Chatbot

Kaijian Li, Yang Li, Yuchen Fang

December 27, 2018

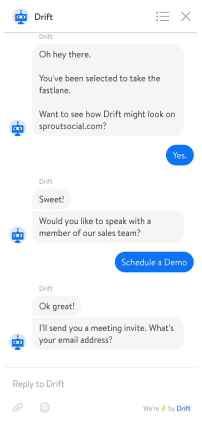
Abstract

Today chatbot is comming into our life. So our group want to build a chatbot. We download data from internet and build a complex model. The final chatbot do works.

1 Introduction

Chatbots is a bot which can chat with you like a real human. Since today people are becoming more lonely, chatbots can help these people to get through the lonely time. A search shows in China there are more than 20,000,000 young people between 20 to 39 years old who stay alone. This situation is a big problem now. We think chatbots can help with it. What's

(a) pic1.



Drift

Drift

Oh hey there.

You've been selected to take the fastlane.

Want to see how Drift might look on sproutsocial.com?

Yes.

Drift

Sweet!

Would you like to speak with a member of our sales team?

Schedule a Demo

Drift

Ok great!

I'll send you a meeting invite. What's your email address?

Reply to Drift

We're ⚡ by Drift

(b) pic2.

Figure 1: chatbot in our lifes

more, today more and more chatbots are coming into our lifes. In Figure 1(a) this is a chatbot from Microsoft. Her name is Xiao Bing. Figure 1(b) is a chatbot for lane business. This chatbot can automatically sale the products to customers following the desided steps.

Since chatbots are useful in real life, our group wants to build a simple chinese chatbot which can chat with human. We combined with rule based method and neural method. Finally the chatbot which we made do worked.

2 Dataset

2.1 Dataset

We collect a lot of corpus from the internet, which is showed in the Table 1. The formation of the dataset is showed in Table 2. There are many asks and answers and they form the ask-ans pairs. The quality of these data are different. In the project we choose qingyun, tieba and xiaohuangji datasets to train our model. Qingyun dataset is some ask-ans pairs from a chat group. It is close to daily life. Tieba dataset is from baidu teiba. It's large but contains a lot of noises. Hoever we still use it, because we check the ask-ans pairs in tieba. Most of them are meaningful and we also do some filters on it. Finally xiaohuangji dataset is from a xiaohuangji chatbot. The problem with this dataset is it contains a lot of indecent words. But we will check these words by hand-made rules. So we think this dataset can be used in our project.

Name	douban	ptt	qingyun	subtitle	tiba	weibo	xiaohuangji
size	352w	40w	10w	274w	232w	443w	45w

Table 1: Datasets

Ask	Ans
你踢足球吗	我不知道怎么玩
谁是你妈妈	其实我没有妈妈
谁是最好的足球 PLAYER	马拉多纳是伟大的. Sinsemillia 甚至更好.

Table 2: Data Examples

2.2 Pre-Process

Before we build our model, we need do some pre-process to our data. There are three parts: word segmentation, data clean and building vocabulary.

Word Segmentation We use jieba to segment sentence. jieba is a free and powerful Chinese segmentation package.

Data Clean First we need to remove the sentences which are too long. We set the max length as 30 words. Because we just want to build a simple chatbot, people won't chat with others in this way. Second step is to remove the special characters. These are noises in the dataset, so we need delete them before training.

Vocabulary Finally we using the pre-precessed dataset to build the vocabulary. And we choose the most common 50,000 words as our vocabulary. Also we need to add smome specail words into the vocabulary, including <PAD>, <UNK>, <GO>, <EOS>. <PAD> is used for padding. <UNK> means the unknow words. <GO> is the start mark and <EOS> is the end mark.

3 Rule Match

3.1 Function

Rule Match part is the first processing operation on users' queries. We set up many rules to match the users' queries type and provide responses to some basic queries, such as the response " 记得带伞哟！ " to query " 今天天气怎么样？ ". In this processing part, we answer a lot of simple and basic inquiries, and those that are more difficult to answer, outside of our pre-established rules, will be processed by subsequent processing part, such as retrieval, seq2seq, etc.

The Rule Match is carried out according to the following parts, which is shown in Figure2. Firstly, it checks if a user is asking the same question repeatedly. Then it will check whether the query contains sensitive words. Finally, if the query pass all examination, the Rule Match will use rule patterns to match the query, and determine if there is a suitable response to in the rule base. If all above checks cannot provide a proper response, the query will be sent to next processing part, Retrieval.

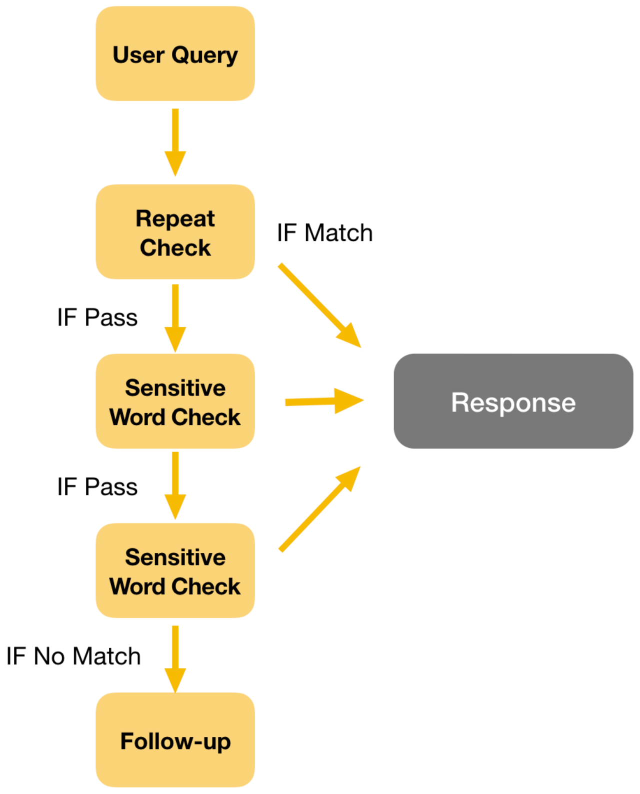


Figure 2: Rule Match Workflow

3.2 Repeat Check

In the Rule Match processing workflow, we check repeat first. Based on different user IDs, it creates temporary records to hold historical conversations. If it detects that the same user has continuously entered the same query, it will randomly pick a reply from the "Repeat Dialog Response".

For example, if the user ask " 今天天气怎么样?" for many times, it will answer " 怕你淋雨，记得带伞" at the first time, and response " 呜呜呜 换个话题好不好", " 额……重复这么多遍，是不是有陷阱 我才不上当呢", " 你这样对待我我会伤心的，我们换个话题好吗" to the subsequent iterations. Repeated detection makes the chat robot's response more natural and closer to the person's real reply.

3.3 Sensitive Word check

We have created a list of sensitive words, most of which are sensitive words such as pornography, politics, and abusiveness. We perform string matching of users' queries without word segmentation. When the user question contains these words, we randomly select the reply as the return value from the "sensitive word reply base". These responses contain " 房间里的人听着，你已经被锁定位置了", " 截图存证。我未参与以上聊天" ," 这个不能聊，休想陷害我", etc. Sensitive word detection ensures that our design is humanized and does not violate relevant laws and regulations.

3.4 Rule Base Match

Rule Base Match is the key part of Rule Match processing. When queries pass the previous check, the Rule Base Match will search the response in Rule Base according to rule patterns match. We summed up a large number of fixed-mode responses from the online text. For example, the query of "unhappy" has following responses "Can I help you?", "What happened?" "I am a little unhappy when I hear you say this. I really hope that I can Help you...", etc. These kinds of responses are chosen in the Rule Base and if the query contains the corresponding key words.

There are two types of Rule Pattern Match . The first mode is a single set match. For example, we have a set of key words: (' 不开心', ' 不高兴', ' 不愉快', ' 心情不好'). If our question contains any of these keywords, we will return the corresponding response: (' 我可以帮助你吗', ' 发生了什么吗', ' 听你这么说我也有点不开心, 真希望能够帮到你什么...', etc.). Another mode is multiple set match. For example, we have 2 sets of key word: ([' 天气', ' 气候'],[' 怎么样']). In this mode, only when the query contains both keywords in the sets will we return the corresponding response: Only when the question contains both keywords in the collection will we return the corresponding response: (' 你要问我天气啊，自己看窗外呀！', ' 天有不测风云，我也不知道…', etc.)

In this section, questions about the fixed pattern will be answered. More complex issues will be processed later.

4 Retrieval

4.1 Function

In the search processing section, we use the Xiaohuangji dialogue data, which includes a lot of dialogue. We use problem similarity matching. If the user's question is highly similar to one of the conversation data, we will use the corresponding answer.

Firstly, we calculate the similarity between the users' queries and queries in Xiaohuangji dialogue data. If the similarity exceeds the threshold, where we set it as 0.8, the corresponding response will be return. Otherwise, the query will be sent to seq2seq to get response. The workflow is shown in Figure3.

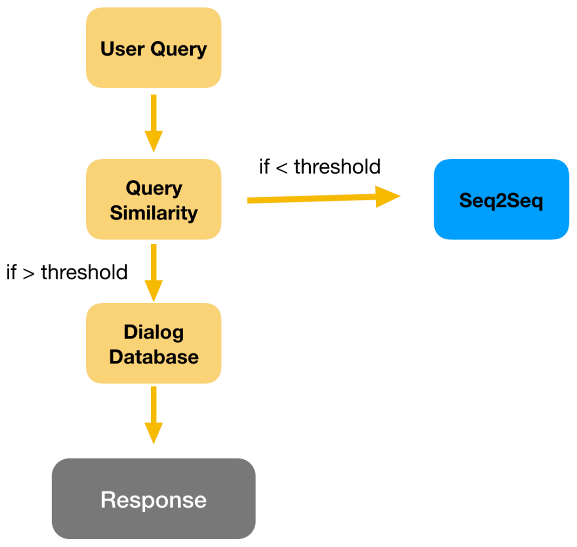


Figure 3: Retrieval

4.2 Similarity Calculation

4.2.1 Experiment set-up

We first collect xiaohuangji dialog data, separate queries and answers, use jieba word segmentation to process them, and store them in two files in the corresponding order.

Then, we use pre-trained word embedding as the word representation, where we use sgns.zhihu.bigram-char as word embedding.

4.2.2 TF-IDF

TF-IDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. The tf—idf value increases proportionally to the number of times a word appears in the document and is offset by the number of documents in the corpus that contain the word, which helps to adjust for the fact that some words appear more frequently in general.

In the case of the term frequency tf(t,d), the simplest choice is to use the raw count of a term in a document, i.e., the number of times that term t occurs in document d. If we denote the raw count by $f_{t,d}$, then the simplest tf scheme is tf(t,d) = $f_{t,d}$.

The inverse document frequency is a measure of how much information the word provides, i.e., if it's common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word (obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient):

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

with

N: total number of documents in the corpus N = |D|

$|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\mathrm{tf}(t, d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

Then TF-idf is calculated as

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \cdot \mathrm{idf}(t, D)$$

4.2.3 Calculation

In this part, we use word embedding to represent each word in query. And we use the weighted sum of word embedding as a representation of query, where the weight is the TF-idf weight. We do same to all queries in xiaohuangji dialog. Now we do vector product to get similarity. If the similarity is larger than our threshold, o.8, we use the corresponding response as return value. Otherwise, the query will be sent to seq2seq. It is worth noting that if the quality of the response generated by seq2seq is very low, then we still take the result of the result as the return value.

5 Generative Model

If the program can't find an answer in the former steps, it will use a generative model to generate an answer. In this project we choose seq2seq model [1], which is widely used in nlp tasks, such as representation model, word-sense-disambiguation and so on. In order to improve the seq2seq model, we also use attention mechanism and DPGAN.

5.1 Seq2seq

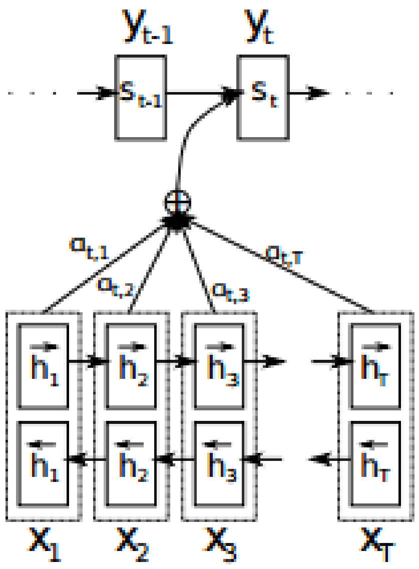


Figure 4: seq2seq model

Figure 2 is our seq2seq model. In the encoder part we use bidirectional lstm. We think the relations between the words in the queries is directional. And bidirectional lstm can extract the bidirectional relations between the words. The input of the encoder is the embedding of the words. We tried pre-trained word embedding. But the result is not good. Finally we randomly initialize the embedding and update it during training.

The decoder part we use attention mechanism and rnn decoder. Because attention mechanism can capture the long term dependencies compared with traditional seq2seq model. To choose the final answer, we use beam search, which can speed up the search process. Finally we use dropout to protect from over-fitting, which is performed well in deep learning.

5.2 DPGAN

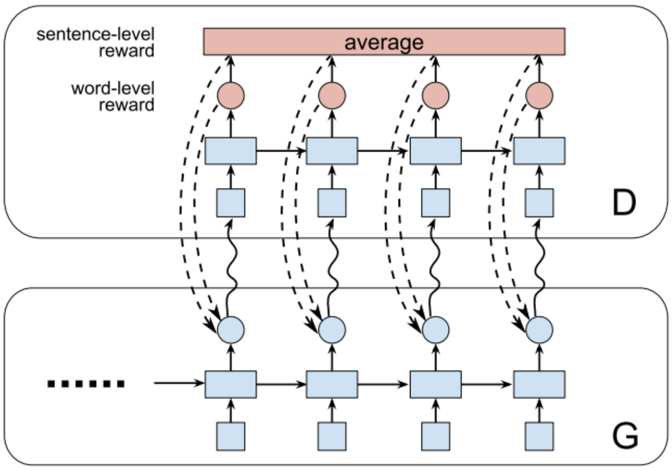


Figure 5: DPGAN

GAN is widely use in machine learning tasks. In this project we try to use DPGAN [2] to improve the seq2seq model. In DPGAN there are a generator and a discriminator. This difference between DPGAN and traditional GAN is that the discriminator is built on a unidirectional LSTM. And use the ouptut of the language model, cross-entropy, as the reward. Specifically, given a sentence y_t, the cross-entropy based reward for the k^{th} word is calculated as

$$R(y_t, k) = -logD_{\Theta}(y_{t,k}|y_t, < k) \tag{1}$$

Besides word-level reward, it also uses sentence-level reward

$$R(y_t) = -\frac{1}{K}\sum_{k=1}^{K} logD_{\Theta}(y_{t,k}|y_t, < k) \tag{2}$$

The program will first train the generator. Then it uses this generator to generate some samples to train the discriminator along the the original training data. Then using confrontation training: 1.using the results from discriminator and the original data to train the generator. 2. using the results from the generator and the original data to train the discriminator. Then repeat processes 1 and 2.

6 Experiments

We train our model and compare our model with Microsoft Xiao Bing. Here is the details of our experiments.

6.1 Implementation Details

We use jieba to all the Chinese sentence including the dataset and the queries. The size of the vocabulary is 50,000. The dimension of the hiddenstates fo the encoder and the decoder are both 300. The dropout keep probability is 0.7. The width of beam search is 3.Learning rate is 0.0001. The batch size is 200. The number of units in the LSTM cell is 256.

6.2 Competitor Models

We use Microsoft Xiao Bing as our competitor model. You can access it by wechat. Since Xiao Bing has many functions, such as playing small games, we will just compete the text chat function.

6.3 Evaluation Methods

We adopt human judgments to compare the performance of different models. Human will judge whether the answers are neutral and meaningful. The meaningful part is judged from the following five criteria:

(a) Grammar and Fluency: Responses should be natural language and free of any fluency or grammatical errors;

(b) Logic Consistency: Responses should be logically consistent with the test ask;

(c) Semantic Relevance: Responses should be semantically relevant to the test ask;

(d) Scenario Dependence: Responses can depend on a specific scenario but should not contradict the first three criteria;

(e) Generality: Responses can be general but should not contradict the first three criteria;

7 Results

Ask	Ans-ours	Ans-Xiao Bing
你会赌博嘛	开门，查水表	既然不是赌神周润发，就别指望靠它发财了
今天天气怎么样	空气不错，好开心	你在哪里啊
晚上玩不玩游戏	认真读书，天天向上	没关系的，我们可以聊别的嘛
三鹿奶粉好喝嘛	后妈的明智之选	也许味道还不错，但过量的话后果很严重...
你有女朋友嘛	没有	不忍心和剩男抢妹子哈哈

Table 3: results

Here are the results. in the results we can see, our model can answer some simple question. Hoever the Xiao Bing's answer are more humanize. For most simple question, our model's answer is meaningful and neutral. But if we ask a complex question like : " 我好担心我的父亲，他生病住院了", the answer is " 我也是", which is not meaningful. But in general our model is useful.

8 Conclusion

Our model doesn't perform well as us expected. We think there are several reasons: 1. the data is small and contains lots of noises. We just use a part of our data to train due to the time reason. 2. The model is too simple. We just use a simple seq2seq model. And the DPGAN perform bad. Maybe we can use a more complex model to capture more information. 3. The model doesn't converge. Because we are lack time and GPU sources, this model just trained half day.

References

[1] Wu Y, Schuster M, Chen Z, et al. Google's neural machine translation system: Bridging the gap between human and machine translation[J]. arXiv preprint arXiv:1609.08144, 2016

[2] Jingjing Xu, Xuancheng Ren, Junyang Lin, Xu Sun DP-GAN: Diversity-Promoting Generative Adversarial Network for Generating Informative and Diversified Text, EMNLP 2018