Survey on Augmented Sequence to Sequence Models for Generative Dialog Systems

Suriyadeepan Ramamoorthy

[1]VOICEMONK
Hyderabad, India

suriyadeepan.r@gmail.com

1. Introduction

Despite it's simple design, the sequence to sequence model, performs surprising well on sequence to sequence mapping tasks, like Machine Translation, Text to speech, etc,. One of it's interesting application is dialog systems or virtual agents or chatbots or conversational agents. Many researchers have explored the possibility of building an automated data-driven dialog system using this model. Although, they were able to produce some interesting results, which showed tremendous potential, the dialog systems built with sequence to sequence models, lack severely in several sub-tasks of conversational modeling. Recent research in this front, introduces multiple variants of the vanilla sequence to sequence architecture, which address the mentioned problems. This work presents an exploration of such variants of sequence to sequence architecture and suggests the use of explicit memory, to improve the performance of conversational models.

2. Sequence to Sequence Architecture

The Sequence to Sequence model was introduced in [Cho et al. 2014] for Statistical Machine Translation. In Machine Translation task, source sentence in one language is translated into a target sentence in another language. A generic RNN can map one sentence to another, given the alignment between the source and the target sentence, is known apriori. This isn't the case in Machine Translation. Typically the length of source and target sequences vary from one sample to another. The seq2seq model handles variable length sequences, by mapping a variable length source sequence to a fixed length vector and then, mapping the vector to the variable length target sequence.

The seq2seq model consists of two RNNs : an encoder and a decoder. The encoder takes a sequence as input and processes one symbol at a time. The final state of the encoder is considered the **context**. It represents the task relevant summary of the input sequence. The context is set as the initial state of the decoder. The decoder, at each time step, is conditioned on the context and the previously generated symbols. In other words, the decoder performs language modeling to generate the target sequence, conditioned on an input sequence.

In language modeling, the output at each timestep t is given by the conditional distribution over all the symbols, w_j in the vocabulary, $j = 1, ...K$. The probability of the sequence $\mathbf{x}$ can be computed by combining individual probabilities.

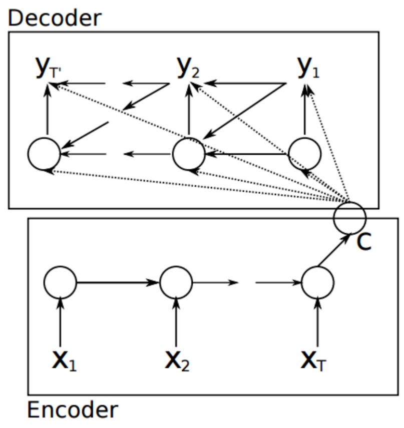

Figure 1. Illustration of RNN Encoder-Decoder from [Cho et al. 2014]

$$p(x_t|x_{t-1},...x_1) = \frac{exp(score(x_t))}{\sum_{j=1}^{K} exp(score(w_j))} \tag{1}$$

$$p(x) = \prod_{t=1}^{T} p(x_t|x_{t-1},....x_1) \tag{2}$$

The conditional distribution of target sequence, conditioned on source sequence is reduced to the probability distribution of the target sequence conditioned on the context. The conditional probability of the next symbol in the decoder is a function (g) of the previous symbols and the context.

$$p(y_t|y_{t-1},...y_1,\mathbf{c}) = g(h_t, y_{t-1}, \mathbf{c}) \tag{3}$$

$$p(y|x) = \prod_{t=1}^{T} p(y_t|y_{t-1},...y_1,\mathbf{c}) \tag{4}$$

The encoder and decoder are jointly trained to maximize the conditional log-likelihood,

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^{N} logp_\theta(y_n|x_n) \tag{5}$$

[Vinyals and Le 2015] uses sequence to sequence model for building a Neural Conversational Model.

3. Limitations and Challenges

3.1. Context Representation

The context produced by the encoder is the major bottleneck in sequence to sequence models. The source sentences are typically variable length sequences. If a model faces a source sentence that is longer than any of the sentences it has encountered during the training, it fails to effectively summarize the sequence, leading to poor translation. The function of the encoder as be seen as, compressing a variable length sequence into a vector, and the decoder can be seen as uncompressing the compressed vector. Naturally, when the source sequence becomes longer, the compression becomes lossy, leading to poor performance in translation.

3.2. External Context

It may be helpful for the model to take into account, the context of the conversation, apart from the words uttered. The nature of the conversation is heavily influenced by this external context. It may include the properties of the speaker and that of the listener, the number of listeners, the environment, etc,.

3.3. Intent

The objective of communication, from an individual's (speaker) point of view, is to achieve a particular goal, by influencing the listener(s). Modeling the intention of the speaker, is essential to adapt and mold the conversation, to express the user's goals explicitly or implicitly.

3.4. Personality

Apart from understanding and responding to natural language queries, an agent is typically given a personality. This is true for both open-ended system and for goal-oriented systems. Providing a coherent personality to an agent, is not a trivial task, especially if the system is trained end-to-end, based on training data. It is desirable for an agent to evolve and maintain a personality, based on an initial list of personality traits. Such a system would need to assume a personality, based on an external knowledge graph.

4. Enhanced Models

4.1. Align and Translate

The use of fixed length context is a bottleneck in improving the performance of seq2seq architecture. In [Bahdanau et al. 2014], the context is extended, by allowing the model to search for parts of the source sequence relevant to predicting a target word. The search figures out the alignment between a target word and words from the source sequence. The source sequence is encoded into a sequence of vectors and the decoder is allowed to choose a subset of these vectors adaptively.

A bidirectional RNN is used as encoder, which consists of two RNNs processing the input sequence from either directions. At each time step t, it consists of two internal states, h_i and g_i. h_i captures the flow of information from left to right $x_1, x_2, ...x_i$, while g_i captures the information from right to left $x_{i+1}, ...x_n$. Hence, at each time step, the forward ($\overrightarrow{h_j}$) and the backward ($\overleftarrow{h_j}$) states are concatenated to $h_j = [\overrightarrow{h_j}^T ; \overleftarrow{h_j}^T]^T$ contains the summary of the sequence from either directions.

While predicting the target words, the decoder chooses a context vector c_j for each word w_j. The context vector c_j, is a weighted sum of encoder states.

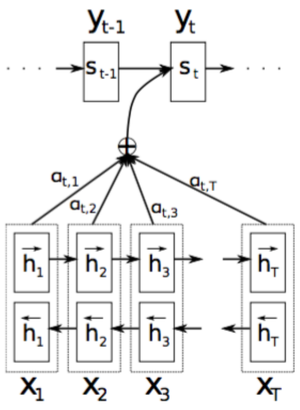

Figure 2. Attention mechanism

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j \tag{6}$$

The weights α_{ij} of encoder state vectors h_j, is given by,

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T_x} exp(e_{ik})} \tag{7}$$

The alignment model e_{ij} scores how well inputs around position j match with output at position i. The alignment model parameterized by a is learned jointly with the rest of the network.

$$e_{ij} = a(s_{i-1}, h_j) \tag{8}$$

4.2. Attention with Intention

The model proposed in [Yao et al. 2015], consists of an encoder, decoder and an intention network. The intention network model the dynamics of the intention process. Conversations in general, consist of two levels of structures. In the lower level, there is attention, which plays a role in processing of utterance (ie.) which words to pay attention to. In the higher level, the intention regulates the discourse structure and coherence. The seq2seq model which performs well on tasks like, Machine Translation, Grapheme-to-phoneme mapping, Named Entity Tagging. The dialog process involves multiple turns.

The intention network is defined as,

$$h^{(i,k)} = f\left(c_T^{(s)}, h^{(i,k-1)}, h_T^{(t,k-1)}\right) \tag{9}$$

where $c_T^{(s)}$ is the context created by the encoder, k represents the index of current turn, $h_T^{(t,k-1)}$ is the final hidden state of decoder, during the previous turn of conversation $k-1$.

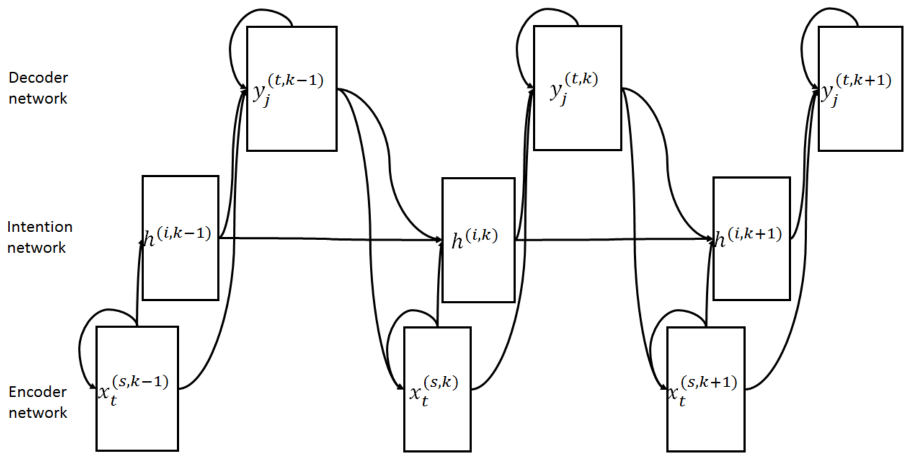

Figure 3. Attention with Intention model

The hidden state of the decoder network is a function of previous state, previous output and the context. The initial state of the decoder network is the final state of the intention network. The context is selected from the set of encoder states, by using the attention mechanism proposed in [Bahdanau et al. 2014].

$$h_j^{(t)} = f\left(y_{j-1}^{(t)}, h_{j-1}^{(t)}, c_j^{(t)}\right) \tag{10}$$

$$c_j^{(t)} = z\left(h_{j-1}^{(t)}, c_t^{(s)} : \forall t = 1, ..., T\right) \tag{11}$$

4.3. End-to-End Dialogue Systems

Generative Conversation models must be capable of realistic, flexible interactions. The problem of dialog systems can be seen as a Partially Observed Markov Decision Process (POMDP). Such a model will be a cognitive system capable of natural language understanding, reasoning, decision making and Natural Language Generation. When compared to Neural Machine Translation systems, dialogue systems bear the burden of selecting from a wide range of plausible responses and a lack of phrase alignment between query and response. In [Serban et al. 2015], the seq2seq model is enhanced by mapping dialogue utterances to dialog states, followed by action generation mechanism. The dialog states are mapped to dialog acts - stochastically generate response utterances. The cornerstone of this model is that an internal state is maintained over long conversations.

Dialog is a sequence of M utterances involving two actors, $D = U_1, ...U_M$. Each utterance, U_i consists of N_i tokens, $U_i = w_i1, w_i2, ...w_iNi$. The tokens consists of words and speech acts like pause and end of turn. The probability of dialogue D is given by,

$$P_\theta(U_1, ..., U_M) = \prod_{m=1}^{M} P_\theta(U_m|U_{<m}), \tag{12}$$

$$= \prod_{m=1}^{M} \prod_{n=1}^{N_m} P_\theta(w_{m,n}|w_{m,<n}, U_{<m}) \tag{13}$$

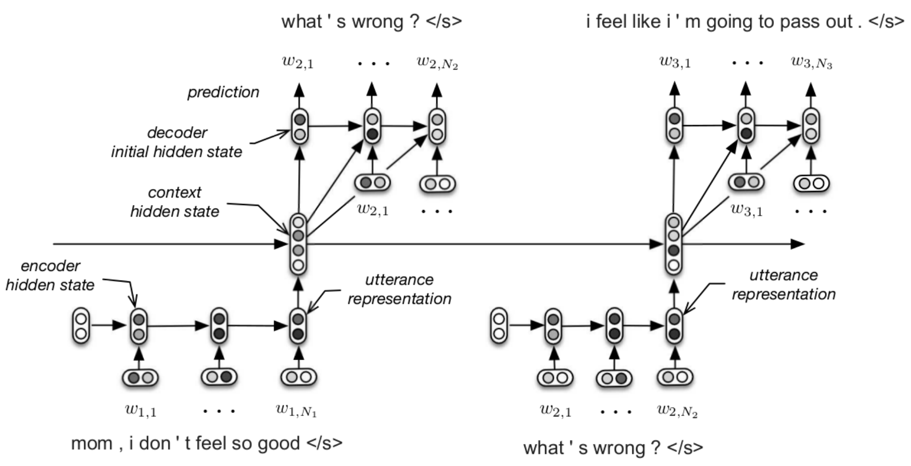

Figure 4. HRED architecture for dialog composed of three turns

The model consists of three RNNs - Encoder, Context RNN and Decoder. The encoder encodes tokens within an utterance. The context RNN encodes temporal structure of utterances. It allows information and gradient to flow over long time spans. The decoder RNN predicts one token at a time. The context vector represents the state of the dialog system. The prediction is conditioned on the state of the context RNN. The context RNN models the common ground between the speakers, like topic of conversation, concepts shared between them, etc,. It also reduces the number of computational steps between utterances.

5. Memory Augmentation

Neural Networks are memoryless. Memory is simulated through dependencies learned while training. An RNN acts as both the controller and the memory. Memory networks, neural networks augmented with memory, introduce a controller-memory split. Adding explicit memory to a dialog system, has quite a few advantages. It allows the system to maintain a history of the conversation. It allows the system to consider a list of facts, while answering a question. The system performs multiple lookups over the facts, attends to relevant facts, before responding to a query. The attention mechanism focuses on facts that are crucial for answering the question. Neural networks with memory augmentation, are termed Memory Networks. We will discuss End-to-End Memory Networks [6] and Dynamic Memory Networks [8], used for goal oriented question answering tasks.

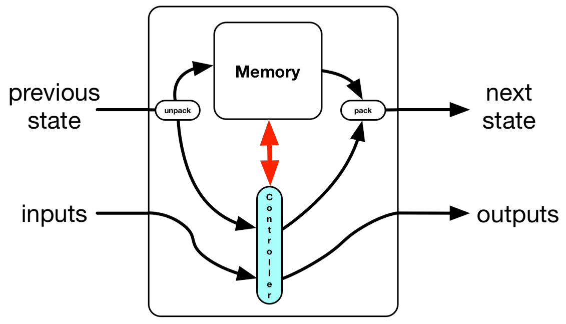

Figure 5. Conroller-Memory split

5.1. End-to-End Memory Networks

[Sukhbaatar et al. 2015] proposes a neural network with external memory. It is based on Memory Networks introduced in [Weston et al. 2014], which uses hard attention mechanism and attention supervision. The hard attention mechanism stochastically selects a memory vector from the memory, conditioned on the question vector. End to End Memory Network consists of an episodic memory module and a controller module. It uses soft attention mechanism and multiple lookups (hops) to read from the memory module. They are jointly trained with backpropagation. Supervision is applied only on the final output.

The facts or stories that contain the answer to the question, are encoded into memory vectors and stored in the episodic memory. The controller encodes the question into the question vector. The question vector acts as the addressing signal that influences the read from memory. A dot product of question vector and memory vectors is done, followed by a softmax. It yields a probability distribution over the memory vectors influenced by the question vector. A weighted sum of memory vectors (facts) is taken from memory. The controller state is updated to this vector. The read-from-memory process is repeated, now with the controller state influenced the read. Multiple lookups are performed on the memory, to get a good representation for the final controller state. The final controller state is used to predict the answer to the question.

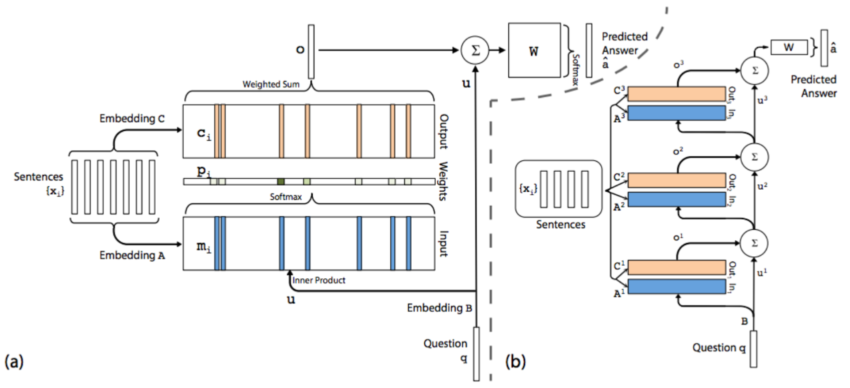

Figure 6. End-to-End Memory Networks (a): Single hop (b): three hops

Match between memory and controller state u is given by,

$$p_i = softmax(u^T m_i) \tag{14}$$

The output memory representation is a weighted sum of encoded form of input facts.

$$o = \sum_i p_i c_i \tag{15}$$

The final prediction is made by combining the final state of the controller and the ouput memory representation.

$$\hat{a} = softmax(W(o + u)) \tag{16}$$

5.2. Dynamic Memory Networks

Dynamic Memory Networks, introduced in [Kumar et al. 2015] and updated in [Xiong et al. 2016], makes use of a hierarchical recurrent sequence model to answer questions. The architecture is similar to [Sukhbaatar et al. 2015]. It consists of separate modules for encoding facts, questions, episodic memory and attention mechanism. The episodic memory module performs transitive reasoning over multiple sequences of facts. The input and question module share embeddings for sentence tokens. The input module encodes the facts into context, a sequence of vectors. The episodic memory module iterates over the context vectors, at each step updating the internal memory state. This process is repeated for multiple episodes, each time, updating the memory state, based on attention mechanism conditioned on the question vector. It should be noted that the question vector is set as the initial memory state.

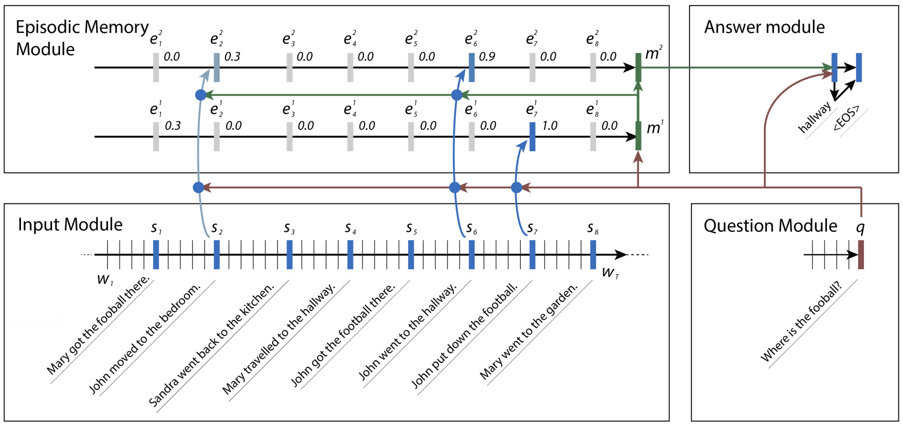

Figure 7. A sample qa example from bAbI tasks [Weston et al. 2015], illustrating the gating mechanism for attention

Multiple episodes i.e. multiple passes over the inputs, facilitates transitive inference. The first pass may uncover the need to retrieve additional facts, which is done in the subsequent episodes. The attention mechanism is implemented by modifying the gating function in the RNN cell. Gated Recurrent Unit (GRU) is used as the RNN cell. At each pass, the gate values are obtained using a scoring function G. G is a 2 layered neural network, which operates on a large feature vector, constructed from the context, memory state and the question vector. The architecture of the answer module depends the form of answers, in the question-answering task. If the answer is just one symbol, then it is just a classification problem. The final state of the memory, is used to produce a probability distribution over the vocabulary of answers. If the answer is a sequence, like a natural language response, an RNN is used as decoder, to predict a sequence of symbols. The initial state of the RNN is the final memory state.

For each pass, the gate values are computed as,

$$g_t^i = G(c_t, m^{i-1}, q) \tag{17}$$

The scoring function G is defined as,

$$G(c, m, q) = \sigma\left(W^{(2)} tanh\left(z(c, m, q) + b^{(1)}\right) + b^{(2)}\right) \tag{18}$$

where

$$z(c, m, q) = \left[c, m, q, c \odot q, c \odot m, |c - q|, |c - m|, c^T W^{(b)} q, c^T W^{(b)} m\right] \tag{19}$$

The gating mechanism that uses g_t^i, for memory update during each pass, is defined below.

$$h_t^i = g_t^i GRU(c_t, h_{t-1}^i) + (1 - g_t^i) h_{t-1}^i \tag{20}$$

$$e^i = h_{T_C}^i \tag{21}$$

The answering module consists of an RNN if the answer is a sequence of symbols, which defined below.

$$y_t = softmax(W^{(a)} a_t) \tag{22}$$
$$a_t = GRU([y_{t-1}, q], a_{t-1}) \tag{23}$$

6. Conclusion

We have explored some of the interesting models that augment the vanilla sequence to sequence model, to overcome some of it's limitations. We have also introduced memory networks, which maintains a memory state during each session, to keep track of the facts and answer a question, by iterating through the facts and performing reasoning. This kind of explicit memory look-up mechanism, if added to seq2seq model, could improve it's performance, by facilitating simple reasoning. Human level Conversational Models that can pass the Turing test, are a long way away, but the existing generative conversational models show great potential. There are quite a few open research problems to be solved here.

References

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473.*

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078.*

Kumar, A., Irsoy, O., Su, J., Bradbury, J., English, R., Pierce, B., Ondruska, P., Gulrajani, I., and Socher, R. (2015). Ask me anything: Dynamic memory networks for natural language processing. *CoRR, abs/1506.07285.*

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2015). Building end-to-end dialogue systems using generative hierarchical neural network models. *arXiv preprint arXiv:1507.04808*.

Sukhbaatar, S., Weston, J., Fergus, R., et al. (2015). End-to-end memory networks. In *Advances in neural information processing systems*, pages 2440–2448.

Vinyals, O. and Le, Q. (2015). A neural conversational model. *arXiv preprint arXiv:1506.05869*.

Weston, J., Bordes, A., Chopra, S., and Mikolov, T. (2015). Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.

Weston, J., Chopra, S., and Bordes, A. (2014). Memory networks. *arXiv preprint arXiv:1410.3916*.

Xiong, C., Merity, S., and Socher, R. (2016). Dynamic memory networks for visual and textual question answering. *arXiv*, 1603.

Yao, K., Zweig, G., and Peng, B. (2015). Attention with intention for a neural network conversation model. *arXiv preprint arXiv:1510.08565*.