71]. If MTL is performed sequentially, that is one task is learned after the other, it can be viewed as a form of inductive transfer which refers to the ability to improve the current task after having learned another related task.

Multi-task learning in the Seq2Seq model was first explored for machine translation problems [70], [72] and the success led other researchers to explore it for chatbot answer generation to address the issue of generic and inconsistent answer by Seq2Seq model.

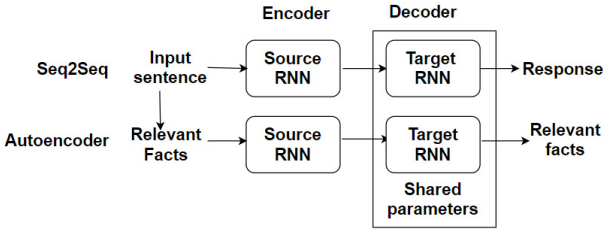

FIGURE 11. Multi-task learning framework for Seq2Seq with autoencoder (Figure adapted from [75].

Reference [73], [74] adopted sequential MTL whereby their Seq2Seq model is first trained with a dialog corpus to learn answer generation and then with an additional corpus to learn to generate a more consistent answer. Reference [73] leveraged on a style corpus (consisting of utterances of movie characters) to generate a more stylistically consistent answer while [74] used a manually curated personality corpus to generate an answer with personality.

Several authors proposed parallel MTL whereby in addition to the primary task of generating answers to a given question, a secondary task is trained simultaneously. In parallel MTL, the standard loss function is replaced with MTL loss function and usually takes the form as shown in (26) where L_{main} refers to loss of the main task, which is the answer generation, L_n refers to the loss of auxiliary tasks, and α_{main} *and* α_n refers to the weight given to each loss accordingly.

$$\alpha_{main}L_{main} + \alpha_n L_n \quad (16)$$

Reference [35] and [75] used very similar encoder-decoder modeling in their MTL approach (Figure 11) to make the answer more meaningful and factual. In their respective works, in addition to the standard Seq2Seq model, an additional autoencoder model was proposed. The function of the autoencoder is to retrieve additional facts from an external source based on keywords in the input sentence and inject them to the answer being generated. An autoencoder is a method of obtaining the original sequence embeddings based on the Seq2Seq framework. Instead of mapping source to target as in a typical Encoder-Decoder Seq2Seq model, the autoencoder predicts the input sequence itself [75]. The parameters of the decoder are shared across the two tasks. In other words, after training the Seq2Seq model to generate the answer, the autoencoder is trained to use the same decoder to make the answers more meaningful.

Using a different approach (Figure 12), [76] modified the standard Seq2Seq model by introducing multiple decoders to generate emotionally coherent answers. In their model, they defined six (6) emotion categories ('like', 'sadness', 'disgust', 'anger', 'happiness', 'other') and trained each decoder to decode answers for each emotional category. All the answers are then ranked to choose the final answer.

In [31], the secondary task trained is to classify the answers as positive or negative. Similarly, in[55], the secondary task was to learn an emotional detection classifier to detect the

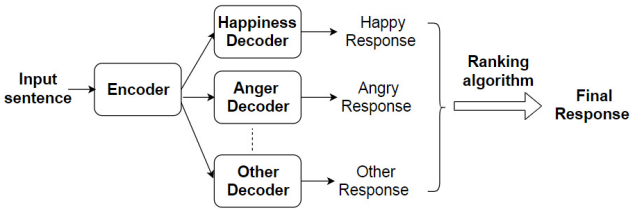

FIGURE 12. Multi-task learning framework for Seq2Seq using a single encoder and multiple decoders (Figure adapted from [76]).

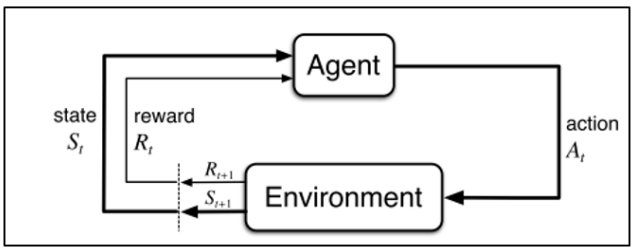

FIGURE 13. The agent-environment interaction in MDP. (Figure adapted from [78]).

emotion category based on the input sentence and then generate the emotion embedding accordingly to be passed to the decoder. Reference [61] proposed and trained an encoder selection model as their secondary task to choose a specific encoder based on topics identified in the input.

3) DEEP REINFORCEMENT LEARNING (DRL)

Reinforcement Learning (RL) is a machine learning framework for learning and continuously improving policies by a computer algorithm, the so-called agent, through interacting with its environment. As depicted in figure 13, in an Markov Decision Process (MDP), given a set of internal states $S = s1, s2 \ldots, sn$ and a set of predefined actions $A = a1, a2 \ldots, am$, the agent takes action a determined by current policy at state s and transition to a new state s', and receive a reward r from the environment. The agent aims to maximize some cumulative reward through a sequence of actions. Each such action forms a transition tuple (s, a, r, s,) of an MDP [77], [78]. Reinforcement learning algorithms that incorporate deep learning (implemented as a neural network) is known as deep reinforcement learning (DRL).

DRL has been studied and proposed by various authors to address the limitations of the Seq2Seq model and to further improve the quality of the natural answer generated. For this, the sequence generation is cast as the RL framework. The Seq2Seq model (the RNN or GRU or LSTM) is viewed as the agent interacting with the external environment (input word and context vector at every time step). The parameters of this agent define the policy which performs an action (predicting the next word in the sequence at every time step). After taking an action the agent updates its internal states (hidden units). Once the agent has reached the end of the sequence, it observes a reward. The loss function is usually defined as the negative expected reward of the full sequence. The reward can be any metrics like Bilingual Evaluation Understudy (BLEU) [79], Recall-Oriented Understudy for Gisting Evaluation (ROUGE) [80], METEOR [81] or developer-defined. Since arbitrary length sequences can be generated by the Seq2Seq model, the action space is infinite.

TABLE 1. Implementation of deep reinforcement learning by researchers.

Author	Reward Functions
[59]	Reconstruction, Language model, Topic coherence, BLEU
[84]	Ease of answering, information flow, and semantic coherence
[85]	Stance, Sentiment, and Stalemate

To ensure successful implementation of reinforcement learning for the Seq2Seq model especially NAG, an appropriate reward and reward function is needed. Thus, most of the research centers around computing an accurate reward function to feedback and optimize the Seq2Seq model policy. Commonly found methods to find the compute the rewards in Seq2Seq literature are policy gradient and actor-critic.

Policy Gradient: Policy gradient methods are an appealing approach specifically because they directly optimize the cumulative reward and can straightforwardly be used with nonlinear function approximators such as Seq2Seq. Policy gradients methods maximize the expected total reward by repeatedly estimating the gradient.

Starting from an arbitrary policy, the model proposes a sequence, and the ground-truth target or developer defined rewards or automatic metrics like BLEU are used to compute a reward for the proposed sequence. The reward is used as a weight for the log-likelihood of the proposed sequence. The policy is then continuously modified so that the agent will obtain the largest reward possible.

However, starting with an arbitrary random policy (which can be poor), can lead to a high branching factor and cause learning to be very difficult and very time-consuming. Typically, the Seq2Seq model will be first trained using the cross-entropy loss function as a warm start before policy gradient reinforcement learning kicks in [32], [82], [83].

References [59], [84], and [85] proposed offline learning-based policy gradient reinforcement learning for their dialogue generation model. These authors proposed several reinforcement learning reward functions to replace the standard loss function. The final reward is computed as the weighted sum of all the rewards. Table 1 summarises the reward functions proposed by the researchers.

Reference [78] on the other hand, used a dataset of input-output-reward triples to perform RL based training for their model. The output is the answer generated by the model given the input based on previous training and the reward is the score given by human evaluators which are then used to update the policy parameters. However, it is widely acknowledged that rewards that are manually defined can lead to suboptimal answer quality [86].

As opposed to offline learning, [87] proposed an interactive based reinforcement learning mechanism whereby for each answer generated by the model, user feedback is obtained and feedback to the model to update model param based on single question-answer pair. This method of updating model

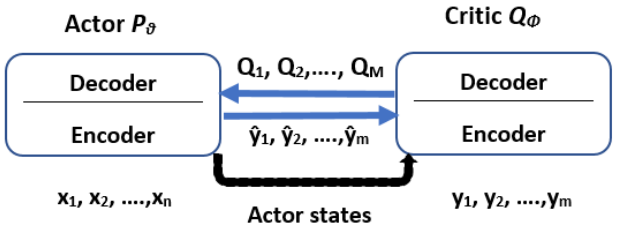

FIGURE 14. The actor-critic model.

parameters based on a single example is called one-shot learning [88].

Actor-Critic: The actor-critic model (Figure 14) is a more advanced type of RL that was proposed for Seq2Seq learning in the answer generation problems. An actor-critic model consists of the Critic component (*estimates the value function which is either action-value or state-value*), and Actor component (*updates the policy distribution in the direction suggested by the Critic*).

Reference [89] proposed actor-critic reinforcement learning for their dialogue sequence generation problem to address the exposure bias problem by replacing teacher forcing with test time metric (BLEU scores) during model training. In their approach, the actor and critic consist of encoder-decoder networks. The actor is a typical Seq2Seq model that generates an answer given an input sentence. It was trained in three (3) phases. In the initial phase, the actor is trained using standard log-likelihood. This is done because starting training with randomly initializing actor-critic will be problematic because the actor and critic will not be able to provide sufficient training signals to each other. Once actor training is done, its parameters are frozen, and the critic is trained by feeding it samples from the actor. Then in the third phase, both the actor-critic is trained but this time, instead of using the log-likelihood as the loss function, the Q values computed by the critic are used to approximate the gradient and train the actor to optimize its answer generation.

4) ADVERSARIAL LEARNING (AL)

Adversarial learning (AL) is another popular method proposed to further improve the Seq2Seq based answer generation quality. The idea for adversarial learning for the Seq2Seq model is borrowed from Generative Adversarial Network (GAN) proposed by [90]. GAN is a deep learning model based on game theory. It consists of two (2) neural networks:– generator (G) and discriminator (D). The idea behind the game theory is that the discriminator (D) will be fed with true data (from actual data) and false data (generated by generator (G) alternatively in batches so that it learns how to differentiate between the two types of data and eventually able to differentiate between true and false data. In conversation modeling, adversarial learning is used to provide feedback (in terms of adversarial loss) to the model to improve its answer generation quality.

Figure 15 shows the generalized framework for an AL implementation. The generator is a Seq2Seq network. But discriminator can be any neural network such as RNN

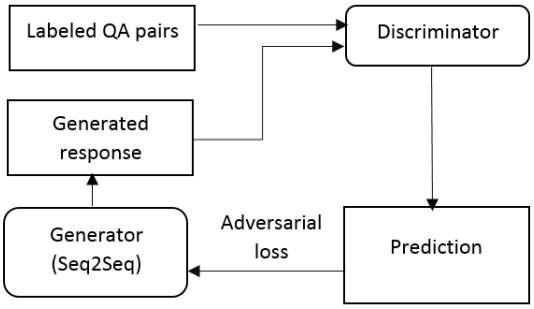

FIGURE 15. Adversarial learning framework for Seq2Seq.

TABLE 2. Implementation of adversarial learning by researchers.

Author	Discriminator Network	Adversarial Loss
[86]	Hierarchical LSTM	Probability of the input being human-generated, or machine-generated
[94]	Bidirectional LSTM	Probability of the input being true or false
[95]	Unidirectional LSTM	Cross-entropy loss of the discriminator
[93]	Deep Structured Similarity Model (DSSM) [96]	Cosine dissimilarity between generated sentence and target sentence embeddings
[97]	Convolution Neural Network (CNN)	Probability of the input being true or false
[98]	Seq2Seq model	Q values computed by the discriminator

or Convolution Neural Network (CNN) or even another Seq2Seq network. In AL, the discriminator network is trained first with labeled question-answer (QA) pair to make a binary prediction. In most cases, the generator will also be trained separately in parallel with discriminator for a certain number of epochs using the cross-entropy loss function as a warm start, similar to the approach taken for any other Seq2Seq based reinforcement learning. Once the discriminator training is done, its weights are fixed and the generator (Seq2Seq) network is now trained again. But this time, the answers generated by the Seq2Seq network are passed to discriminator to make predictions. The predictions are then fed back to the generator as adversarial loss which is used to approximate the gradient and update the generator via policy gradient [91], [92] to optimize its answer generation. The equilibrium between generator and discriminator is achieved when the generated data distribution matches actual data (labeled QA-pair) distribution [93].

There are differences in how the various researchers built and trained the discriminator network to make predictions and compute the adversarial loss. For example, [94] used a bidirectional LSTM discriminator and compute the probability of the input as true or false as the adversarial loss to be fed to the generator. Table 2 summarizes the approaches taken by the researchers surveyed in this review.

C. BEAM SEARCH

To address the issue of the language model influence,[37], [53] proposed decoding with the 'Diverse Beam Search' (DBS) algorithm as an improvement on the existing beam

search method. DBS incorporates diversity between the beams by maximizing an objective that consists of a standard sequence likelihood term and a dissimilarity metric between the beams (17). k is the beam size, λ is the weight to control how aggressive to promote diversity and Δ^k is the dissimilarity score.

$$p'(y_t) = \log p(y_{<t}|X) + \lambda\Delta^k) \tag{17}$$

In a slightly different approach, [29] proposed the 'Stochastic Beam Search' algorithm to encourage diversity in the answer. In their approach, instead of adding one word at a time for each time step, they add D number of candidate words to each beam and then compute the conditional log-probabilities of each of the beam and then perform the ranking to choose k top beams. This step is repeated until <EOS> token is generated, or maximum sequence length is reached.

There are different algorithms used to determine dissimilarities such as Hamming distance, n-grams comparison, cosine dissimilarly and affective dissimilarity [53].

D. COMPLEMENTARY MECHANISMS

In addition to the structural modifications and augmented learning, there are also proposals to further improve Seq2Seq2 model answer generation quality and will be discussed in this sub-section.

1) DECODER INPUT

One plausible reason for generating a low-quality answer in a typical Seq2Seq model is that the first word is predicted by the decoder is based start token in addition to context vector from the encoder. Adding the start token is necessary for Seq2Seq decoder to work but at the same time, it is not natural and learnable and doesn't have any association with the predicted first word. Reference [74] proposed models that train a separate network to learn to predict the first word during decoding. In their model, the decoder only starts predicting the second word onwards.

In another work, instead of letting the decoder to predict the first word based on encoded information as in a typical Seq2Seq model, [30] overwrite the process by using point-wise mutual information (PMI) to predict a keyword that is used the starting point to generate two (2) sub-sequences of the answer. The keyword is not assumed to be the first word of the answer and its location can be anywhere in the answer. There are two (2) pairs of encoder-decoder networks. As in the typical Seq2Seq model, both of the encoders encode the question. However, decoders behave differently. The decoder of the first pair (backward sequence decoder) generates the first sub-sequence (words in backward sequence) that is placed before the keyword. The decoder of the second pair (forward sequence decoder) generates the second sub-sequence that is placed after the keyword.

2) COPY MECHANISM

To address the inability to deal with rare words and named entities, [33] proposed copy mechanism, whereby the decoder is trained to either (i) **generate** the next word (as in typical seq2seq model) or (ii) **copy** certain words from the input sequence to be generated as part of the output sequence. To facilitate copying words from input sentences for output generation, an additional vector M, which is the hidden states of the input sentences, was introduced. Thus, the probability of generating any target word $\boldsymbol{y_t}$ is given by the mixture of probabilities (18). g and c stand for 'generate mode' and 'copy mode' respectively.

$$p(y_t) = p(y_t, g|s_t, y_{t-1}, c_t, M) + p(y_t, c|s_t, y_{t-1}, c_t, M) \tag{18}$$

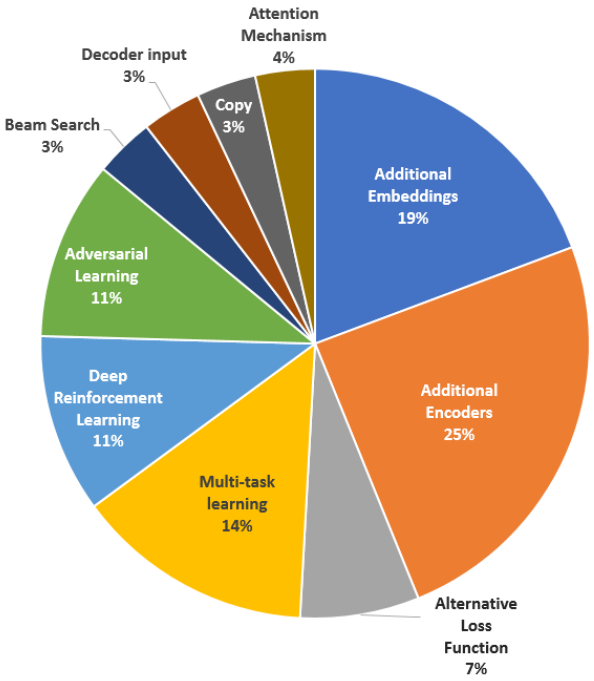

FIGURE 16. Summary of enhancement methods to improve the quality of sequence-to-sequence based natural answer generation models. Out of a total ten (10), the top three (3) popular methods are additional embeddings, additional encoders, and multi-task learning.

In the same line of research, [34] went one step further whereby the decoder is now trained to either (i) **generate** the next word (as in typical seq2seq model) or (ii) **copy** certain words from the input sequence to be generated as part of the output sequence or (iii) **copy** retrieved and matched facts from memory. To facilitate this, an additional deep neural network-based (DNN) knowledgebase (KB) encoder is introduced. This KB encoder is trained to retrieve facts from the knowledge base and score the matching between questions and facts.

3) ATTENTION MECHANISM

Leveraging on the global attention mechanism proposed by [39], [14] proposed a hybrid attention mechanism whereby the final hidden vector of the first encoder and global attention vector of the second encoder is concatenated and used for decoding. The global attention vector of the second encoder is computed by the decoder.

On another note, [99] proposed a slightly different attention mechanism whereby the attention mechanism is now

TABLE 3. Factors and proposed enhancements.

Enhancements / Factors	Structural Changes		Learning Strategies				BS	Additional Mechanisms		
	AE	AEn	ALF	MTL	DRL	ARL	BS	DI	CP	Att
Insufficient Information in Question	✓ [36] [51] - [59]	✓ [2] [34] [35] [41] [61] - [65]								
Long conversation		✓ [17] [36] [42] [44] [45]								
Rare Words									✓ [33] [34]	
Long Sentences										✓ [14] [99]
Cross-Entropy Loss Function			✓ [28] [53] [68] [69]	✓ [31] [35] [55] [61] [73] - [76]	✓ [59] [78] [84] [85] [87] [89]	✓ [86] [93] [94] [95] [97] [98]				
Exposure Bias					✓ [89]			✓ [30]		
Language Model Influence							✓ [29] [53]	✓ [74]		

AE	Additional embeddings
AEn	Additional Encoders
ALF	Alternate loss function
MTL	Multi-task learning
DRL	Deep reinforcement learning
ARL	Adversarial reinforcement learning
BS	Beam search
DI	Decoder Input
CP	Copy Mechanism
Att	Attention mechanism

divided into two (2). First, during encoding, the encoder projects K number of context vectors. Then, during decoding, the decoder scores each of the context vectors and concatenates them using the score as weight and eventually use the final vector for predicting the next word.

V. SUMMARY AND CONCLUSION

As shown in Figure 16, there are a total of ten (10) methods proposed to address the weakness and further enhance the Seq2Seq model for the natural answer generation problem. Table 3 summarises the factors contributing to the weakness and various enhancements proposed by researchers to address those factors. It should be noted that several authors deployed a combination of methods to further enhance the Seq2Seq model Although the enhancements are making the model complicated, it is deemed necessary to address the structural weakness of the Seq2Seq model which results in meaningless answers. These enhancements provide support for the Seq2Seq model during training and prediction to generate meaningful answers.

The top three (3) of the popular enhancements methods that were found during this literature study are additional encoders, additional embeddings, and multi-task learning. Additional encoders and embeddings enable additional input to be provided to model and be very effective to skew the model output to a desired quality and outcome. Using a student learning process analogy, this is akin to referring to a dictionary or thesaurus when studying so that they better understanding the meaning of certain terms. On the other hand, multi-task learning improves performance by enabling the model to learn more than one task either in sequence or simultaneously, which contributes to improvement in the answer generation task. Using the same student learning process analogy, this is akin to attending additional tuition classes. In some cases, multi-task learning was done in conjunction with additional encoders or embeddings.

This review is an attempt to provide readers with a comprehensive literature review on the weaknesses and issues observed when modeling natural answer generation using standard Seq2Seq model and the state-of-the-art enhancements proposed to address and resolve those weaknesses and issues. Readers and potential researchers planning to model their chatbots can refer to this review to get an understanding of how they can further improve their model and reduce the risk of duplicating the work.

This review focused on the issues and enhancements to the Seq2Seq model itself. Potential extensions include performing a comprehensive literature review on the training data and evaluation metrics used by various researchers for their natural answer generation setting.

APPENDIX

See Figure 16.

ACKNOWLEDGMENT

The authors would like to thank AOARD for the support and express our appreciation to all who have contributed to this research.

REFERENCES

[1] M. Hardalov, I. Koychev, and P. Nakov, "Towards automated customer support," in *Proc. Int. Conf. Artif. Intell. Methodol. Syst. Appl.*, 2018, pp. 48–59.

[2] C. Xing, W. Wu, Y. Wu, J. Liu, Y. Huang, M. Zhou, and W.-Y. Ma, "Topic aware neural response generation," in *Proc. 21st AAAI Conf. Artif. Intell.*, vol. 17, 2017, pp. 3351–3357.

[3] J.-P. Sansonnet, D. Leray, and J.-C. Martin, "Architecture of a framework for generic assisting conversational agents," *Intelligent Virtual Agents* (Lecture Notes in Computer Science). Berlin, Germany: Springer, 2006, pp. 145–156.

[4] A. M. Turing, "Computing machinery and intelligence," in *Parsing the Turing Test*, vol. 59. Dordrecht, The Netherlands: Springer, Oct. 1950, pp. 433–460.

[5] J. Weizenbaum, "ELIZA—A computer program for the study of natural language communication between man and machine," *Commun. ACM*, vol. 26, no. 1, pp. 23–28, Jan. 1983.

[6] V. Sharma, M. Goyal, and D. Malik, "An intelligent behaviour shown by chatbot system," *Int. J. New Technol. Res.*, vol. 3, no. 4, pp. 52–54, 2017.
[7] H.-Y. Shum, X.-D. He, and D. Li, "From eliza to XiaoIce: Challenges and opportunities with social chatbots," *Frontiers Inf. Technol. Electron. Eng.*, vol. 19, no. 1, pp. 10–26, Jan. 2018.
[8] J. Walker, "Chatbot comparison—Facebook, Microsoft, Amazon, and Google," in *Techemergence*. Emerj Artificial Intelligence Research, 2018.
[9] M. Zhang, N. Yu, and G. Fu, "A simple and effective neural model for joint word segmentation and POS tagging," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 26, no. 9, pp. 1528–1538, Sep. 2018.
[10] Y. Hao, Y. Zhang, K. Liu, S. He, Z. Liu, H. Wu, and J. Zhao, "An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 221–231.
[11] Z. Ye, R. Cai, Z. Liao, Z. Hao, and J. Li, "Generating natural answers on knowledge bases and text by sequence-to-sequence learning," in *Proc. ICANN*, 2018, pp. 447–455.
[12] O. Dušek, J. Novikova, and V. Rieser, "Evaluating the state-of-the-art of End-to-End natural language generation: The E2E NLG challenge," *Comput. Speech Lang.*, vol. 59, pp. 123–156, Jan. 2019.
[13] O. Vinyals and Q. Le, "A neural conversational model," in *Proc. 31st Int. Conf. Mach. Learn.*, Lille, France, vol. 37, 2015. [Online]. Available: https://sites.google.com/site/deeplearning2015/accepted-papers
[14] L. Shang, Z. Lu, and H. Li, "Neural responding machine for short-text conversation," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, vol. 1, 2015, pp. 1577–1586.
[15] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1724–1734.
[16] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.
[17] I. Serban, A. Sordoni, Y. Bengio, A. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 3776–3783.
[18] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *J. Artif. Intell. Res.*, vol. 61, pp. 65–170, Jan. 2018.
[19] S. Arsovski, S. Hui, and A. David, "Open-domain neural conversational agents: The step towards artificial general intelligence," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 6, 2018.
[20] E. Merdivan, D. Singh, S. Hanke, and A. Holzinger, "Dialogue systems for intelligent human computer interactions," *Electron. Notes Theor. Comput. Sci.*, vol. 343, pp. 57–71, May 2019.
[21] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [Review Article]," *IEEE Comput. Intell. Mag.*, vol. 13, no. 3, pp. 55–75, Aug. 2018.
[22] H. Chen, X. Liu, D. Yin, and J. Tang, "A survey on dialogue systems: Recent advances and new frontiers," *ACM SIGKDD Explor. Newslett.*, vol. 19, no. 2, pp. 25–35, Nov. 2017.
[23] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed-representations-of-words-and-phrases-and-their-compositionality," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 3111–3119.
[24] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1532–1543.
[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
[26] D. Britz, A. Goldie, M.-T. Luong, and Q. Le, "Massive exploration of neural machine translation architectures," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 1442–1451.
[27] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, pp. 1929–1958, Jan. 2014.
[28] J. Li, M. Galley, C. Brockett, J. Gao, and B. Dolan, "A diversity-promoting objective function for neural conversation models," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2016, pp. 110–119.
[29] Y. Shao, S. Gouws, D. Britz, A. Goldie, B. Strope, and R. Kurzweil, "Generating high-quality and informative conversation responses with Sequence-to-Sequence models," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2210–2219.
[30] L. Mou, Y. Song, R. Yan, G. Li, L. Zhang, and Z. Jin, "Sequence to backward and forward sequences: A content-introducing approach to generative short-text conversation," in *Proc. COLING*, 2016, pp. 3349–3358.
[31] Y. Huang and T. Zhong, "Multitask learning for neural generative question answering," *Mach. Vis. Appl.*, vol. 29, no. 6, pp. 1009–1017, Feb. 2018.
[32] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, "Sequence level training with recurrent neural networks," in *Proc. ICLR*, 2016, pp. 1–16.
[33] J. Gu, Z. Lu, H. Li, and V. O. K. Li, "Incorporating copying mechanism in sequence-to-sequence learning," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2016, pp. 1631–1640.
[34] S. He, C. Liu, K. Liu, and J. Zhao, "Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 1, 2017, pp. 199–208.
[35] M. Ghazvininejad, C. Brockett, M. W. Chang, B. Dolan, J. Gao, W.-T. Yih, and M. Galley, "A knowledge-grounded neural conversation model," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5110–5117.
[36] Y. Wang, W. Rong, Y. Ouyang, and Z. Xiong, "Augmenting dialogue response generation with unstructured textual knowledge," *IEEE Access*, vol. 7, pp. 34954–34963, 2019.
[37] A. K. Vijayakumar, "Diverse beam search for improved description of complex scenes," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 7371–7379.
[38] J. Pouget-Abadie, D. Bahdanau, B. van Merrienboer, K. Cho, and Y. Bengio, "Overcoming the curse of sentence length for neural machine translation using automatic segmentation," in *Proc. 8th Workshop Syntax, Semantics Struct. Stat. Transl. (SSST)*, 2014, pp. 78–85.
[39] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. ICLR*, 2015, pp. 1–15.
[40] J. Li and D. Jurafsky, "Neural net models for open-domain discourse coherence," in *Proc. Conf. Empir. Methods Nat. Lang. Process.*, 2017, pp. 198–209.
[41] I. Serban, T. Klinger, G. Tesauro, Y. Bengio, and A. Courville, "Multiresolution recurrent neural networks: An application to dialogue response generation," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 3288–3294.
[42] Z. Tian, R. Yan, L. Mou, Y. Song, Y. Feng, and D. Zhao, "How to make context more useful? An empirical study on context-aware neural conversational models," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 2, 2017, pp. 231–236.
[43] Y. Park, J. Cho, and G. Kim, "A hierarchical latent structure for variational conversation modeling," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, 2018, pp. 1792–1801.
[44] I. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. Courville, and Y. Bengio, "A hierarchical latent variable encoder-decoder model for generating dialogues," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 3295–3301.
[45] X. Shen, H. Su, Y. Li, W. Li, S. Niu, Y. Zhao, A. Aizawa, and G. Long, "A conditional variational framework for dialog generation," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, vol. 2, pp. 504–509, 2017.
[46] Y. N. Dauphin, A. Fan, M. Auli, and D. Grangier, "Language modeling with gated convolutional networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, Aug. 2017, pp. 933–941.
[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 5998–6008.
[48] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proc. 34th Int. Conf. Mach. Learn. (JMLR)*, vol. 70, 2017, pp. 1243–1252.
[49] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, and K. Kavukcuoglu, "Neural machine translation in linear time," 2016, *arXiv:1610.10099*. [Online]. Available: https://arxiv.org/abs/1610.10099
[50] S. Chen, C. Li, F. Ji, W. Zhou, and H. Chen, "Review-driven answer generation for product-related questions in E-commerce," in *Proc. 12th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2019, pp. 411–419.
[51] J. Li, M. Galley, C. Brockett, G. Spithourakis, J. Gao, and B. Dolan, "A persona-based neural conversation model," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, 2016, pp. 994–1003.

[52] H. Zhou, M. Huang, T. Zhang, X. Zhu, and B. Liu, "Emotional chatting machine: Emotional conversation generation with internal and external memory," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 730–738.

[53] N. Asghar, P. Poupart, J. Hoey, X. Jiang, and L. Mou, "Affective neural response generation," in *Proc. Eur. Conf. Inf. Retr.*, 2018, pp. 154–166.

[54] C. Huang, O. Zaiane, A. Trabelsi, and N. Dziri, "Automatic dialogue generation with expressed emotions," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2018, pp. 49–54.

[55] F. Liu, Q. Mao, L. Wang, N. Ruwa, J. Gou, and Y. Zhan, "An emotion-based responding model for natural language conversation," *World Wide Web*, vol. 22, no. 2, pp. 843–861, Jun. 2019.

[56] Y. Peng, Y. Fang, Z. Xie, and G. Zhou, "Topic-enhanced emotional conversation generation with attention mechanism," *Knowl.-Based Syst.*, vol. 163, pp. 429–437, Jan. 2019.

[57] T. Dryjanski, P. Bujnowski, H. Choi, K. Podlaska, K. Michalski, K. Beksa, and P. Kubik, "Affective natural language generation by phrase insertion," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 4876–4882.

[58] B. Pan, Y. Yang, H. Li, Z. Zhao, Y. Zhuang, and D. Cai, "MacNet: Transferring knowledge from machine comprehension to sequence-to-sequence models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 6092–6102.

[59] M. Yang, W. Tu, Q. Qu, Z. Zhao, X. Chen, and J. Zhu, "Personalized response generation by dual-learning based domain adaptation," *Neural Netw.*, vol. 103, pp. 72–82, Jul. 2018.

[60] J. Wang, X. Wang, Z. Xu, Z. Wang, and B. Wang, "Group linguistic bias aware neural response generation," in *Proc. 9th SIGHAN Workshop Chin. Lang. Process.*, 2017, pp. 1–10.

[61] C. Gao and J. Ren, "A topic-driven language model for learning to generate diverse sentences," *Neurocomputing*, vol. 333, pp. 374–380, Mar. 2019.

[62] Y. Long, J. Wang, Z. Xu, Z. Wang, B. Wang, and Z. Wang, "A knowledge enhanced generative conversational service agent," in *Proc. 6th Dialog Syst. Technol. Challenges Workshop*, May 2018, pp. 1–5.

[63] Z. Wang, Z. Wang, Y. Long, J. Wang, Z. Xu, and B. Wang, "Enhancing generative conversational service agents with dialog history and external knowledge," *Comput. Speech Lang.*, vol. 54, pp. 71–85, Mar. 2019.

[64] Y. Song, C.-T. Li, J.-Y. Nie, M. Zhang, D. Zhao, and R. Yan, "An ensemble of retrieval-based and generation-based human-computer conversation systems," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4382–4388.

[65] D. Ren, Y. Cai, X. Lei, J. Xu, Q. Li, and H.-F. Leung, "A multi-encoder neural conversation model," *Neurocomputing*, vol. 358, pp. 344–354, Sep. 2019.

[66] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[67] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie, "A hierarchical recurrent encoder-decoder for generative context-aware query suggestion," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage. (CIKM)*, 2015, pp. 553–562.

[68] S. Jiang, P. Ren, C. Monz, and M. de Rijke, "Improving neural response diversity with frequency-aware cross-entropy loss," in *Proc. World Wide Web Conf. (WWW)*, vol. 2, 2019, pp. 2879–2885.

[69] M. Wei and Y. Zhang, "Natural answer generation with attention over instances," *IEEE Access*, vol. 7, pp. 61008–61017, 2019.

[70] M.-T. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," in *Proc. ICLR*, 2016, pp. 1–10.

[71] R. Caruana, "Multitask learning," *Mach. Learn.*, vol. 28, pp. 41–75, Jul. 1997.

[72] D. Dong, H. Wu, W. He, D. Yu, and H. Wang, "Multi-task learning for multiple language translation," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 1723–1732.

[73] R. Akama, K. Inada, N. Inoue, S. Kobayashi, and K. Inui, "Generating stylistically consistent dialog responses with transfer learning," in *Proc. 8th Int. Jt. Conf. Nat. Lang. Process.*, vol. 2, 2017, pp. 408–412.

[74] W.-N. Zhang, Q. Zhu, Y. Wang, Y. Zhao, and T. Liu, "Neural personalized response generation as domain adaptation," *World Wide Web*, vol. 22, no. 4, pp. 1427–1446, Jun. 2018.

[75] Y. Luan, C. Brockett, B. Dolan, J. Gao, and M. Galley, "Multi-task learning for speaker-role adaptation in neural conversation models," in *Proc. 8th Int. Joint Conf. Nat. Lang. Process.*, vol. 1, 2017, pp. 605–614.

[76] R. Zhang, Z. Wang, and D. Mai, "Building emotional conversation systems using multi-task Seq2Seq learning," in *Proc. Nat. CCF Conf. Nat. Lang. Process. Chin. Comput.*, vol. 1, 2017, pp. 612–621.

[77] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2017.

[78] K. Kandasamy, Y. Bachrach, R. Tomioka, D. Tarlow, and D. Carter, "Batch policy gradient methods for improving neural conversation models," in *Proc. ICLR*, 2017, pp. 1–14.

[79] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BLEU: A method for automatic evaluation of machine translation," in *Proc. 40th Annu. Meet. Assoc. Comput. Linguist.*, Jul. 2002, pp. 311–318.

[80] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Proc. Text Summ. Branches ACL Workshop*, 2004, pp. 74–81.

[81] A. Lavie and A. Agarwal, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proc. ACL Workshop Intrinsic Extrinsic Eval. Meas. Mach. Transl. Summarization*, Jun. 2005, pp. 65–72.

[82] F. Agostinelli, G. Hocquet, S. Singh, and P. Baldi, "From reinforcement learning to deep reinforcement learning: An overview," in *Braverman Readings in Machine Learning. Key Ideas From Inception to Current State* (Lecture Notes in Computer Science), vol. 11100. Cham, Switzerland: Springer, 2018, pp. 298–328.

[83] N. Ding and R. Soricut, "Cold-start reinforcement learning with softmax policy gradient," in *Proc. 31st Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 2817–2826.

[84] J. Li, W. Monroe, A. Ritter, D. Jurafsky, M. Galley, and J. Gao, "Deep reinforcement learning for dialogue generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1192–1202.

[85] W. Zhang, L. Li, and D. Cao, "Exploring implicit feedback for open domain conversation generation," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 547–554.

[86] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial learning for neural dialogue generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2157–2169.

[87] N. Asghar, P. Poupart, X. Jiang, and H. Li, "Deep active learning for dialogue generation," in *Proc. 6th Joint Conf. Lexical Comput. Semantics (SEM)*, 2017, pp. 1–6.

[88] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.

[89] D. Bahdanau, "An actor-critic algorithm for sequence prediction," in *Proc. ICLR*, 2017, pp. 1–17.

[90] I. Goodfellow, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. Courville, and Y. Bengio, "Generative adversarial networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 1–9.

[91] R. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.

[92] D. Silver, "Lecture 7: Policy gradient," Univ. College London, London, U.K., Tech. Rep., 2015. [Online]. Available: http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching_files/pg.pdf

[93] Y. Zhang, M. Galley, J. Gao, Z. Gan, X. Li, and C. Brockett, "Generating informative and diverse conversational responses via adversarial information maximization," in *Proc. 32nd Conf. Neural Inf. Process. Syst.*, 2018, pp. 1810–1820.

[94] E. Bruni and R. Fernandez, "Adversarial evaluation for open-domain dialogue generation," in *Proc. 18th Annu. SIGDIAL Meeting Discourse Dialogue*, Aug. 2018, pp. 284–288.

[95] J. Xu, X. Ren, J. Lin, and X. Sun, "Diversity-promoting GAN: A cross-entropy based generative adversarial network for diversified text generation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 3940–3949.

[96] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for Web search using clickthrough data," in *Proc. 22nd ACM Int. Conf. Conf. Inf. Knowl. Manage. (CIKM)*, vol. 13, 2013, pp. 2333–2338.

[97] Z. Xu, B. Liu, B. Wang, C. Sun, X. Wang, Z. Wang, and C. Qi, "Neural response generation via GAN with an approximate embedding layer," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 617–626.

[98] Y.-L. Tuan and H.-Y. Lee, "Improving conditional sequence generative adversarial networks by stepwise evaluation," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 27, no. 4, pp. 788–798, Apr. 2019.

[99] C. Tao, S. Gao, M. Shang, W. Wu, D. Zhao, and R. Yan, "Get the point of my utterance! Learning towards effective responses with multi-head attention mechanism," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 4418–4424.

KULOTHUNKAN PALASUNDRAM received the B.S degree in computer science (Hons.) and the master's degree in information technology from Universiti Kebangsaan Malaysia, in 1995 and 1998, respectively. He is currently pursuing the Ph.D. degree in intelligent computing with Universiti Putra Malaysia. His research interests include artificial intelligence, deep learning, big data, natural language processing, and dialogue generation.

NURFADHLINA MOHD SHAREF is an Associate Professor with the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Malaysia. Besides chatbot, her current projects are multi-objective deep reinforcement learning, multi-task deep learning for multi-class tweets classification, and deep tensor factorization model for recommendation systems. Her research interests revolve around text mining, recommendation systems, and semantic Web.

KHAIRUL AZHAR KASMIRAN received the Ph.D. degree from the University of Sydney, Australia, in 2012. He is a Senior Lecturer with the Department of Computer Science, Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Malaysia. His interests include deep learning, reinforcement learning, performance engineering, formal verification, and software development

AZREEN AZMAN received the bachelor's degree in information technology majoring in information systems engineering from Multimedia University, Malaysia, in 1999, the Diploma degree in software engineering from the Institute of Telecommunication and Information Technology, in 1997, and the Ph.D. degree in computer science specializing in information retrieval with the University of Glasgow, Scotland, in 2007. He was accepted directly to second year with Multimedia University. After serving in the industry for a few years, he enrolled for the Ph.D. degree in January 2003. He is currently an Associate Professor with the Universiti Putra Malaysia. His current research interests include information retrieval, text mining, natural language processing, and intelligent systems. He serves as a Committee Member of the Malaysian Society of Information Retrieval and Knowledge Management (PECAMP) and the Malaysian Information Technology Society (MITS).