Customer Question Answering Chatbot Based on End-to-End Seq2Seq Mechanism

Zhao Zhang, Anyuan Xu, Siwei Zhao, Haoyu He
Khoury College of Computer Sciences
Northeastern University

Abstract

Chatbots, or dialog systems, are ubiquitous deployed in Customer service, Entertainment, Health care and Marketing nowadays, aiming to serve multiple purposes in different domains using Natural Language Generation (NLG). Semantic parsing models with applications in task oriented dialog systems require efficient sequence to sequence (seq2seq) architectures to be run on-device. In this project, we implemented from scratch and experimentally explored three encoder decoder RNN-based models [1] (LSTM, GRU and PRADO) on a customer support dataset which is publicly shared on Kaggle [2]. Moreover, to our knowledge, it is the first attempt to use the lightweight RNN-based models to build generative chatbots.

1 Introduction

Modern society is not imaginable without personal virtual assistants and conversational chatbots. There has been many exquisite chatbots such as, e.g., Siri, Alexa, Google assistant, Cortana; they are completely changing our communication habits and interactions with technology. However, Chatbot shares a long history which can trace back to 1950. Alan Turing proposed the concept of chatbot for the first time. Scientists were then rushing into this field and made significant attributions. Unfortunately, the development of chatbots has been constrained by the computation, memory capacity and architecture of model for a long time. Until recently, with the rise of natural language processing, the application of chatbots has made a huge leap. Specifically, The encoder-decoder seq2seq architecture.

In this project, we explore encoder-decoder seq2seq-based deep learning (DL) architecture solutions and using this architecture to train a generative chatbot on extremely large datasets containing nearly three million conversations between human and customer support chatbots. The first thing we did was data preprocessing in order to clean and tokenize data. Then we started from basic LSTMGRU as encoder and decoder to some variants such as and PRADO. After that we evaluated the performance of each model and compared that across the whole model set both intuitively and mathematically by comparing BLEU scores and other similarities metrics. Finally, we recommend as our best model based on our project and discussed some further works.

2 Background

Chatbot is a program that mimic human dialog through Nature Language Generation. The research on chat bots originated from the article "Computing Machinery and Intelligence" published by Alan M. Turing in "Mind" in 1950. The article started with "Can machines think?" machine can participate in an imitation game to verify whether the machine can “think” The earliest Chatbot ELIZA was built in 1966 and was developed by Joseph Weizenbaum. Since then, many and many Chatbots emerged in advantage of advanced computation and new theory. Nowadays, Deep learning models such as LSTM, GRU, have made great progress in the NLP field, including machine translation, speech recognition and Chatbots.

[1] Code and evaluation results can be found in https://github.com/Cli212/Lightweight-Language-Models-to-Generate
[2] https://www.kaggle.com/thoughtvector/customer-support-on-twitter

3 Dataset

The data used for training need to be well structured, which means each conversation has only one question and only one answers. we found the Customer Support on Twitter dataset fit this requirement and it also has large data volume with clear annotations. it has over 3 million tweets which takes up approximately 516Mb. Data structure is as following.

tweet id	author id	inbound	created at	text	response tweet id	in response to tweet id

4 Methodology

4.1 Data Preprocessing

Before the raw data(question and answer text) can be used for models, data prepossessing is necessary. The main purpose for data prepossessing is to filter out unnecessary information or noise in text data and remove rows that are will not be helpful for our study.

Language Detect: In this project only English is our target language for training and predicting, therefore languagedetect library is used to remove other languages except English.
Nested Questions: To have better question and answer text quality and also reduce data set size, only question is asked first and its corresponding reply would be kept and other text would be removed.
Duplicated Questions: Some questions are asked more than once, it causes our data set has duplicated queries and made the training become hard. The resolution is to only keep one question per duplication and its answer.
Clean Tweeter Text: Two libraries re and emoji are used to remove special characters, http links, emojis in all texts, as those are noise in our data.
Lemmalization: Lemmatize each word in original text to simplify vocab. Create new text after lemmatization. WordNetLemmatizer is used in this step.
Byte Pair Encoding(BPE): Initially traditional tokenize method is used to split sentences into words. However, the resulting vocab size is too large(100k+) and will cause difficulty for later training, therefore a BPE model which is pretrained on Wikipedia is introduced and it can reduce the vocab size to lower than 10k.

After all those steps, we get BPE tokens for each sentence. Counter is used to extract all words and each word's frequency and establish vocab. All tokens are converted into index based index of words in vocab. Each sentence will not just be represented by tokens, but also a list of index. The final output files of data prepossessing includes 2 files, one for vocab(vocab.txt), one for index after converted tokens into index given vocab for all questions and answers.(index.csv). Those two are input data for Seq2Seq models in later steps.

4.2 Seq2Seq

Sequence to sequence (seq2seq) is one of the main approaches that are ubiquitously used in the applications of NLG. Language models (LM) stemming from deep neural networks such as skip-gram, recurrent neural network (RNN), long-short term memory (LSTM) network are efficient in representing semantic information. Hence they are popularly used as encoders and decoders in seq2seq architectures. Take a sequence of words as input, a LM based seq2seq model converts word tokens into a fixed dense vector and then the decoder generate an sequence of tokens based on a target vocabulary. Semantic representation models with applications in task-oriented dialogue systems require efficient sequence to sequence (seq2seq) architectures to be run on-device. In this project, we build the seq2seq pipeline with lightweight RNN variants — LSTM, GRU, and PRADO — as encoders and GRU as decoder augmented by Attention Mechanism [2].

4.2.1 LSTM & GRU

A recurrent nueral network (RNN) model calculates a vector named hiddenstate h_n by taking a sequence of words $w_1, w_2, ..., w_L$:

$$h_n = f(h_{n-1}, w_n), n \in [1, L], h_0 = 0,$$

where L is the length of sentence and $f(\cdot, \cdot)$ denotes a parametrized non-linear function, such as sigmoid, hyperbolic tangent, long-short term memory (LSTM) and gate recurrent unit (GRU). A vanilla RNN model with sigmoid as f function is prone to lose contextual information especially when the sequence long, LSTM and GRU are the solutions to handle this phenomenon. Both of them are using internal weights as gates to

learn to remember or discard information in a word-by-word context. Take LSTM as an example, the equations of it can be summarized as follows:

$$
\begin{aligned}
i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\
f_t &= \sigma(W_f[h_{t-1}, x_t] + b_f) \\
\widetilde{C_t} &= tanh(W_c[h_{t-1}, x_t] + b_c) \\
o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\
h_t &= tanh(f_t \cdot C_{t-1} + i_t \cdot \widetilde{C_t}) \cdot o_t
\end{aligned}
$$

where W_i, W_f, W_c, W_o are learnable weights defined above and C_t is the cell state. By using these learnable weights as gates, the model can prevent lose information when the sequence is long.

4.2.2 PRADO

PRADO is a projection-based embedding-free neural encoder that is proved to be effective for natural language processing tasks significantly perform LSTM models. It uses a projected embedding layer to substitute normal embedding layer in RNN-based models to build the word encoder. Denote the number of words in the vocabulary as V, instead of mapping every word w_i to a fixed d-dimension vectors by a large parameter $W \in \mathbb{R}^{d \cdot V}$, PRADO first fingerprints words and use a hashing function to extract B bit features then map the two-bit sequence to a set $-1, 0, 1$, resulting in a vector $v_i \in \{-1, 0, 1\}^B$. B is a hyperparameter that we can set for the vector size. This projection method allows us to generate compact embeddings with lower memory fingerprints and is proved to outperform LSTM, GRU a lot in text classification tasks.

Despite the proficiency in text classification and tagging tasks [4], few work has studied whether this excellence can persist when extending PRADO to language generation tasks.. In our project, we implement PRADO from scratch and apply it as an encoder in the seq2seq architecture.

4.2.3 Attention Mechanism

Between the encoder and decoder, we resort to the attention mechanism that allows the decoder to pay attention to certain parts of the input sequence, rather than using the entire fixed context at every step. We use the General Attention proposed in [2] calculate attention weights, or energies, using all of the encoder's hidden states and the hidden state of the decoder from the current time step only. There are three score functions can be used to calculate the attention scores:

$$
\text{score}(h_t, \overline{h_s}) = \begin{cases} h_t^T \overline{h}_s & dot \\ h_t^T W_a \overline{h}_s & general \\ v_a^T tanh(W_a[h_t; \overline{h}_s]) & concat \end{cases}
$$

where h_t is current target decoder state and $\overline{h}_s$ is all encoder states.

4.3 Evaluation

In this study, BLEU is used as basic evaluation metric to evaluate the performance of the Chatbot.

The evaluation metric BiLingual Evaluation Understudy (BLEU) is a score in range of [0, 1] that measures the quality of the language model, the higher BLEU score indicates that the model has a better performance. BLEU is a metric that has high correlation with the quality judgement of a real human [1]. The BLEU is defined as:

$$
BLEU = min(1, exp(1 - \frac{reference - length}{output - length})(\prod_{i=1}^{4} precision_i)^{\frac{1}{4}}
$$

$$
precision_i = \frac{\sum_{snt \in Cand-corpus} \sum_{i \in snt} min(m^i_{cand}, m^i_{ref})}{w^t_i = \sum_{snt' \in Cand-corpus} \sum_{i' \in snt'} m^{i'}_{cand}}
$$

where

m^i_{cand} is the count of i-gram in candidate matching the reference translation

m_{ref}^{i} is the count of i-gram in the reference translation

w_{t}^{i} is the total number of i-grams in candidate translation

For in-depth analysis of our results, in addition to using BLEU which is word-overlap based metric, we further evaluate our model using qualitive metrics such as Embedding Average Cosine Similarity (EACS), Greedy Matching Score (GMS) from [5], the GMS is computed as following:

$$GM(C, r) = \frac{G(C, r) + G(r, C)}{2}$$

$$G(C, r) = \frac{\sum w \in Cmax_{\hat{w} \in r} cos\_sim(e_w, w_{\hat{w}})}{|C|}$$

where

C is the candidate sentence

r is the reference sentence

$cos\_sim()$ is the cosine similarity function

EACS and GMS are unsupervised metrics in task-oriented dialogue and evaluate model based cosine similarities between words that fits our experimental setting better. These metrics resort to public word vectors GloVe [3] to catch the high-dimension semantic representation of sentences and calculate the similarity of representations extracted from predicted sentences and references.

5 Experimental Results

Due to the high time cost of training due to the large amount of data (800K pieces), we only use the GRU and PRADO to conduct experiments and randomly pick 400K pieces as the training and evaluation data in a 8:2 train test split. As for the hyperparameters, we keep them the same for the two models for a fair comparison. Embedding size (d in PRADO) as 128 and hidden (b in PRADO) size as 256, max sequence length is fixed as 32 and the attention mechanism is 'general' method in both architectures. To train the model, we keep the batch size as 64 and train 30 epochs on training data then evaluate them on test data. Results can be found in Figure 1. After the training, we will automatically save the parameters and checkpoints so we can load them in user command evaluation. Figure 2 is an example of chatting with the trained bot in command line.

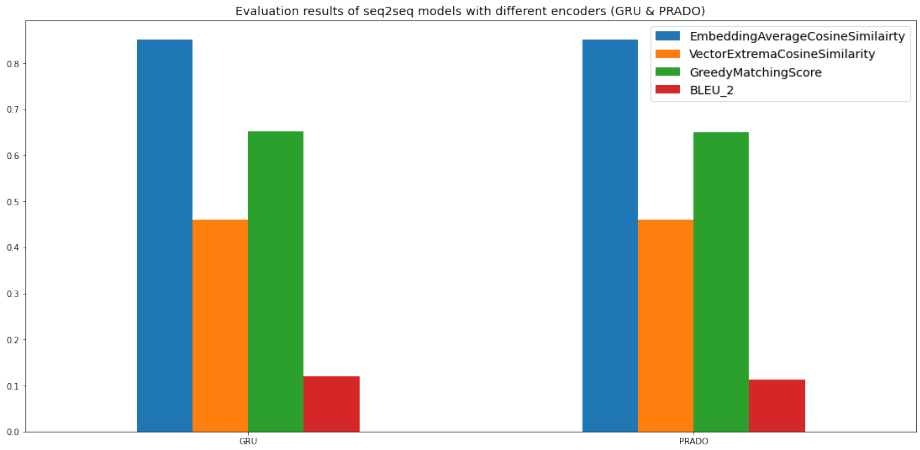

Figure 1: Figure 1 shows the results of evaluation metrics between the GRU and PRADO models, including Embedding Average Cosine Similarity, Vector Extrema cosine similarity Greedy Matching Score, and BLEU score. We can see that the results are very similar between the two encoders. Evaluation results of seq2seq models with different encoders (GRU & PRADO). Their results are close, stating that the projection-based PRADO does not outperform GRU in language generation task as it does in classification tasks. And the PRADO-based seq2seq model is smaller in size than GRU-based seq2seq model (93M vs. 112M).

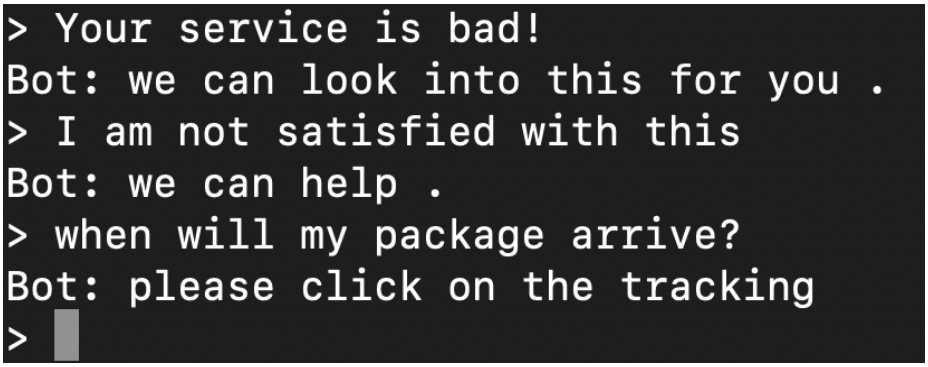

Figure 2: The user interface of the chatbot, loading parameters from saved training checkpoints. The bot can answer some basic customer questions, and it sounds like a human

6 Discussion and Future Work

In this project, we only tried RNN based model, LSTM and PQRNN, other classical models which could also have good performance have not been tested such as attention and transformers. For evaluation, we simplly used BLEU as basic evaluation metric combined with other metrics, but BLEU only reflects common word frequency and may ignore context. Others popular metrics like ROUGE and METEOR can be used to provide more information about the relationship between predicted response and actual response. The performance of our models is also not that ideal, repetition words are frequently seen and some combinations of tokens does not really make sense or really make a sentence. Especially when BPE is using, some combination of tokens even can not make a word. The policy of choosing next word and current state tracking should be reinforced in future experiments to produce better predicted responses.

References

[1] Chris Callison-Burch, Miles Osborne, and Philipp Koehn. Re-evaluating the role of Bleu in machine translation research. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 249–256, Trento, Italy, April 2006. Association for Computational Linguistics.

[2] Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics.

[3] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[4] Aditya Siddhant Prabhu Kaliamoorthi, Edward Li, and Melvin Johnson. Distilling large language models into tiny and effective students using pqrnn. *CoRR, abs/2101.08890*, 2021.

[5] Shikhar Sharma, Layla El Asri, Hannes Schulz, and Jeremie Zumer. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *arXiv preprint arXiv:1706.09799*, 2017.