# Study of Neural Network Based Approaches to implement Modern Chatbots

Parvathi Saxena
Manipal University
Jaipur,Rajasthan

Samrit Kumar Maity
C-DAC
Pune, Maharashtra

Manish Kumar Nirmal
C-DAC
Pune, Maharashtra

Akhilesh Kumar Sharma
Manipal University
Jaipur, Rajasthan

***Abstract:*** ***With the emerging trend of chat applications across the globe has made chatbots a new technology that has changed the way we do our conversations. It is a technology that brings many advantages with it for businesses that adopt it. Having an interactive chatbot whose narrative is not programmer driven but instead its own would give rise to a even stronger virtual companion. But these chatbot applications have certain shortcomings. The chatbot fails to understand the context of a conversation and looses the trail of it while conversing, or it might fail to process large training sets due to scalability issues. Here we will discuss the popular neural network models and word embedding techniques for building a smarter chatbot. We also address the shortcomings of these approaches so that we can identify the research area that needs attention to create a smart chatbot that can successfully identify the user's query and generate the right and meaningful response. Mostly chatbots fail to truly understand human conversations, making it understand the context of the conversation will be the first step to a better chatbot.***

**Keywords: Chatbots; LSTM(Long-Short Term Memory); deep learning; RNN (Recurrent Neural Network); CNN(Convolutional Neural Network); ChatBot; Conversational agent; AI (Artificial Intelligence); NLP (Natural Language Processing); ML (Machine Learning);**

## I. INTRODUTION

2016 was the year when chatbots came into the lime light. WhatsApp, Messenger, WeChat, QQ Mobile and a bunch of other popular messaging platforms now host chatbots that developers have built for them. Companies are increasingly engaging with their customers through chatbots. A smart chatbot could be taught of as a service that is driven by artificial intelligence and its narrative is self-driven. Various applications of chatbots other than chat support include booking, counseling, monitoring public chatrooms etc.

Earlier a chatbot's narrative was programmer driven, in the sense that the programmer would have to predict all possible inputs for the bot and make a list of those responses for every input. All of this failed as it was impractical for the programmer to predict all the possibilities. Before deep learning hit the scene, the classical approach to designing a chatbot included, (i) taking a labeled dataset and tokenize the text, (ii) create BoW (Bag of Words), (iii) convert the BoW into TF-IDF matrix(to reduce weight of common words), and (iv) train a classification algorithm(like Naïve Bayes. SVM, Random Forest) on this matrix. Using machine learning approach will lose the word ordering information[20]. E.g. "Pizza Hut is better than Dominos" & "Dominos is better than Pizza Hut" don't mean the same but will have same representation.

It suffers from data sparsity causing it hard to find a decision boundary. But with Neural Networks there has been a shift in this approach. Initially, we will encode all words into vectors and then feed these vectors as input to the neural network so as to train them for classification. What is to be kept in mind is that language is ambiguous and this imposes a challenge on the chatbot application to understand the interpretation of the word(i.e. context). To keep a conversation going the bot is required to stay in touch with the conversation's person/object. This would require the bot to successfully identify the object. But because of ambiguity, the conversation's object may change i.e. the user(human aspect) will change the conversation about something else. Understanding this change is also necessary. In the neural network approach since words are encoded into vectors, feeding them as input for processing would lead to either vanishing/exploding gradients problem. And these can cause the bot to falsely miss a word. Another issue would be processing very large datasets. Conversations today have evolved over the years. There are typos created in texting and tweeting like WRU for 'Where aRe yoU'. Millions of tweets and replies to social media posts are generated. Using this data for training would immensely help in training the network. There are two approaches to implementing deep learning in chatbots, (1) taking different components and applying it to each of them. One system is deep learning based used to interpret the language and another one to track the state of a conversation and then another one generates a response. To train this model, each of the separate systems would be trained separately to do its own task. The chatbot would collectively use those results from each system. This is unnecessarily complex to build. A better type (2) is called the "End-to-End" systems. These are chatbots that use one system that is trained on one dataset. There is no assumptions about the use case or the structure of the dialog. It is trained on the relevant data. End-to-End systems are what we should all be striving for. intuitively they make sense and they're starting to outperform all other systems.

The aim is to investigate and compare the alternative approaches that were followed by various researchers to implement smart chatbots in different scenarios.

This comparison will attempt to answer the following question: "What will make an even smarter chatbot?".This paper has been organized in 8 sections i.e. section I gives an introduction to the area of the subject, section II explains popular word embedding techniques, section III explains sequence-to-sequence model, section IV explains the convolution network model, section V explains the long-short term memory, section VI explains gated recurrent unit for text classification, section VII gives the conclusion of the study and section VIII gives the refernces of this study.

## II. WORD EMBEDDINGS

Many Deep Neural Network models today are based on word embedding, where each word in the sentence is mapped to a low dimensional vector.Word Embedding is a NLP technique for finding links between words from raw text through vectors. The basic idea is to convert words in a sentence into meaningful vectors(a.k.a semantic hashing). It learns things based on the context. With the context it can find other words with the similar context based on close spatial positions. Consider the following example, if we are looking for a pizza parlor for dinning, with word embedding we could find it by, in the fig.1, we can find pizza serving restaurants nearby and non-pizza serving restaurants far away. The basis is that word embedding is trained in an unsupervised way i.e. no labelled data is there. Data Sparsity is reduced since we are not dealing with 1's and 0's instead we have floating values[5]. But Gehui[4] et. al argued that word embedding cannot express the interpretation of the word in context as it may suffer from exploding or vanishing gradientproblems.

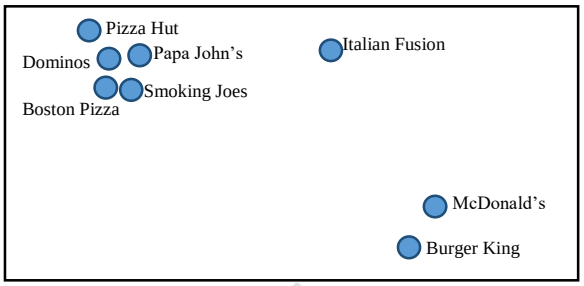


**Fig 1: Example for word embedding**

Word embedding allows to analyze words mathematically. The high dimensional vectors represent words and each dimension encodes a different property. It is a simple light-weigth neural network that takes a word as a vector. In this technique the words of a sentence are broken down and each word is then converted into a vector. After that it is passed through a neural network to find close words to that word. The output of this network.

A. word2vec

It is a light-weight neural network that takes a word as a vector. In this technique the words of a sentence are broken down and each word is then converted into a vector. After that it is passed through a neural network to find close words to that word. The output of this chain is a set of vectors that are spatially close to the input vector. It is a predictive approach.

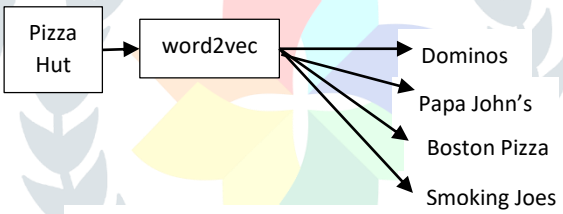


**Figure 2: word2vec processing**

Consider the example, where the user query is "Find me a pizza parlor like Pizza Hut.". Here each word in the query will be broken down and converted into a vector. With the Pizza Hut vector we will be adequate to find a set of vectors including {'Dominos','PapaJohn's','BostonPizza','Smoking Joes'}. Using this approach enables us to better exploit inter-lexical dependencies between different parts of the input[2].

B. Co-occurrence Matrix

There are other word embedding techniques based on co-occurrence matrix. One common method is to decompose the co-occurrence matrix into smaller matrices. This technique is based on linear algebra and includes breaking the matrix into (say) 2 matrices and the multiplication of these two matrices gives the co-occurrence matrix.

| | Pizza Hut | Italian Fusion | Burger King |
|---|---|---|---|
| Pizza Hut | 0 | 3 | 1 |
| Italian Fusion | 3 | 0 | 1 |
| Burger King | 1 | 1 | 0 |

**Table 1: Co-occurrence Matrix**

C. Glove

GLObalVEtors(GLOVE)[10] is a count-based approach. A co-occurrence matrix is constructed based on context by words. For each row (i.e. word) it will count the frequently occurring context(i.e. column). Since the number of columns can be large, it factorizes the matrix to get a

lower dimensional matrix which represents words by features. For each row has a feature representation for each word. They can also be trained on large datasets. It performs faster than word2vec.

## III. SEQUENCE-TO-SEQUENCE

A Seq2seq model generally consists of two recurrent neural networks. One is the encoder and the other is the decoder. The main objective of this model is to convert a sequence of symbols into a fixed length vector that will help to encode only important information and losing the unnecessary

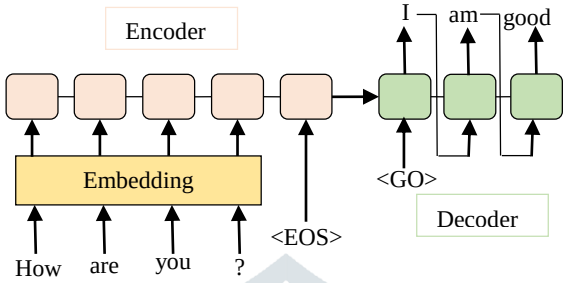


**Fig 2: Sequence-to-sequence**

information. Each hidden state in the encoder will influence the next hidden state until the final hidden state is achieved. This state is the context (a.k.a thought vector or summary of sequence). With the context, the decoder generates another sequence which is influenced by both the context and previous state. The model processes input as one symbol(word) at a time. Quoc[2] et.al proposed a seq2seq framework that generates simple conversations given a large conversational training dataset. Their model was trained on a large dataset of movie subtitles. Since there was no determining who is talking they considered every sentence as a new question dialogue and the next sentence would the reply dialogue. They used a greedy approach where during inference if the predicted output didn't match the true output then the predicted output would be fed back as an input to predict the next output. They used padding for the input data. So during encoding when the <EOS> symbol is found the decoder is activated. This was the first approach to exploit the encoder-decoder architecture. Their model lacked to ensure consistency and general world knowledge, therefore, failing to keep the conversation going. Existing conversational datasets don't contain all of the world's knowledge and therefore this becomes a primary challenge in building fully data-driven conversational models[2]. Marjan[3] et.al proposed a knowledge-grounded approach where it avoids to learn the same conversational pattern for each distinct entity. It relies on retrieved facts to generate a response. It is made of two GRU networks, unlike Quoc's[2] approach where two LSTM networks are used. They used a separate memory network to pick relevant responses (facts) from the knowledge base. The knowledge base consisted of both world knowledge and conceptual knowledge based on conversations. This was collected from various sources like Wiki, Foursquare, and Twitter. The context generated from the encoder and the facts from the memory network are passed to a softmax function, its output is fed to the decoder to generate the final output(answer).

Various challenges faced by this model include (1) it cannot handle variable length sequences, and (2) large vocabulary size will slow the training process. Common solutions to (1) include (a) padding and (b) bucketing.

1. PADDING
   This technique converts a variable length sequence into a fixed length sequence by appending pads(symbols) into that sequence. Symbols include,
   1) EOS (End Of Sequence)
   2) PAD - Filler
   3) GO – indicator to start decoding
   4) UNK (Unknown) – used when the word is not in the vocabulary

   e.g.

*Question:*"FIND ME A PIZZA PARLOR LIKE PIZZA HUT?"

⇓

[PAD, "?", "HUT", "PIZZA", "LIKE", "PARLOR", "PIZZA", "A", "ME", "FIND"]

⇓

length =10

*Answer*: "DOMINOS, PAPA JOHN'S, BOSTON PIZZA, SMOKING JOES"

⇓

[GO, "DOMINOS", "PAPA JOHN'S", "BOSTON PIZZA", "SMOKING JOES", PAD, PAD, PAD, PAD, PAD]

⇓

length =10

   The problem faced by this approach is overshadowing actual information i.e. for large sequences we set the length as 100 then for smaller sequences there will be larger number of pads and very less data. This is wastage of memory.

2. BUCKETING
   This technique converts a variable length sequence into buckets of different sizes.

e.g. Buckets are defined as [(5,10), (10,15), (20,25), (40,45)]

*Question:*"NAME SOME PIZZA PARLORS?"

⇓

["?", "PARLORS", "PIZZA", "SOME", "NAME"]

⇓

length =5

*Answer:*"PIZZAHUT, DOMINOS, PAPA JOHN'S, BOSTON PIZZA, SMOKING JOES"

⇓

[GO, "DOMINOS", "PAPA JOHN'S", "BOSTON PIZZA", "SMOKING JOES", PAD, PAD, PAD, PAD, PAD]

⇓

length =10

Sequence-to-sequence models must be trained with large datasets but the resulting output may not maintain consistency and fluency[2].

## IV. CONVOLUTION NETWORK

Convolutional Neural Network(CNN) model was inspired by animal visual cortex. They are excellent feature extractors in image regardless of position in an image. Generally used in computer vision however recently applied to NLP tasks like text classification and is giving promising results. The selection criteria for text classification is similar to image classification, only difference is that instead of pixel values we have matrix of word vectors[6]. It performs better than statistical models. Collobert[5] et.al first proposed CNN for text. In their work considering a toy problem like given a sentence we will have word embeddings for each word, to convolve over these we require a window size. A simple concatenation is done on these word embeddings to get a single vector. This vector is then multiplied with a weight matrix that has the weights of the convolution layer. This process is followed until the whole sentence is convolved. With the multiple vectors found at the end of convolving the entire sentence, max-pooling is done. Here the maximum value from each vector will be taken to form a single vector (a.k.a most relevant vector of the sentence) that is then fed to a multi-layer perceptron to get the final output. Kim[6] proposed multiple models of CNN for text classification. They were basically (1)CNN-rad which is a baseline model in which the words are randomly initialized and only changed during the training process. (2)CNN-static in which the word vectors are not changed i.e it has pre-trained vectors from word2vec. (3)CNN-non-static model the back-propagation is done on the word vectors too so they change their value when they are trained. (4)CNN-multichannel model uses both static and non-static models. Here the model has two sets of word vectors where each set is a channel to which a filter is applied. Nadeem[7] et.al proposed a CNN for text classification for question-answering domain. The model is based on window sizes (3,4,5) and used 33 filters to roll over the entire sentence matrix reducing it to a low dimensional matrix. The embedded sentences are than padded to make all the word matrices to the same size. Liu[1] proposed a CNN model for extreme multi-label classification called the XML-CNN(eXtreme Multi-Label). They made a small tweak to the existing CNN model, like swiping through the convolution filter in the word-embedding dimension (i.e. the spatial dimension for an image). This helped capture salient features. They compared their model with non-deep learning models like FastXML, SLEEC and PD-Sparse, and deep learning models like FastText, Bow-CNN and CNN-Kim. XML-CNN would scale to extremely large problems, large feature spaces and label spaces. Dauphin[15] et. al gave evidence that gated CNNs used for language modeling tasks outperformed LSTMs. Adel[32] et. al work supported CNN over LSTM/GRU based models for long sentences. In general CNNs are hierarchical is nature as compared to RNNs which are sequential[31].

## V. LONG-SHORT TERM MEMORY

With RNN(Recurrent Neural Networks) being used to learn from stored information over extended time intervals proved time-consuming and insufficient, which lead to the discovery of LSTM(Long-Short Term Memory) in 1997[7]. These neurons can train on something over time and remember something from an infinite number of steps back too. Its structure has a more complex neuron. It is similar to an RNN and processes information sequentially. The difference is the operations with the LSTM cells. These neurons have an internal hidden state that can be updated or it can be forgotten with the operations. RNN is good for sequence processing but suffers from short-term memory[11]. LSTM and GRU were created to mitigate this with a mechanism called gates. It has 4 gates namely, (1)cell gate, (2) forget gate, (3)input gate, (4)output gate. The cell gate carries information throughout the sequentially processing. In theory, even information from earlier time steps can be carried all the way to the last time step

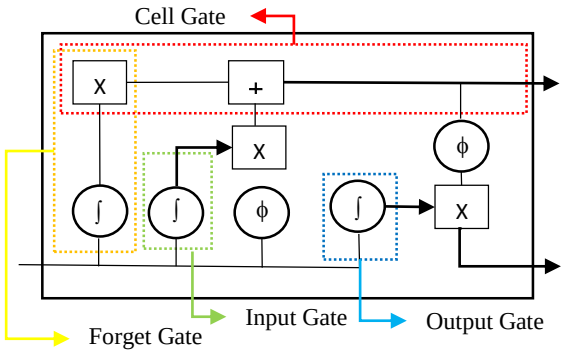


**Figure 3: LSTM**

and therefore reducing the effects of short-term memory. The gates are just different neural networks that decide which information is allowed on the cell gate and regulate the flow of information from one time step to another. Gates contains sigmoid(∫) and tan(ϕ) activation functions. The ∫ is similar to the ϕ where instead of squishing values to -1 to 1, it will be between 0 and 1. This helps in update and forget operation. The forget gate decides what information should be updated or forgotten. Information from a previous hidden state and information from the current input is passed to ϕ, to get output between 0(forget) and 1(remember). To regulate the network, the previous hidden state and the current state are passed to ϕ, its output is multiplied with ∫ which decides what information to keep. With these information, we calculate the cell state which is then multiplied with the forget vector[9]. The output from the input gate is used to do polarize addition which updates the cell state to new values giving a new cell state. The output gate decides what the next hidden state should be. The hidden state is used for prediction and contains information about previous inputs.

The term LSTM came about because the standard RNN operates in such a way that the hidden state activations are influenced by the other local activations closest to it, which corresponds to "short-term memory". While the network weights are influenced by the computation that takes place over the entire sequence, which corresponds to "long-term memory". This redesign of RNN caused the activation state to also act as weights and preserve information over long periods of time.

The exploding/vanishing gradients problem[17] was analyzed in Hochreiter[7] et. al work. Su[8] et. al proposed a LSTM-based multi-layer embedding model for a Q&A chatbot for the elderly that is used to extract the semantic information between words and sentences. Ney[12] et. al compared LSTM with the traditional RNN using English and French corpus in terms of perplexity and word error rate. LSTM outperformed RNN proving to be a state-of-the art speech recognition system.Felix[14] et. al found that LSTM memory is limited by the number of memory blocks and increasing them homogenously would not help. Some change to the model was necessary for keeping an effective learning process. They also found that where information is kept or thrown, can be triggered by noisy input sequences and therefore the LSTM must be fed with appropriate feature detector.

## VI. GATED RECURRENT UNIT

A newer generation of RNN is the GRU which is similar to LSTM. There is no cell gate instead used a hidden state to transfer information. It only has two gates namely (1) reset gate and (2)update gate. The update gate acts similar to the input gate and the forget gate of the LSTM. It decides what information to keep or forget. They are faster than LSTM because they have less tensor operations.

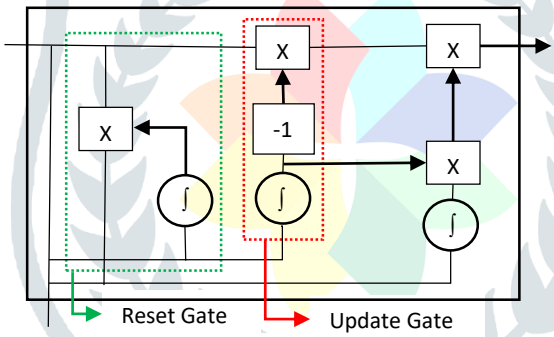


**Figure 4: GRU**

Cho[20] et. al exploited a RNN encoder-decoder model where one GRU layer was used to model the history of the conversation at the sentence-level while another GRU layer was responsible for modelling each sentence at the word-level. They found that encoding long sequences to fixed-length vectors was problematic for translating. Chung[19] et. al proved that GRU performs better than LSTM in the learning process. As the network uses local context and the current input to regulate the flow of information, they found that removing the local context can reduce the computational complexity of the model but also loss of sequential information. Lai[21] et. al proposed a variant(iCARNN) of the traditional GRU by introducing a new gate called the independent gate. This gate controls the context and input causing the gate computations to simplify. This was unlike the RNN, the gate computations at all time steps can be calculated in parallel. But it retains the sequential nature of RNN. They found that iCARNN either performs the best or very close to the best on the various datasets of different sizes they used. Zhang[22] et.al proposed a hierarchical GRU model that incorporates attention mechanism to better capture important information from a document. This approach effectively reduced the computational complexity and improved the prediction accuracy. Yin[31] et. al found GRU to outperform CNN when fed with larger sentences. Trofimovich[33] et. al work on sentiment analysis of Russian tweets proved GRU performs better than CNN and LSTM models.

## VII. CONCLUSION

Chatbots are limited by its own capacity to grasp information and give proper replies. They fail to identify inconsistences or keep up the conversation when the subject changes. It doesn't truly understand the meaning of the sentences it is fed. Using neural networks to improve its learning process has made a tremendous change. Some promising models were discussed in this paper. But there is no right way for selecting the deep neural network model for this problem[31]. We were able to identify that modified RNNs like LSTM and GRUs are providing promising results and successfully learns information to make predictions. LSTMs and GRUs are being used in the state-of-the art deep learning applications like speech recognition, speech synthesis[15], natural language understanding[14] etc. Chatbots have become a new source of interaction between humans and AI.

## VIII. REFERENCES

[1] Liu, J., Chang, W.C., Wu, Y. and Yang, Y., 2017, August. Deep learning for extreme multi-label text classification. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 115-124). ACM.

[2] Vinyals, O. and Le, Q., 2015. A neural conversational model. arXiv preprint arXiv:1506.05869. Ghazvininejad, M., Brockett, C., Chang, M.W., Dolan, B., Gao, J., Yih, W.T. and Galley, M., 2017. A knowledge-grounded neural conversation model. arXiv preprint arXiv:1702.01932.

[3] Shen, G., Yang, Y. and Deng, Z.H., 2017. Inter-weighted alignment network for sentence pair modeling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (pp. 1179-1189).

[4] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K. and Kuksa, P., 2011. Natural language processing (almost) from scratch. Journal of Machine Learning Research, 12(Aug), pp.2493-2537.

[5] Kim, Y., 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[6] Amin, M.Z. and Nadeem, N., 2018. Convolutional Neural Network: Text Classification Model for Open Domain Question Answering System. arXiv preprint arXiv:1809.02479.

[7] Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.

[8] Su, M.H., Wu, C.H., Huang, K.Y., Hong, Q.B. and Wang, H.M., 2017, December. A chatbot using LSTM-based multi-layer embedding for elderly care. In Orange Technologies (ICOT), 2017 International Conference on (pp. 70-74). IEEE.

[9] Gers, F.A., Schmidhuber, J. and Cummins, F., 1999. Learning to forget: Continual prediction with LSTM.

[10] Pennington, J., Socher, R. and Manning, C., 2014. Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).

[11] Gers, F.A., Schraudolph, N.N. and Schmidhuber, J., 2002. Learning precise timing with LSTM recurrent networks. Journal of machine learning research, 3(Aug), pp.115-143.

[12] Bengio, Y., Simard, P. and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 5(2), pp.157-166.

[13] Martens, J. and Sutskever, I., 2011. Learning recurrent neural networks with hessian-free optimization. In Proceedings of the 28th International Conference on Machine Learning (ICML-11) (pp. 1033-1040).

[14] Sundermeyer, M., Schlüter, R. and Ney, H., 2012. LSTM neural networks for language modeling. In Thirteenth annual conference of the international speech communication association.

[15] Dauphin, Y.N., Fan, A., Auli, M. and Grangier, D., 2016. Language modeling with gated convolutional networks. arXiv preprint arXiv:1612.08083.

[16] Van Den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K., 2016. Wavenet: A generative model for raw audio. CoRR abs/1609.03499.

[17] Miller, J. and Hardt, M., 2018. When Recurrent Models Don't Need To Be Recurrent. arXiv preprint arXiv:1805.10369.

[18] Garigliotti, D. and Balog, K., 2018, March. Towards an Understanding of Entity-Oriented Search Intents. In European Conference on Information Retrieval (pp. 644-650). Springer, Cham.

[19] Chung, J., Gulcehre, C., Cho, K. and Bengio, Y., 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555.

[20] Bahdanau, D., Cho, K. and Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[21] Tran, Q.H., Lai, T., Haffari, G., Zukerman, I., Bui, T. and Bui, H., 2018. The Context-dependent Additive Recurrent Neural Net. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (Vol. 1, pp. 1274-1283).

[22] Shi, Y., Yao, K., Tian, L. and Jiang, D., 2016. Deep LSTM based feature mapping for query classification. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 1501-1511).

[23] Zhang, L., Zhou, Y., Duan, X. and Chen, R., 2018, March. A Hierarchical multi-input and output Bi-GRU Model for Sentiment Analysis on Customer Reviews. In IOP Conference Series: Materials Science and Engineering (Vol. 322, No. 6, p. 062007). IOP Publishing.

[24] Shen, G., Yang, Y. and Deng, Z.H., 2017. Inter-weighted alignment network for sentence pair modeling. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (pp. 1179-1189).

[25] Lai, T.M., Bui, T. and Li, S., 2018. A Review on Deep Learning Techniques Applied to Answer Selection. In Proceedings of the 27th International Conference on Computational Linguistics (pp. 2132-2144).

[26] Lan, W. and Xu, W., 2018. Character-based Neural Networks for Sentence Pair Modeling. arXiv preprint arXiv:1805.08297.

[27] Csáky, R.K. and Recski, G., Deep Learning Based Chatbot Models.

[28] Choi, Y., Kim, S. and Lee, J.H., 2016, August. Recurrent Neural Network for Storytelling. In Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems, 2016 Joint 8th International Conference on (pp. 841-845). IEEE.

[29] Su, M.H., Wu, C.H., Huang, K.Y., Hong, Q.B. and Wang, H.M., 2017, December. A chatbot using LSTM-based multi-layer embedding for elderly care. In Orange Technologies (ICOT), 2017 International Conference on (pp. 70-74). IEEE.

[30] Bhagwat, V.A., 2018. Deep Learning for Chatbots.

[31] Yin, W., Kann, K., Yu, M. and Schütze, H., 2017. Comparative study of cnn and rnn for natural language processing. arXiv preprint arXiv:1702.01923.

[32] Adel, H. and Schütze, H., 2016. Exploring different dimensions of attention for uncertainty detection. arXiv preprint arXiv: 1612.06549.

[33] Trofimovich, J., 2016. Comparison of neural network architectures for sentiment analysis of russian tweets. In Computational Linguistics and Intellectual Technologies: Proceedings of the International Conference Dialogue 2016, RGGU.