Creating a Chatbot for Three-Way Conversations

Amber Li [1] Emma Liu [2] Julia Wang [3] Kate Xu [4]

[1]amli@mit.edu, [2]emmaliu@mit.edu, [3]jwlw2022@mit.edu, [4]katexu@mit.edu

Abstract

A lot of natural language processing research features dialogue systems and chatbots that can synthesize two-way conversations. We took the conversations a step further to include three particpants. We trained a Seq2Seq model on the PERSONA-CHAT dataset [1] and adapted it to train our three-way chatbot. We improved our Seq2Seq model performance by adding attention and adjusting the embedding size.

Chatbots, also known as dialogue systems or conversational agents, are instrumental to language learning and entertainment. We want to build a chatbot that generates engaging, informative, and consistent dialogue in the chit-chat setting.

We started with a Seq2Seq model without attention, fine-tuned the embedding size and hidden layers, and later added attention.

Background

Corpus-Based Chatbots

Corpus-based chatbots typically leverage dialogue datasets to find appropriate responses [**?**], and they can be fun for random chit-chat and function as social robots. There are two main types of responses: response by information retrieval (IR) and response by generation. We focus on corpus-based response by generative models, which come up with new sentences by conditioning on dialogue history and generating responses one word at a time. It is known that IR-based chatbots can only mirror training data, and generation-based chatbots sometimes speak nonsense.

Zhang et. al. [1] improved dialogue generation using profile information about a conversation partner, called a "persona." They created the PERSONA-CHAT dataset, which includes five-sentence descriptions of profiles, for their goal of creating a more engaging chatbot. They trained Seq2Seq and Memory Networks to produce more personal, specific, consistent, and engaging responses than persona-free models. [**?**] has extended the hierarchical recurrent encoder-decoder neural network to the dialogue domain.

PERSONA-CHAT Dataset

The PERSONA-CHAT dataset is crowdsourced via Amazon Mechanical Turk. There are three stages of data collection:

1. Personas: crowdsourced 1,155 possible personas
2. Revised personas: additional rewritten sets of personas with related sentences that are rephrases, generalizations, or specializations
3. Persona chat: pair two Turkers and assign each a random persona and have them chat

Each persona consists of five sentences with a maximum of fifteen words per sentence. Revised personas are used to mitigate the problems of word overlap from previous datasets such as SQuAD. Humans might accidentally repeat profile information verbatim or nearly word-for-word, so the PERSONA-CHAT dataset uses rewritten sentences that have similar implications but different meanings.

Evaluation

Quantitative Metrics

We used validation perplexity to assess the quality of our Seq2Seq model training with and without attention. Perplexity is a simple, multifunctional, and powerful metric that can be used to evaluate not only language modeling, but also for any generative task that uses cross entropy loss, such as speech recognition and open-domain dialogue.

Methods, Models, & Experiments

Data Preprocessing

We created a modified PERSONA-CHAT dataset to adapt the data to conversation between three interlocutors. Each entry in our dataset has two consecutive sentences of dialogue from PERSONA-CHAT and a target, which is the sentence of dialogue immediately after the two sentences used as input. The output from the model is the response of the third interlocutor. The size of our modified dataset was 244,996 training sentences and 14,600 validation sentences.

Hyperparameters

For experiments, we fine-tuned number of layers (1 and 3) and embedding size (256 and 512). All experiments used a batch size of 128 and trained for 10 epochs.

Baseline Seq2Seq without Attention

We adapted the HW3 Seq2Seq code to use our modified PERSONA-CHAT dataset. The encoder and decoder models each consist of a GRU.

Seq2Seq with Attention

We also added Bahdanau Attention to the baseline Seq2Seq model [2]. This is an MLP-based attention module. There is a context vector c_i, computed for each word of the input sequence, that depends on the previous decoder hidden state s_{i-1}, as well as all the encoder hidden states $h_1, \cdots, h_M$. This context vector is used to compute each RNN hidden state:

$$s_i = f(s_{i-1}, y_{i-1}, c_i).$$

To get the context vector c_i, we compute a weighted sum of the h_i,

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j,$$

where the weights α_{ij} are computed by a softmax over attention energies e_{ij},

$$e_{ij} = a(s_{i-1}, h_j).$$

The e_{ij} are the result of an *alignment model* [2]. We use a linear layer and tanh activation.

Results

Our best performing model is Seq2Seq with attention, an embedding size of 512, and 1 layer.

Model Label	Seq2Seq Model	Embed Size	Num Layers	Val Perplexity
A	w/o Attention	256	1	18.694
B	w/o Attention	512	1	13.454
C	w/o Attention	512	3	31.352
D	Attention	256	1	17.278
E	Attention	512	1	9.423
F	Attention	512	3	23.972

Table 1. Validation Perplexities of Various Seq2Seq Models after 10 Epochs

Observe that validation perplexity is lower for higher embedding size and fewer layers. Larger embedding vectors learn more context, so they improve performance as shown by perplexity and examples. More layers increases the perplexity because there are more options that seem viable to the model. However, the example responses are comparable.

Examples

The source concatenates 2 consecutive responses (from 2 interlocutors) in a conversation from the PERSONA-CHAT dataset. The target represents the next response in the conversation from the dataset. The predictions represent the next responses in the conversation from our models.

Example 1

Source: "What is your favorite meat to eat? I would have to say its prime rib. Do you have any favorite foods?"
Target: "I like chicken or macaroni and cheese."
Predictions:

1. *Model A:* "I like to petition for a living."
2. *Model B:* "I like apples a lot. What is your favorite food?"
3. *Model C:* "I like to eat cuba libres."
4. *Model D:* "I like food, I eat a lot of meat. I like food and Mexican food."
5. *Model E:* "I love homemade coffee. I eat it every day."

Example 2

Source: "I agree. What do you do for a living? I'm a researcher I'm researching the fact that mermaids are real"
Target: "Interesting. I'm a website designer. Pretty much spend all my time on the computer."
Predictions:

1. *Model A:* "What do you do for a living?"
2. *Model B:* "Interesting. I'm a teacher. I spend most of my time playing video games."
3. *Model C:* "I am a writer. I am blogging."
4. *Model D:* "I am a musician and enjoy going fishing"
5. *Model E:* "I'm an accountant, but I bet that's an interesting job"

Discussion

In general, model A generates predictions with the same sentence structure as either the source or target sentence, but it does not understand context. Model A's response is unrelated to the source in Example 1, and it repeats the question from the source in Example 2. Meanwhile, models B, C, D, and E generate predictions that are relevant given the context. However, these models also tend to mimic the sentence structure of the source or target sentence rather than understand their generated responses. In Example 1, models C and E do not learn that cuba libre and coffee are drinks and cannot be eaten, and model D's response about liking food is repetitive. Model E's response to Example 2 even remarks that the previous interlocutor's job is interesting. Accordingly, we observed that the predictions using an embedding size of 512 are more reasonable than those using an embedding size of 256 in terms of relevance.

References

[1] Zhang et. al. "Personalizing Dialogue Agents: I have a dog, do you have pets too?" `https://arxiv.org/pdf/1801.07243.pdf`

[2] Bahdanau, Cho, and Bengio. "Neural Machine Translation by Jointly Learning to Align and Translate." `https://arxiv.org/pdf/1409.0473.pdf`

[3] Annotated Encoder Decoder. Github. `https://github.com/bastings/annotated_encoder_decoder/blob/master/annotated_encoder_decoder.ipynb`