Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS 2019)
IEEE Xplore Part Number: CFP19K34-ART; ISBN: 978-1-5386-8113-8

Humanizing the Chatbot with Semantics based Natural Language Generation

1st Mayuresh Virkar
Computer Science and Engineering,
Walchand College of Engineering,
Sangli(MS), India.
mayuvirkar@gmail.com

2nd Vikas Honmane
Computer Science and Engineering,
Walchand College of Engineering,
Sangli(MS), India.
vikas.honmane@walchandsangli.ac.in

3rd S. Upendra Rao
IT-SAP,
State Bank Global IT Centre,
Navi Mumbai(MS), India.
gmit.sap@sbi.co.in

***Abstract*—This paper introduces approach made for improving the efficiency of the chatbot or artificial conversational entity used in various commercial and banking sector. Humanizing is to improving the response generation ability of the chatbot. In this work, an attempt has been made to generate more natural response for a question asked to an artificial conversational entity by using various Natural Language Processing (NLP) and Natural Language Generation (NLG) techniques. Paraphrase generation plays a main role by generating semantically similar response for a query making it more natural.**

***Index Terms*—Natural Language Processing (NLP), Natural Language Generation (NLG), Paraphrase generation.**

I. Introduction

Chatbot or Artificial Conversational Entity has been wildly used to in today's commercial and banking sector for customer support and information acquisition. Chatbot are the programs often designed to convincingly simulate how a human would behave as a conversational partner, thereby assisting the customer. Natural language generation (NLG) is the natural language processing task of generating natural language from a machine representation system such as a knowledge base using a logical form. It could be said an NLG system is like a translator that converts data into a natural language representation. Current chatbots are trained to understand the user queries and is able to retrieve the answers from a knowledge base to respond to the user. Since these answers are fixed for a particular query asked by the user, the end user can easily find out that a bot is answering the query.

Traditional chatbot system have a drawback of generating a fixed response for the question asked by the customer as the question answer pairs are fixed in the database. As the chatbot always gives the same response for a given question the customer gets disconnected from the chatbot as he realizes that a machine is answering. Humanizing the chatbot will give more human touch to the chatbot by natural language generation which will generate variations of fixed response thereby giving a personal touch to the chatbot. To improve customer satisfaction we shall use techniques like paraphrasing techniques to generate variations and make the response more natural. Natural Language Generation include stages like content determination, document structuring, aggregation and lexical choice. In this work, we use Natural Language Generation (NLG) techniques to generate variations in responses for a fact or information retrieved from knowledge base. We will be using different techniques for paraphrasing and combination of machine learning algorithms. Objective will be to bring any improvement in existing chatbot for customer satisfaction. Paraphrase generation is one such part of Natural Language Generation where semantics is main concerned to be focused. Paraphrasing is expressing a single sentence by different ways keeping the meaning same. It may include replacing different words with same meaning i.e. synonyms or abbreviations along with phrases. For example,

Given sentence:
Can I transfer money overseas
Paraphrases:
Can I send money overseas Do I have facility to transfer funds abroad Can I transfer money outside my country

II. Literature Review

Natural language generation is new and upcoming field so there have been various experts with different perspectives working to enhance this field. There is survey[1] of computational approaches for paraphrasing. Paraphrasing includes methods such as generation, identification and acquisition. Statistical Machine Translation technique[2] has been as per basis on the context of Monolingual translation to generate variations for a single sentence. Neural Machine Translation (NMT) based approach has been used along with Paraphrase Database (PPDB) and Rule based Machine Translation for a Question Answering system [3]. It has proposed a framework for learning paraphrases for question answering and choose correct answers based on the probabilities. Keyword to Question (K2Q) [4] is one more approach used with algorithm called synthetic keyword-question generation along with Keyword Query Generation Model (KQGM) and NMT as well. Structure and Word techniques for paraphrasing have been used to generate accurate and syntactically correct natural language sentences [5]. It typically uses subject-object-verb

form of structures to generate new sentences keeping the meaning same. A black box machine translation [6] point of view has been used for sentential paraphrase generation using PPDB language packs. A method that paraphrases a given sentence by first generating candidate paraphrases and then ranking (or classifying)[7] them. The candidates are generated by applying existing paraphrasing rules extracted from parallel corpora. In deep neural networks[8] approach is about Variational Autoencoder (VAE) along with Long Short-Term Memory(LSTM) together called as VAE-LSTM model for paraphrase generation with references by generating sentences[9] and calculating the loss function for the output.Along with generating the sentences for given input, it is necesssary of calculating the semantic similarity score using words and sentences plays a very crucial role in paraphrase generation using word embeddings[13] and sentence embeddings[14].

III. Methodology

In a chatbot system, usually a chatbot analyzes a query asked by a person and retrives a particular answer from the fixed database. Usually, answer is retrived based on the basic keyword matching and a fixed response is given as the output. This paper basically focuses on the improving this response generation by making it to generate variations for a response rather than a one fixed response for a question asked. The methodology considers a response by a chatbot which will be a sentence as sequence of words to be input for the model. Fig.1 describes a flow chart of the entire model.

A sentence cannot be processed directly to the model so it is need to perform some Natural Language Processing(NLP) to make further operations. Some operations performed as like Tokenization and Parts-of-Speech Tagging.

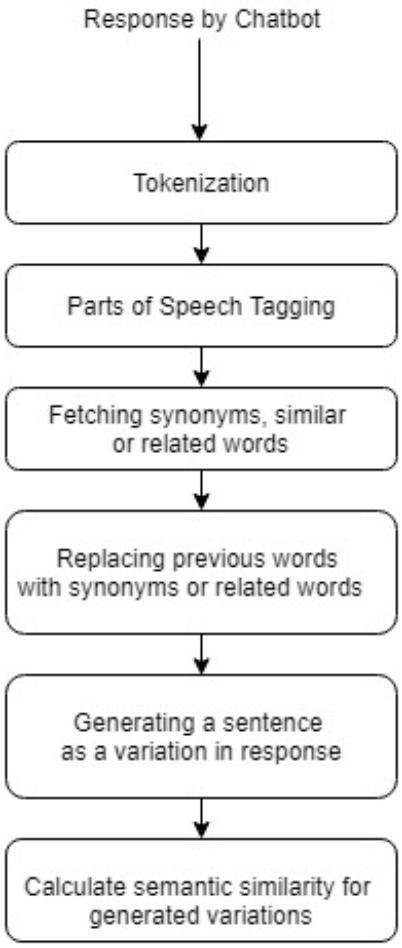

Fig. 1. Methodology

A. Tokenization:

Tokenization is operation used to split a sentence in individual words or tokens. Each word is separate token to be passsed for further processing.

B. Parts-of-Speech Tagging:

Parts-of-Speech Tagging or POS Tagging is operation used to identify the part of speech each individual word represents in the particular sentence. Part of speech can be as noun, adjective, verb, and adverb. Appropriate part of speech of a word in a sentence need to be identified. Some words may have more than one part of speech like word 'visit' can be noun or verb which entirely depends on sentence structure.

C. Word Reinstatement

Now after identifying the parts of speech we need rich lexical resource to substitute words with similar meanings. In this work, WordNet dictionary has been used to make words available for replacements. WordNet is rich collection of words for English language. It provides all that features required for any project work based on text data. So basically *synsets* is one such feature to extract the synonyms for given words in the sentence. *Synsets* are basically a group of synonyms, similar or related words for a particular word in the dictionary. Thus, can be readily used for replacing such words to generate parphrases. For example, *synsets* for word 'balance' have been shown in Fig. 2.

```
In [1]: from nltk.corpus import wordnet as wn

In [2]: wn.synsets('balance')
Out[2]:
[Synset('balance.n.01'),
 Synset('balance.n.02'),
 Synset('proportion.n.05'),
 Synset('balance.n.04'),
 Synset('remainder.n.01'),
 Synset('balance.n.06'),
 Synset('libra.n.01'),
 Synset('libra.n.03'),
 Synset('symmetry.n.01'),
 Synset('counterweight.n.01'),
 Synset('balance_wheel.n.01'),
 Synset('balance.n.12'),
 Synset('balance.v.01'),
 Synset('balance.v.02'),
 Synset('poise.v.04'),
 Synset('balance.v.04')]
```

Fig. 2. Synsets for word 'balance'

```
In [3]: runfile('C:/Users/Mayuresh/.spyder-py3/temp.py', wdir='C:/Users/
Mayuresh/.spyder-py3')
Word2Vec(vocab=15, size=100, alpha=0.025)
[-8.6150639e-04  4.5555965e-03 -1.1489552e-03  1.9877539e-03
  2.6244971e-03 -1.8952532e-03  4.9195602e-03 -4.0874891e-03
 -2.1195877e-03  2.9253492e-03  1.5948007e-03 -2.4741224e-03
 -1.1376872e-03 -4.5113768e-03  2.6972855e-03 -6.2894885e-04
 -4.8228544e-03 -9.9215144e-04 -4.9433131e-03 -1.3111995e-03
 -4.8307772e-03 -4.9786828e-04  4.1566957e-03  4.9467166e-03
  8.3288841e-04  2.1589352e-03 -8.6813886e-04 -1.9146096e-03
 -1.2743225e-03 -2.7706986e-03  1.6237626e-06 -2.3102397e-03
 -4.9914145e-03  8.4247277e-04 -1.8342482e-03  2.1670049e-03
 -4.9864152e-04 -5.4706930e-04 -2.5417018e-03 -4.2824298e-03
 -1.2988031e-03 -3.1940835e-03  3.0859814e-03  2.1587298e-03
  1.7102771e-03  2.4061247e-03 -2.0750263e-03  4.7578379e-03
  2.2488268e-04 -4.2378865e-03  2.8903936e-03 -4.1393107e-03
 -3.6391618e-03  2.5584209e-03 -2.6372813e-03 -3.3096855e-03
 -2.1968002e-03 -1.9625660e-03  4.2472607e-03 -4.7317878e-03
  1.4597967e-03 -2.7334422e-04  1.8725260e-03  4.1557993e-03
 -3.5983517e-03  6.9260248e-04  2.6870861e-03 -4.0491181e-03
  2.7636136e-03  2.6132590e-03 -3.9624632e-03  1.6904215e-03
 -4.2984094e-03  3.4440099e-03  3.5964847e-03  4.4298363e-03
 -1.9420024e-03  4.9562864e-03  2.7562000e-03 -3.3658959e-03
 -2.6702320e-03 -1.7340226e-03 -1.2635945e-03 -3.9128573e-03
  1.1901165e-03  2.1146799e-03 -4.1668140e-03 -1.7223145e-03
  4.1557676e-03  3.7625138e-04  1.3889066e-03 -2.4066723e-03
  1.3277661e-03 -1.4400268e-03  4.5252731e-03 -4.1371658e-03
  3.0713696e-03  4.1667144e-03  3.2928032e-03  2.6917469e-03]
Word2Vec(vocab=15, size=100, alpha=0.025)
```

Fig. 3. Word2vec matrix for word 'balance'

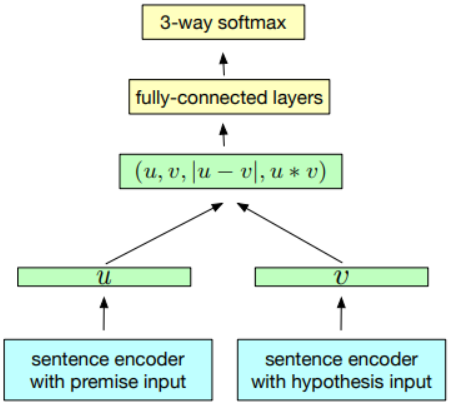

Fig. 4. NLI Training[13]

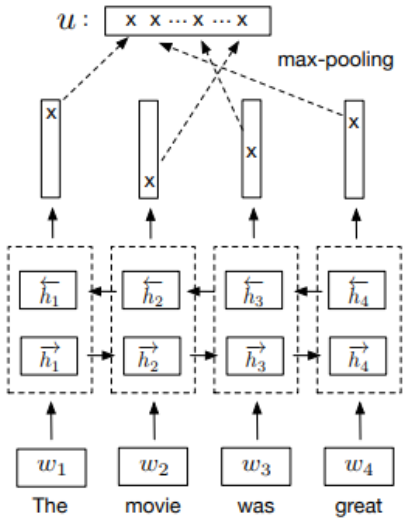

Fig. 5. Bi-LSTM max-pooling network

D. Sentences Semantic Similarity

Sentence Semantic Similarity plays a major role in this work. In previous step, system replaced all the possible word one-by-one to the original sentence to generate essential paraphrses for a sentence. Therefore, all the words available in the *synsets* may not match the actual meaning or context of the sentence. So it is need to calculate the semantic similarity between the sentences to find the actual paraphrases for the sentence.

Sentence Similarity score can been calculated using two methods : word embeddings and sentence embeddings.

1) Word embeddings: Word embeddings is part of feature engineering in Natural Language Processing required for text data. In this, text data is converted to vector form to process the data further. This vector space may vary from 1-dimension to n-dimension. More the dimensions, more better results. There are various techniques available to convert text to numbers, one such is *Word2vec*. Word2vec is popular model developed by Google, which has been used in proposed system.

Word2vec generates a matrix of tensor values with multiple dimensions for a word in a sentence. Fig. 2 shows matrix for word 'balance'. As per figure, multiple rows in matrix describe various dimensions in which word can be used in various contexts. Therfore, while calculating the semantic similarity, these matrix are taken into consideration. *Cosine Similarity* measure is applied on these vectors as average-of-vectors and final semantic similarity score is computed between sentences.

2) Sentence embeddings: Sentence embeddings are similar to that of word, but the difference is that entire sentence is collectively converted to vectors to represent the fact or semantic representation of the sentence. This basically involves the sentence encoders whose architecture is based on neural networks like the Long Short-term Memory networks. This model is trained on Standard Natural Language Inference (SNLI) corpus which finds the textual features, relationships and entailment among the sentences. After training the sentence encoders, the generated vectors are passed through methods like concatenation *(u,v)*, element-wise product*(u*v)* and difference*(u-v)* to extract relations. Further they are passed to fully connected layers for 3-way classification.

E. Sentence encoder Architecture: BiLSTM with mean/max pooling

A wide variety of neural networks for encoding sentences into fixed-size representations are available, but as per experiments[13] Bidirectional LSTM has better results compared to others. A Bi-LSTM computes a set of T vectors $\{h_t\}_t$ for T-words $\{w_t\}_t$. Here, h_t is combination of forward nad backward LSTMs to obtain sentence embeddings.

$$\overrightarrow{h_t} = \overrightarrow{LSTM_t}(w_1, ..., w_T)$$
$$\overleftarrow{h_t} = \overleftarrow{LSTM_t}(w_1, ..., w_T)$$
$$h_t = [\overleftarrow{h_t}, \overrightarrow{h_t}]$$

IV. Implementation Details

Natural Language Toolkit(NLTK) library has been used to perform basic NLP tasks. Pytorch library has been used

Input Sentence	Generated Paraphrase	Word embeddings score	Sentence embeddings
visit your nearest branch with required documents	travel your nearest subdivision with needed papers	0.9687	0.9971
visit your nearest branch with required documents	inflict your nearest branch with required documents	0.9806	0.8765
you can keep track of payments online	you can maintain track of payments online	0.9668	0.9629
you can keep track of payments online	you can preserve track of payments online	0.9716	0.9282
what is daily trasaction limit	what is day-to-day dealings terminal point	0.9226	0.5226
abhijeet is best student in the class	abhijeet is skillfully scholar person in the class	0.8372	0.6749

to construct the sentence encoder architecture with LSTM networks. The model is trained on GPU to boost the performance and reduce the response time.Implementation has been accomplished using Python 3.7 with Anaconda environment and PyCharm Community edition as IDE.

V. Result

Experiment has successfully generated the parapharses as a response by the chatbot. The results of the experiment have been represented in the table. Comparison of the word embedddings score and sentence embeddings score for the paraphrases have been shown clearly in the table. So observing the results there is some fluctuation in the results as in some cases the score of word embeddings is accurate while there is good score of sentence embeddings in other.

VI. Conclusion

In this work, paraphrase generation concept has been introduced for improving the response generation ability of a chatbot. The goal was to generate semantically similar sentence for a response generated by a chatbot. As per results, model generates semantically similar sentence for a given sentence. Comparing the results, sentence emeddings generates more accurate similarity score for calculating the semantic similarity of the sentences as compared to that of word embeddings score. Sentece embeddings extracts the actual fact of the sentence. Observing fist and second paraphrases 'inflict' word does not carry correct meaning as per context. Along with that in third and fourth 'preserve' is not correct word for the context. In last result, 'best' word can have much more criteria to be decided for the semantic representation of the sentence. So sentence embedddings generates correct score as compared to the word embeddings.

References

[1] Isidoros Perikos and Ioannis Hatzilygeroudis. 2016. A Methodology for Generating Natural Language Paraphrases, 7th International Conference.
[2] Chris Quirk, Chris Brockett and William Dolan. 2004. Monolingual Machine Translation for Paraphrase Generation, Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing.
[3] Jonathan Mallinson, Rico Sennrich and Mirella Lapata. 2017. Paraphrasing Revisited with Neural Machine Translation, Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers
[4] Ankush Gupta, Arvind Agarwal, Prawaan Singh and Piyush Rai. 2018. A Deep Generative Framework for Paraphrase Generation, Association for the Advancement of Artificial Intelligence.
[5] Ashwini Gadag and B M Sagar. 2016. A Review on Different Methods of Paraphrasing, International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT).
[6] Li Dong, Jonathan Mallinson, Siva Reddy and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.
[7] Tsung-Yi Lin Michael Maire Serge Belongie Lubomir Bourdev Ross Girshick James Hays Pietro Perona Deva Ramanan C. Lawrence Zitnick Piotr Dollar. 2014. Microsoft COCO: Common Objects in Context Springer International Publishing Switzerland 2014.
[8] John Wieting, Jonathan Mallinson, and Kevin Gimpel. 2017. Learning paraphrastic sentence embeddings from back-translated bitext. In Proceedings of EMNLP
[9] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014.Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems.
[10] Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz,R.; and Bengio, S. 2015. Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349.
[11] Aaditya Prakash, Sadid A. Hasan, Kathy Lee, Vivek Datla, Ashequl Qadir, Joey Liu, Oladimeji Farri. 2016. Neural paraphrase generation with stacked residual lstm networks: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers.
[12] Atish Pawar and Vijay Mago. 2018. Calculating the similarity between words and sentences using the lexical database and corpus statistics:ArXiv 2018.
[13] Alexis Conneau, Douwe Kiela, Loic Barrault, Holger Schwenk,and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data:conneau-EtAl:2017:EMNLP2017.
[14] Mikolov, Tomas and Grave, Edouard and Bojanowski, Piotr and Puhrsch, Christian and Joulin, Armand. 2018. Advances in Pre-Training Distributed Word Representations: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)