Deep Learning For Dialogue Systems

Liangqun Lu, MS project

Abstract

The widely applicable interactive conversational agents require the development of intelligent dialogue systems. Natural language generation is critical in dialogue response generation and Recurrent Neural Networks (RNNs) including long short-term memory (LSTM) have been applied to tackle the task. The end-to-end sequence to sequence (seq2seq) models, in which an Encoder encodes input information and a Decoder generates output based on information encoding and language model, have demonstrated effectiveness in dialogue generation. Reinforcement learning implemented in seq2seq models rewards the conversation with informativity, coherence and ease of answering. Generative Adversarial Networks (GANs) that use a discriminative model to guide the training of the generative model have enjoyed considerable success in generating real-valued data. In this project, we built a LSTM seq2seq model for dialogue generation using pre-trained word embeddings. We applied the model on two public datasets movie dialogues and Reddit utterances, and evaluated the performance using metrics BiLingual Evaluation Understudy (BLEU) and Recall-Oriented Understudy for Gisting Evaluation (ROUGE). We also imported the model in Python Django web framework and provided online interactive data-driven dialogue generations.

Background

Dialogue systems, also known as interactive conversational agents, virtual agents and sometimes chatterbots, are used in a wide set of applications ranging from technical support services to language learning tools and entertainment. The reasonable spoken dialogue generation is believed to utilize the knowledge of speech recognition, language understanding (NLU), dialogue management and natural language generation (NLG) in order to obtain proper responses [1].

The current requests are 20% on task-oriented and 80% non-task-oriented from online shopping [2]. Task-oriented dialogue systems are developed to assist users to complete certain tasks, such as flight booking. The pipeline NLU and dialogue state tracking and NLG were widely used. Non-task-oriented dialogue systems are conversing on open domain such as chatbot. Usually a proper response is generated using generation model or information retrieval. Deep learning has the advantage of automatic feature learning and has been widely applied in areas computer vision and natural language processing (NLP). The recurrent neural network (RNNs) are one class of deep learning which can build sequential connection using neural nodes.

Word tokenization is fundamental for NLP, and previous methods have been developed to delineate word vectors, which are mainly categorized into word co-occurrence and deep learning modeling. Glove is a representative method for word embeddings [3]. Language model is to predict the next word for the given context words. RNN can perform next word generation without the limitation of word grams. Long short-term memory (LSTM) is one class of RNN, composed of a cell, an input gate, an output gate and a forget gate, were developed to deal with the exploding and vanishing gradient problems that can be encountered when training traditional RNNs.

Applying RNN models for robust and scalable dialogue systems is challenging, but there are achievements which make use of deep understanding, previous classic pipelines and and recent state-of-the-art work models. The LSTM sequence-to-sequence (Seq2Seq) model is one class of neural generation model that can solve the sentence input and output problems [4]. In the application of dialogue systems, the optimization is to maximize the probability of generating a response given the previous dialogue turn. The advantages of reinforcement learning (RL) and adversarial learning have also been applied to generate response generation [5] [6,7].

In this project, we propose to make utilize the Seq2Seq systems to learn response generations for dialogue systems. Experimental results show that the char-based seq2seq model and word-based seq2seq models can learn to produce some interactive responses while the output varieties are limited in the open-domain social media datasets.

Related Work

Previous studies of deep learning for dialogue systems have focused on task-oriented tasks and chatbot, which aims to converse on open domains naturally and consistently [2]. The most recent chatbot development is data-driven information retrieval [8] or response generation [9]. The information retrieval systems have the advantage of fluent responses as it selects the best candidate after a large dialogue repository search.

However, the response generation systems can learn the patterns from dialogues directly and apply the patterns to generate new responses.
Dialogue response generation can be phrased as the general source-to-target NLP tasks, such as language translation and text abstraction. The early seq2seq neural models shared the idea of neural machine translation(NMT) which made use of prior context information as well the neural network [9] [10]. These end-to-end seq2seq models encoders the information from input source first, and the encoder generates the output target using both encoded information and language model. The optimization of seq2seq models are similar to NMT so as to maximize the likelihood. This study [11] illustrated Maximum Mutual Information (MMI) can improve diversity and reduce dull responses. In the evaluation of dialogue systems, human judgement include the metrics Adequacy (correct meaning), Fluency (linguistic fluency), Readability (fluency in the dialogue context) and Variation (multiple realization for the same concept). In the viewpoint of computational metrics, word overlap metrics (BLUE and ROUGE) as well word embedding (vector extrema, greedy matching and embedding average) can be applied for evaluation [10]. Other reliable metrics include human rating and response classification [12]. However, there is a gap between human perception and automatic metrics.

The ideas of generative adversarial networks (GANs) and reinforcement learning (RL) have achieved success in deep learning tasks. In the generative adversarial networks, a generator is trained to generate outputs to fool the discriminator while the discriminator is trained to distinguish generated samples from real samples to the point that the generation cannot be separated from the real samples, which have been applied to image generation [13]. The discrete text generations at dialogue systems make the error from discriminator difficult to backpropagate to the generator. Reinforcement learning (RL) is one class of machine learning in which learning agents take actions in the given environment so as to maximize the cumulative reward.

In one study [5] of RL implemented in dialogue generation rewards the conversation with properties: informativity, coherence and ease of answering, with the advantages on diversity, length, better human judges and more interactive responses. The recent work seqGAN [6] used policy gradient reinforcement learning to update from the discriminative model to the generative model and demonstrates significant improvements in synthetic and real-world data.

Recent works show multi-turn context can facilitate the response generation [14] [14]. The transformer entirely based on attention showed novel sequence to sequence generation networks and the parallelization made the computation of attention fast in model training [15]. Another implementation of attentions in Deep Attention Matching Network showed state-of-the-art performance in Multi-Turn Response Selection [14].

In this project, we applied the widely studied LSTM seq2seq models on 2 social media datasets with character-level and word-level tokenization.

Dataset

In the experiments, we applied 2 common social dialogues from Movie and Reddit datasets in attempt to generate responses for social media. The Movie Dialog Corpus (https://www.kaggle.com/Cornell-University/movie-dialog-corpus/home) include 220,579 conversational exchanges between 10,292 pairs of 9,035 movie characters from 617 movies. In total, there are 304,713 utterances. The Reddit (https://www.reddit.com/) shares social news and forum where the content can be socially curated, promoted or commented by the site members. In this study, we downloaded comments from 2018 october and in total there are 8, 396, 812 conversational pairs.

Pre-processing and word embeddings

In the preparation of conversational pairs, we made sequential sentences in each dialogue from the movie characters or comments to the same original posts. In this case, the sentence in the middle can be output for one pair while input for another pair, which may create nonsense conversation pairs. Then for each dataset, we performed preprocessing before seq2seq models in the following steps.

1. Read Input and Target sentences
2. Split sentences into words and return strings # add space before periods
3. **Select sentence length within [1, 5]**
4. Lower all words
5. Split sentences into words (tokenize) by whitespace
6. Remove punctuation from all tokens
7. Filter out stop words
8. Remove non-English words from tokens
9. Tokenize sentences and padding full length

In the results, we obtained 14654 pairs from movie data. In order to make similar number as movie dataset, we considered the first 400000 pairs from Reddit and obtained 12154 pairs for model training.

Table 1: sample size for seq2seq training

Dataset	Total	#[1, 5]*	#training samples	#unique words	#unique characters
“Movie”	221282	14654	14654	4879	39
“Reddit”	400000*	12154	12154	7311	39

Model Architecture

In the model architecture, we considered LSTM seq2seq models on character-based as well as word-based modeling at each dataset. In the word-based models, we chose trainable word embeddings and the pre-trained word embedding Glove, in both of which 100 vector length is set.

We split the whole dataset to 80% training and 20% validation during the model training. We chose the optimizer ‘adam’ (learning rate = 0.001) with 'categorical_crossentropy' and used the metric 'acc' to optimize the model training. We also set regularization kernel_regularizer=regularizers.l2(0.01) and activity_regularizer=regularizers.l1(0.01)). The hidden layer neural size is 128 for char-based models while 256 for word-based models. For the sentence prediction, we applied inference model to select the optimal target.

In the evaluation, we used cosine similarity, BLUE and ROUGE to evaluate the generated sentences using the real targets as reference. As we know, BLUE is to calculate how much the words are predicted from the real targets. ROUGE is to calculate how much the words in the reference appearing in the predicted sentences. We implemented the whole process using Python deep learning model Keras.

Experiment Result

Model summary

In total, there are 6 seq2seq models for 2 datasets. In the model training, there are three input matrices as Encoder input, Decoder input and Decoder target, as shown in the table below. We also plotted the network graph for each model from each dataset in the following.

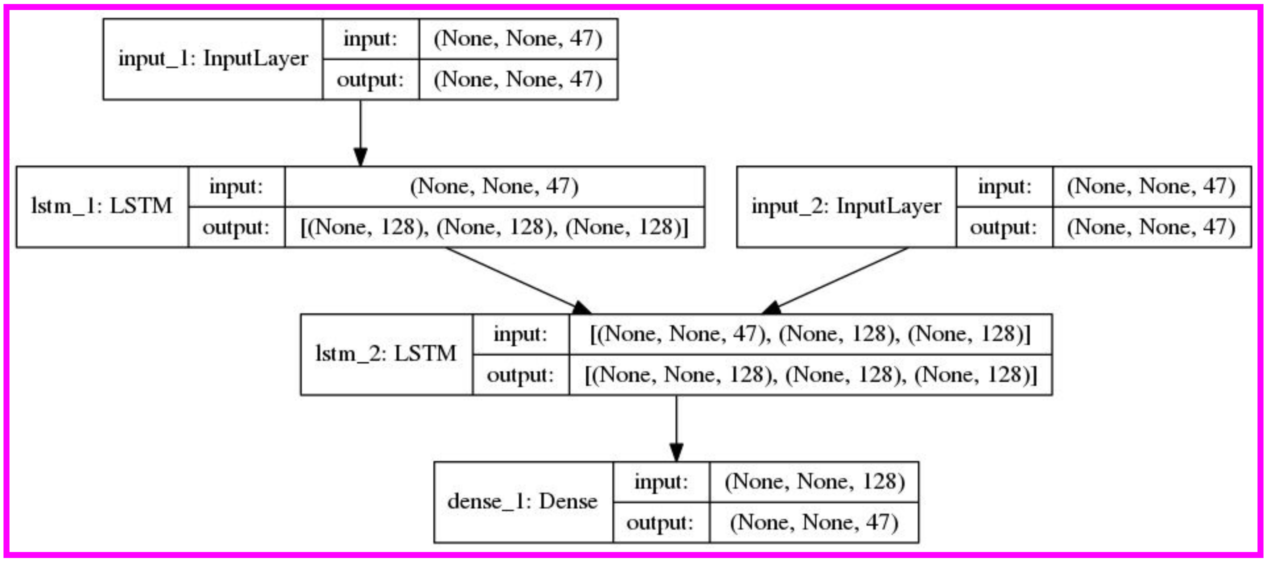
Table 2: input matrices summary for model training

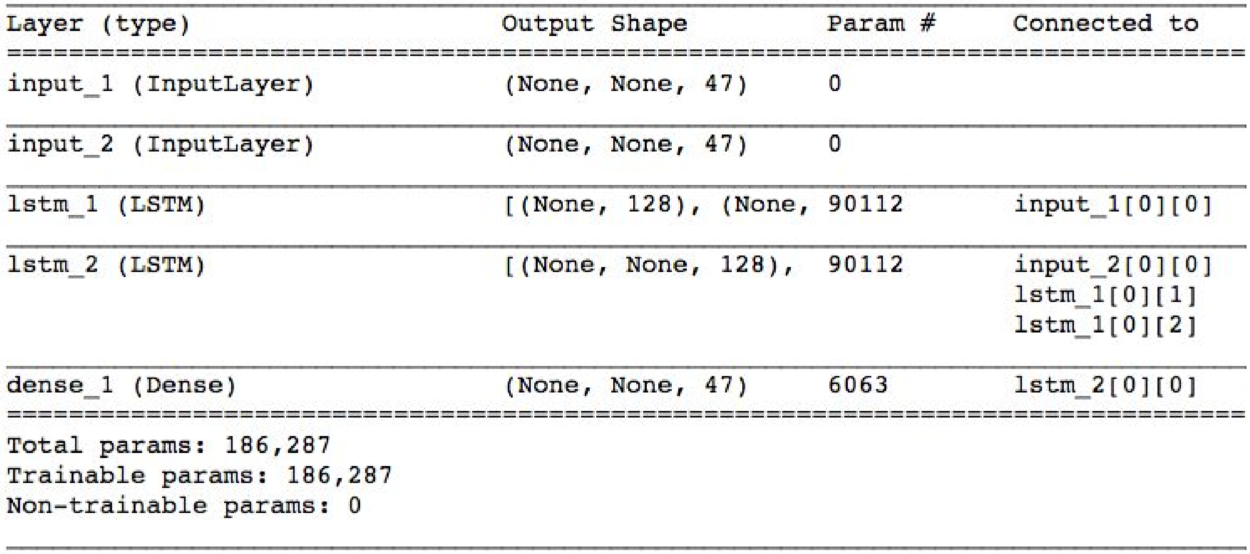
Dataset	Models	Encoder input	Decoder input	Decoder target	#Params
Movie	Seq2seq_char	(14654, 37, 39)	(14654, 37, 39)	(14654, 37, 39)	186,287
Movie	Seq2seq_word_auto	(14654, 7)	(14654, 7)	(14654, 7, 4879)	2,960,839
Movie	Seq2seq_word_pre	(14654, 7)	(14654, 7)	(14654, 7, 4879)	2,960,839*
Reddit	Seq2seq_char	(12154, 49, 37)	(12154, 49, 39)	(12154, 49, 39)	176,039
Reddit	Seq2seq_word_auto	(12154, 8)	(12154, 8)	(12154, 49, 7311)	4,072,263
Reddit	Seq2seq_word_pre	(12154, 8)	(12154, 8)	(12154, 49, 7311)	4,072,263*

*including non-trainable params

Model Architecture Graph and Summary

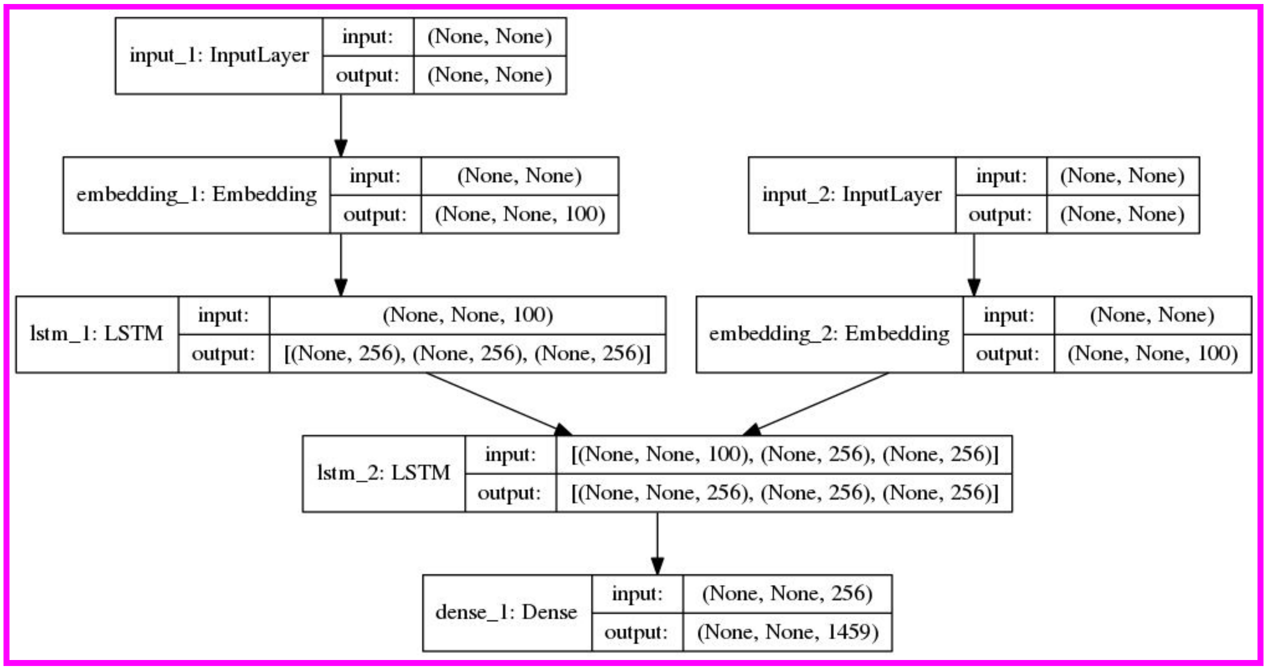
Movie - Seq2seq_char





```
Layer (type)                     Output Shape          Param #     Connected to
==================================================================================
input_1 (InputLayer)             (None, None, 47)      0

input_2 (InputLayer)             (None, None, 47)      0

lstm_1 (LSTM)                    [(None, 128), (None, 90112        input_1[0][0]

lstm_2 (LSTM)                    [(None, None, 128),   90112       input_2[0][0]
                                                                   lstm_1[0][1]
                                                                   lstm_1[0][2]

dense_1 (Dense)                  (None, None, 47)      6063        lstm_2[0][0]
==================================================================================
Total params: 186,287
Trainable params: 186,287
Non-trainable params: 0

```

Movie - Seq2seq_word_auto



input_1: InputLayer | input: (None, None) | output: (None, None)

embedding_1: Embedding | input: (None, None) | output: (None, None, 100)

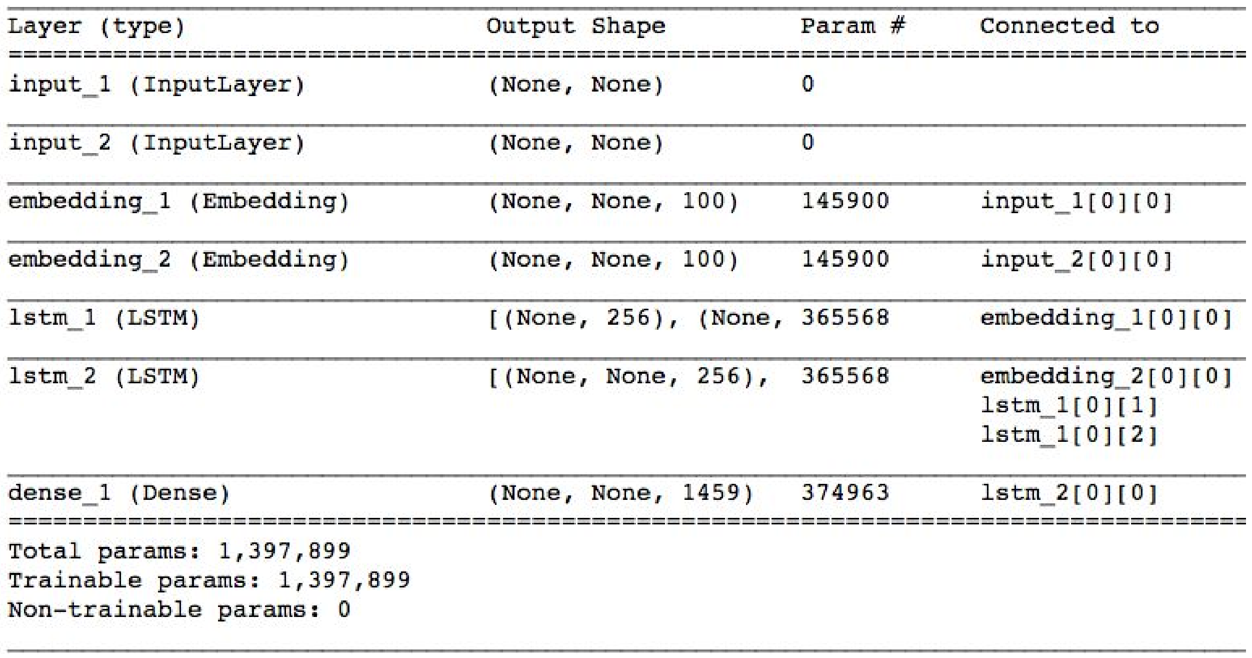
input_2: InputLayer | input: (None, None) | output: (None, None)

lstm_1: LSTM | input: (None, None, 100) | output: [(None, 256), (None, 256), (None, 256)]

embedding_2: Embedding | input: (None, None) | output: (None, None, 100)

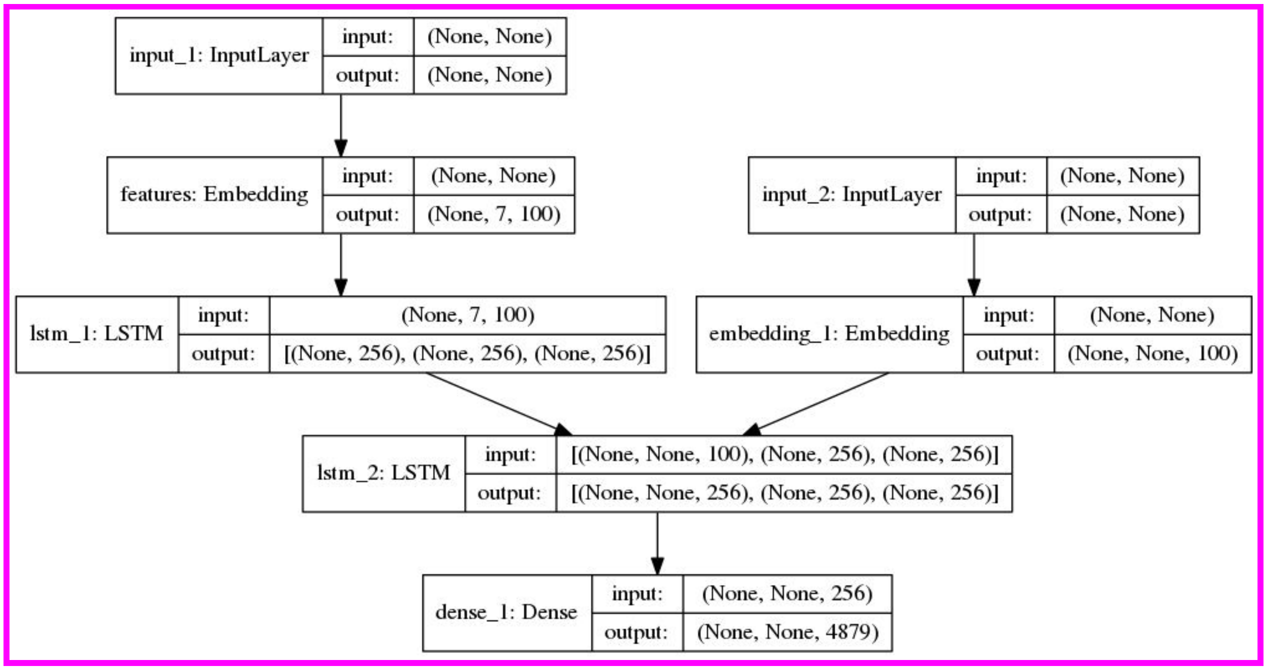
lstm_2: LSTM | input: [(None, None, 100), (None, 256), (None, 256)] | output: [(None, None, 256), (None, 256), (None, 256)]

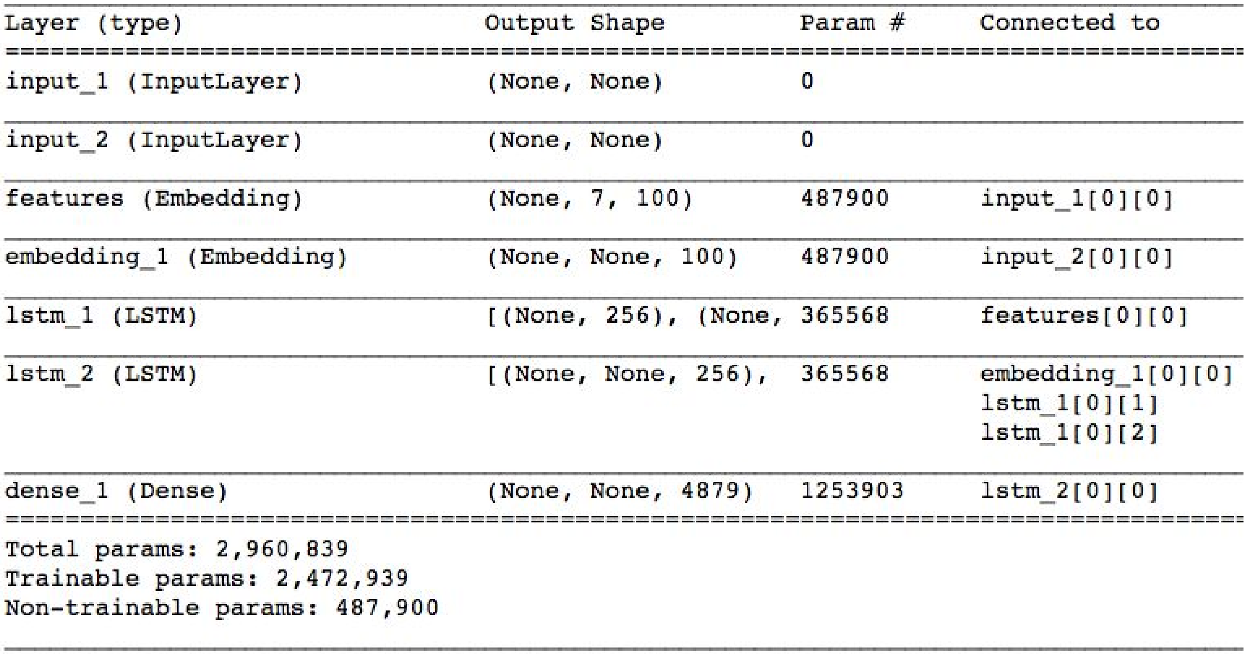
dense_1: Dense | input: (None, None, 256) | output: (None, None, 1459)



```
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, None)         0
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, None)         0
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 100)    145900      input_1[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, None, 100)    145900      input_2[0][0]
__________________________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 256), (None, 365568      embedding_1[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, None, 256),  365568      embedding_2[0][0]
                                                                 lstm_1[0][1]
                                                                 lstm_1[0][2]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, None, 1459)   374963      lstm_2[0][0]
==================================================================================================
Total params: 1,397,899
Trainable params: 1,397,899
Non-trainable params: 0
__________________________________________________________________________________________________
```

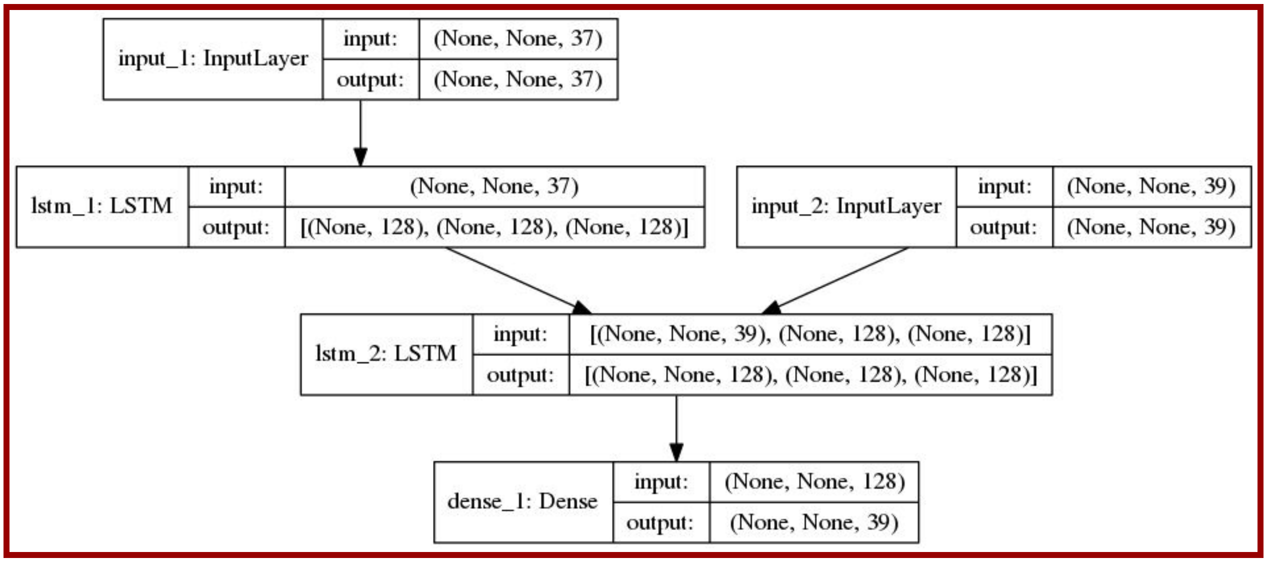
Movie - Seq2seq_word_pre

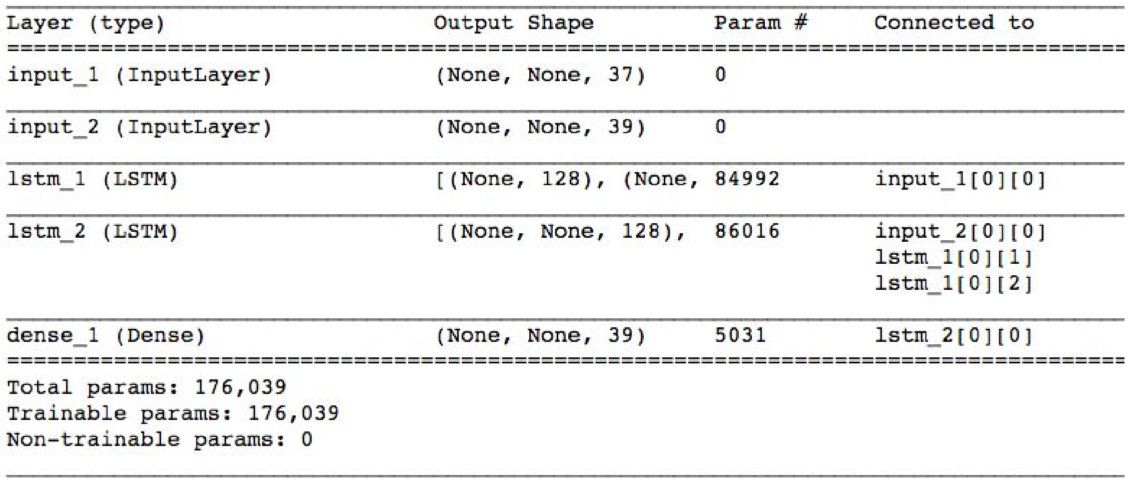




```
Layer (type)                    Output Shape          Param #     Connected to
==================================================================================
input_1 (InputLayer)            (None, None)          0
__________________________________________________________________________________
input_2 (InputLayer)            (None, None)          0
__________________________________________________________________________________
features (Embedding)            (None, 7, 100)        487900      input_1[0][0]
__________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 100)     487900      input_2[0][0]
__________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 256), (None,  365568      features[0][0]
__________________________________________________________________________________
lstm_2 (LSTM)                   [(None, None, 256),   365568      embedding_1[0][0]
                                                                  lstm_1[0][1]
                                                                  lstm_1[0][2]
__________________________________________________________________________________
dense_1 (Dense)                 (None, None, 4879)    1253903     lstm_2[0][0]
==================================================================================
Total params: 2,960,839
Trainable params: 2,472,939
Non-trainable params: 487,900
__________________________________________________________________________________
```

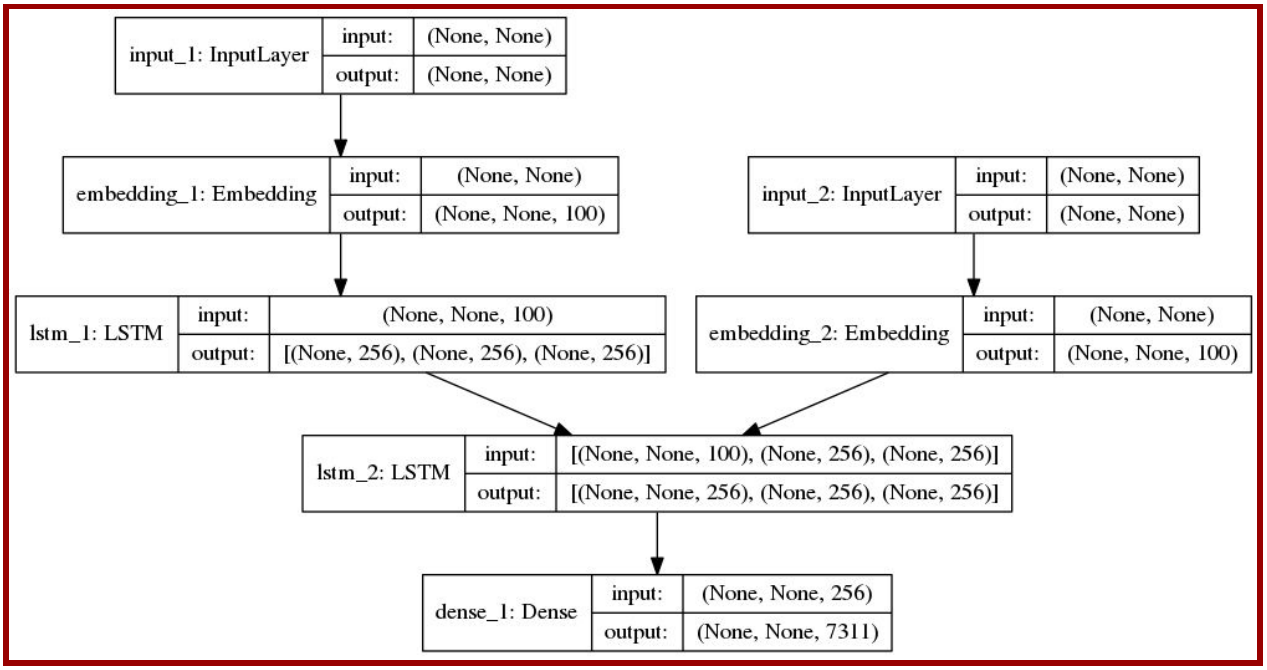
Reddit - Seq2seq_char

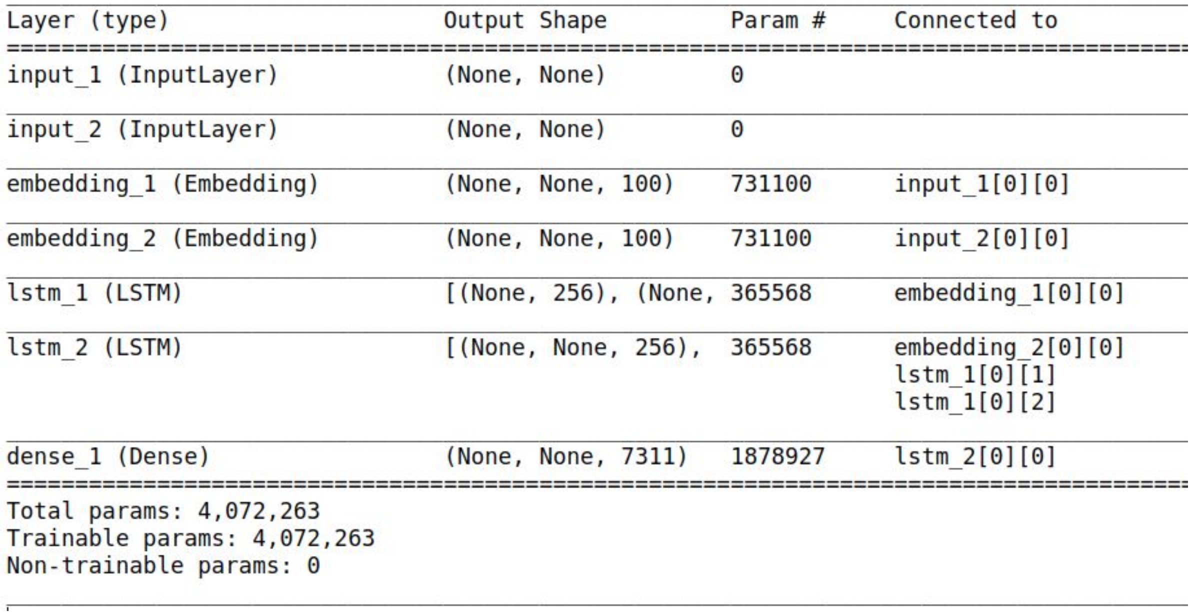




```
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
input_1 (InputLayer)             (None, None, 37)      0
____________________________________________________________________________________________________
input_2 (InputLayer)             (None, None, 39)      0
____________________________________________________________________________________________________
lstm_1 (LSTM)                    [(None, 128), (None,  84992       input_1[0][0]
____________________________________________________________________________________________________
lstm_2 (LSTM)                    [(None, None, 128),   86016       input_2[0][0]
                                                                   lstm_1[0][1]
                                                                   lstm_1[0][2]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, None, 39)      5031        lstm_2[0][0]
====================================================================================================
Total params: 176,039
Trainable params: 176,039
Non-trainable params: 0
____________________________________________________________________________________________________
```

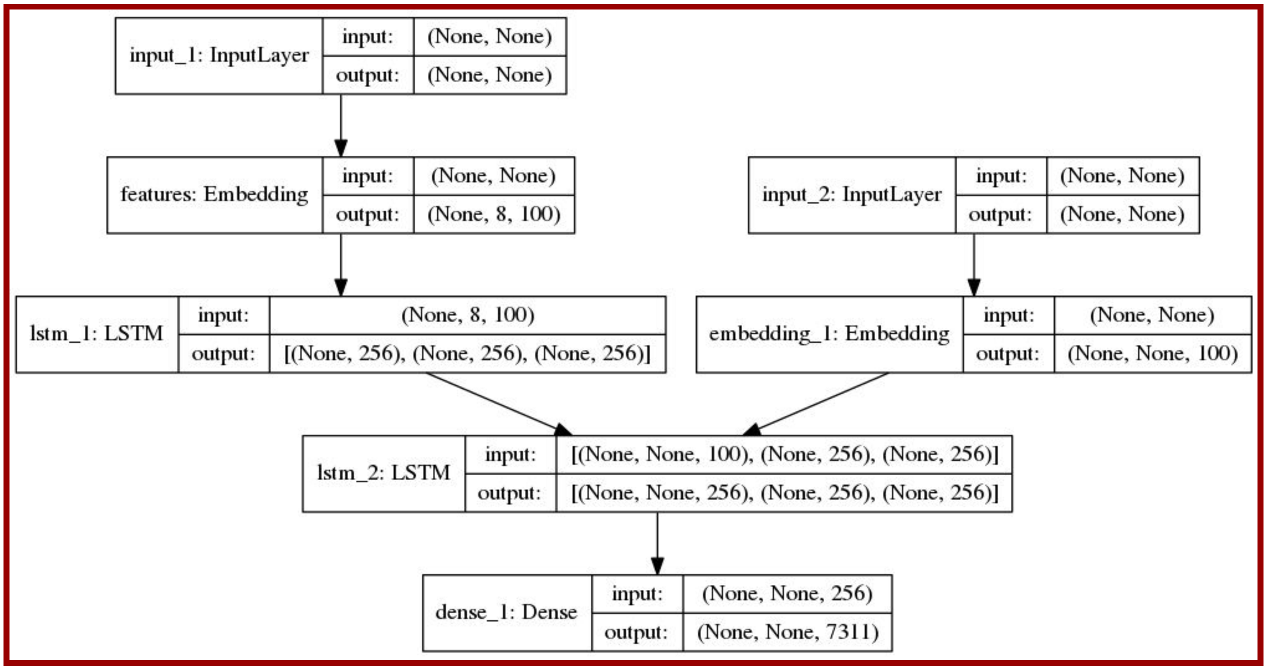
Reddit - Seq2seq_word_auto

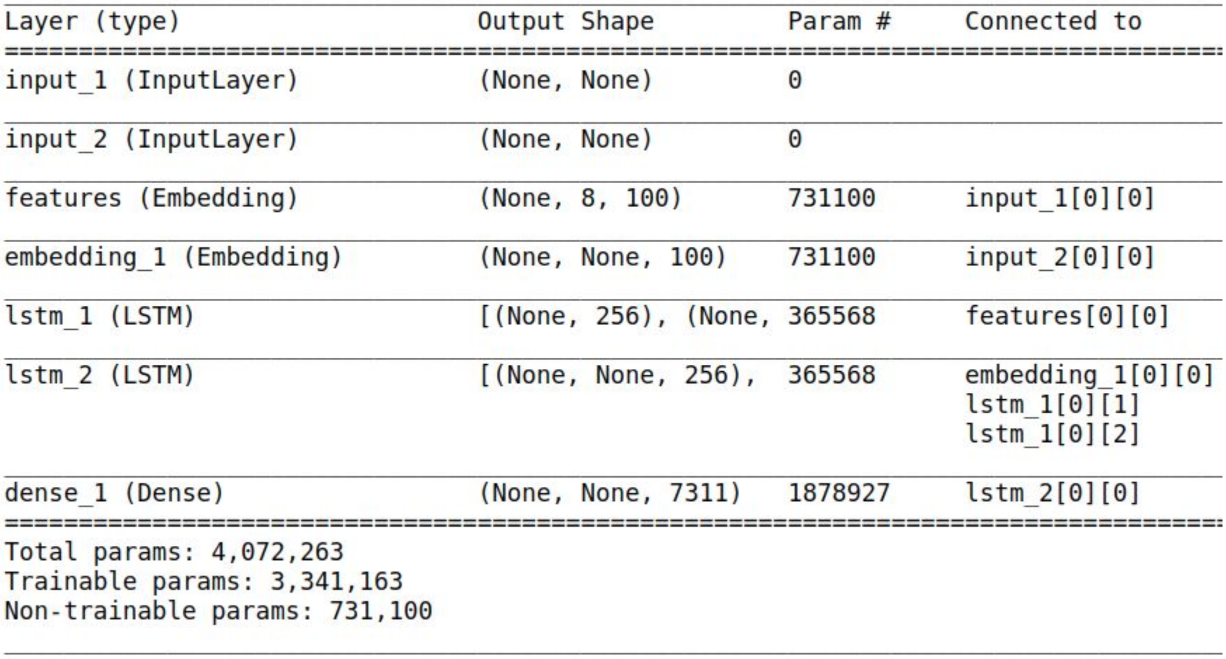




```
Layer (type)                    Output Shape          Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, None)          0
__________________________________________________________________________________________________
input_2 (InputLayer)            (None, None)          0
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 100)     731100      input_1[0][0]
__________________________________________________________________________________________________
embedding_2 (Embedding)         (None, None, 100)     731100      input_2[0][0]
__________________________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 256), (None, 365568      embedding_1[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, None, 256),   365568      embedding_2[0][0]
                                                                  lstm_1[0][1]
                                                                  lstm_1[0][2]
__________________________________________________________________________________________________
dense_1 (Dense)                 (None, None, 7311)    1878927     lstm_2[0][0]
==================================================================================================
Total params: 4,072,263
Trainable params: 4,072,263
Non-trainable params: 0
__________________________________________________________________________________________________
```

Reddit - Seq2seq_word_pre


input_1: InputLayer | input: | (None, None)
output: | (None, None)
features: Embedding | input: | (None, None)
output: | (None, 8, 100)
input_2: InputLayer | input: | (None, None)
output: | (None, None)
lstm_1: LSTM | input: | (None, 8, 100)
output: | [(None, 256), (None, 256), (None, 256)]
embedding_1: Embedding | input: | (None, None)
output: | (None, None, 100)
lstm_2: LSTM | input: | [(None, None, 100), (None, 256), (None, 256)]
output: | [(None, None, 256), (None, 256), (None, 256)]
dense_1: Dense | input: | (None, None, 256)
output: | (None, None, 7311)

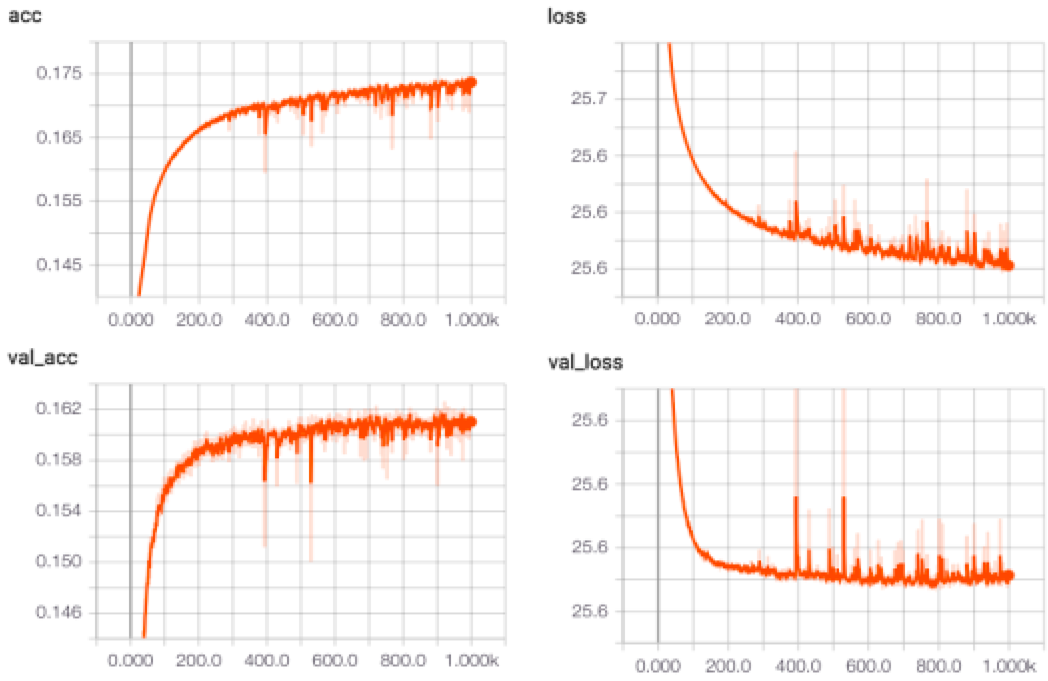


```
________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
================================================================================
input_1 (InputLayer)            (None, None)         0
________________________________________________________________________________
input_2 (InputLayer)            (None, None)         0
________________________________________________________________________________
features (Embedding)            (None, 8, 100)       731100      input_1[0][0]
________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 100)    731100      input_2[0][0]
________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 256), (None, 365568      features[0][0]
________________________________________________________________________________
lstm_2 (LSTM)                   [(None, None, 256),  365568      embedding_1[0][0]
                                                                 lstm_1[0][1]
                                                                 lstm_1[0][2]
________________________________________________________________________________
dense_1 (Dense)                 (None, None, 7311)   1878927     lstm_2[0][0]
================================================================================
Total params: 4,072,263
Trainable params: 3,341,163
Non-trainable params: 731,100
________________________________________________________________________________
```

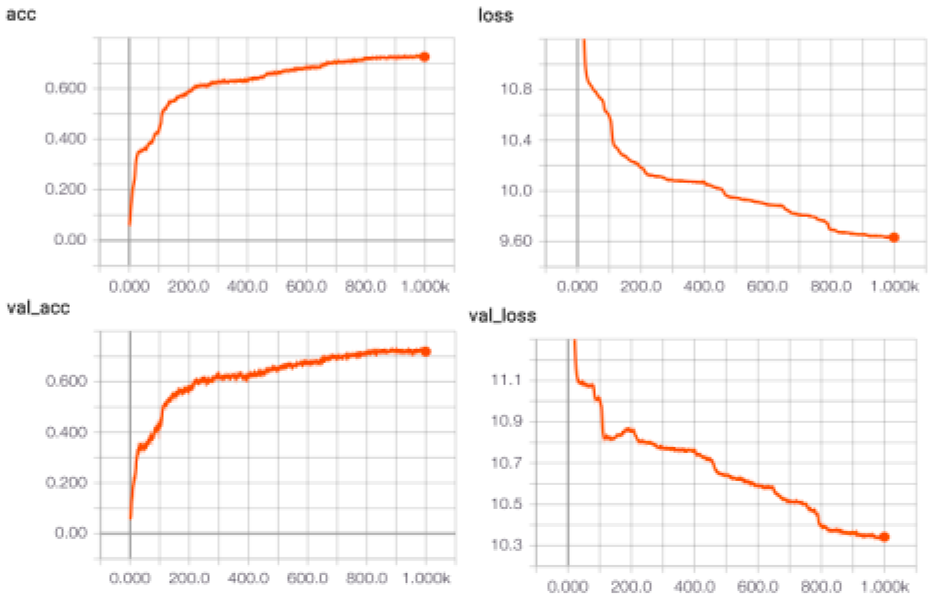
Training processes

With all the determined parameters and model architectures, we trained each model for 1000 epochs and observed the training loss, accuracy and validation loss and accuracy. We showed the example of Movie Seq2seq_char model training processes below. Then the models are used for prediction.

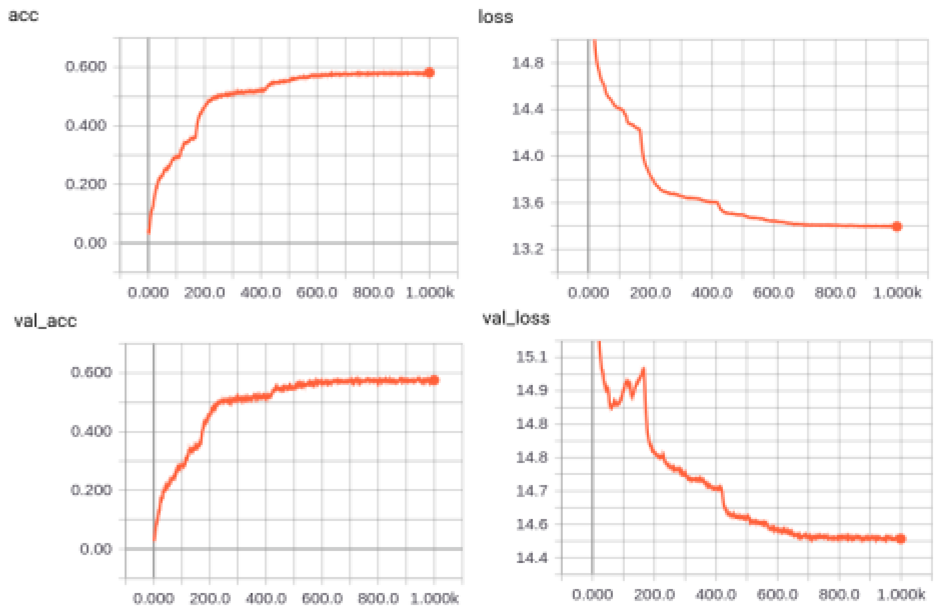
Movie -- Model_seq2seq_models_char1000



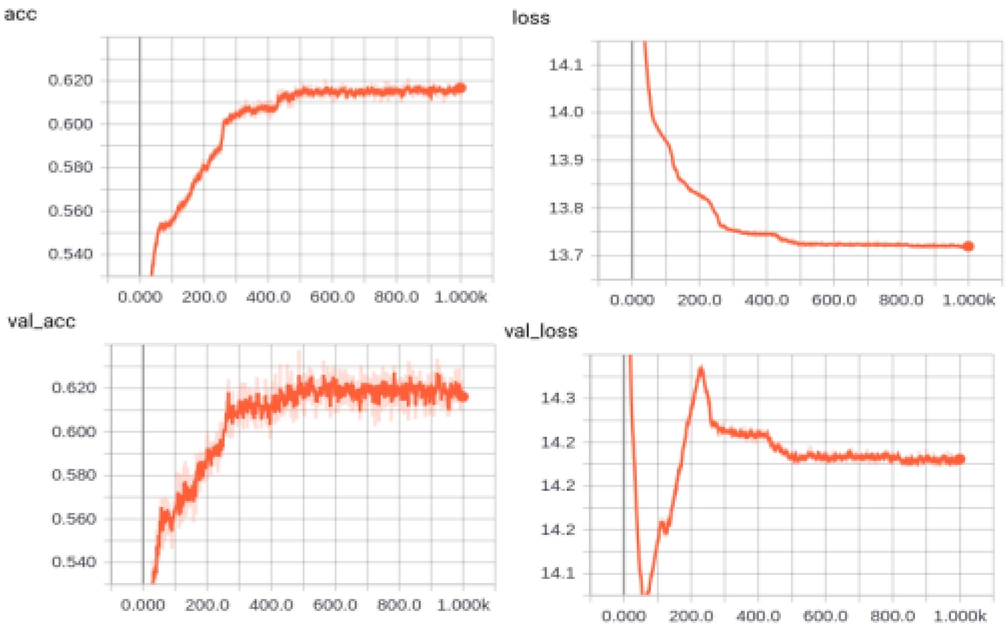
Movie -- Model_seq2seq_models_auto_word1000



Reddit -- Model_seq2seq_models_auto_word1000



Reddit -- Model_seq2seq_models_pre_word1000



Generation example

After we had the prediction models, we tested on a few examples to generate the responses. We also calculated the BLEU and ROUGE scores.

Table: generation examples from char-based models

Model	Input	Reference	Generation	Cosine Similarity	BLEU	ROUGE
Movie -char	have fun tonight	tons,	yes	0	0	0
Movie -char	shut up	does it hurt ,	what	0	0	0
Movie -char	get outta here	what the hell happened ,	what	0	0	0
Movie -char	yeah	what do you think ,	yes	0	0	0
Reddit -char	11 1100 0000	11 1100 0001,	2 000 0	0	0.33	0
Reddit -char	okay this is epic	thanks,	thank	0	0	0
Reddit -char	happy bday	thx,	thank	0	0	0
Reddit -char	hahaha	w h a t,	thank	0	0	0

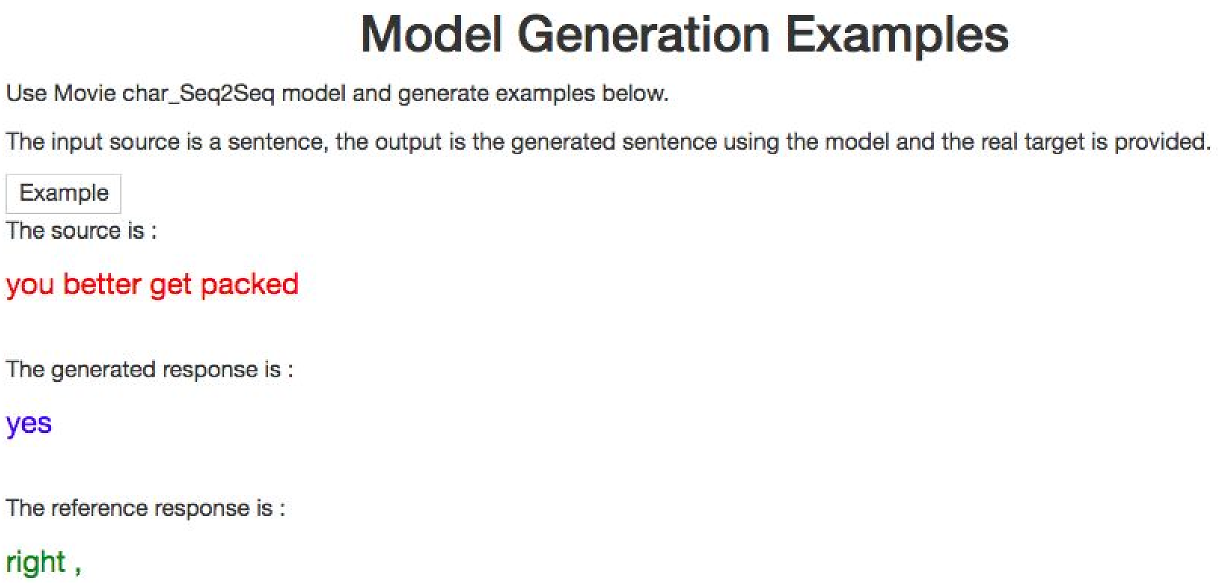
Table: generation examples from word-based models

Model	Input	Reference	Generation	Cosine Similarity	BLEU	ROUGE
Movie-word-auto	have fun tonight	tons	oh oh oh	0.81	0.33	0
Movie-word-auto	wow	let s go	yeah yeah	0.02	0	0
Movie-word-auto	emil	surprise surprise	shit shit	0.33	0	0

Movie-word-auto	do what	this	how how	0.33	0	0
Reddit-word-pre	splinter	w h a t	maybe shit	0.064	0	0
Reddit-word-pre	oh ok thanks	no problem	no no no	0.59	0	0
Reddit-word-pre	thank you	this is the right answer	yes yes yes	0.002	0	0
Reddit-word-pre	1 122 304	1 122 310	309 309 309	0.11	0	0

Web application

Django is Python web framework, which we used to make a website so that we can provide interactive online dialogue generation, as seen http://141.225.146.188:8080/DL/DS.



Model Generation Examples

Use Movie char_Seq2Seq model and generate examples below.

The input source is a sentence, the output is the generated sentence using the model and the real target is provided.

The source is :

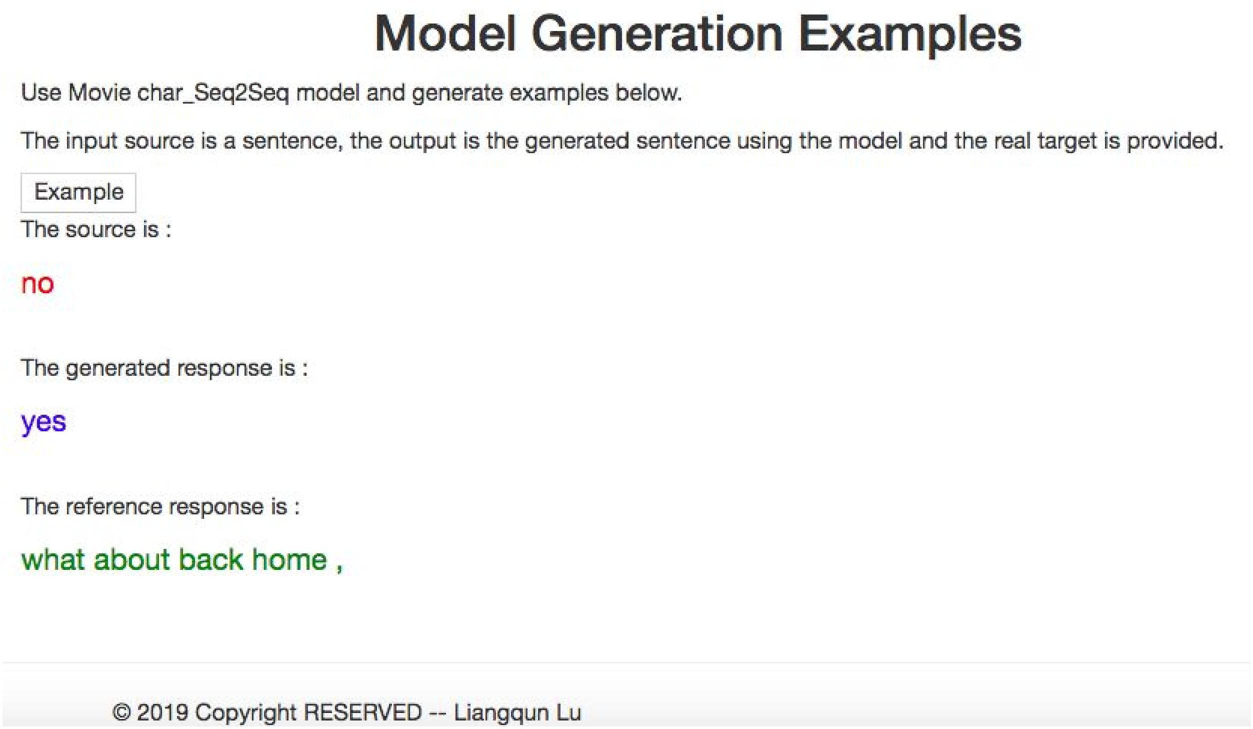
you better get packed

The generated response is :

yes

The reference response is :

right ,



Model Generation Examples

Use Movie char_Seq2Seq model and generate examples below.

The input source is a sentence, the output is the generated sentence using the model and the real target is provided.

The source is :

no

The generated response is :

yes

The reference response is :

what about back home ,

© 2019 Copyright RESERVED -- Liangqun Lu

Future Work

We made applications of seq2seq models using 2 non-task-oriented datasets. However, there are improvements which possibly generate better dialogue responses. In the future, we plan to include task-oriented datasets. In the current Seq2seq models, we consider to add attention layer or transformer. We also consider GANs to train models. We also hope to make use of style transfer to generate personalized responses.

Conclusions

In this project, we applied LSTM seq2seq models to learn models for response generation in 2 datasets. With 1000 epochs training, the char-based model reached around <0.175 accuracy and word-based model reached < 0.75 accuracy. Using the automatic evaluations BLEU and ROUGE, the generations have low score, while make sense sometimes. The generation lacks diversity in all models, but generate random numbers for the numbers input.

References

1. Chen Y-N, Celikyilmaz A, Hakkani-Tur D. Deep Learning for Dialogue Systems. Available: https://aclweb.org/anthology/C18-3006

2. Chen H, Liu X, Yin D, Tang J. A Survey on Dialogue Systems: Recent Advances and New Frontiers [Internet]. arXiv [cs.CL]. 2017. Available: http://arxiv.org/abs/1711.01731

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

4. Sutskever I, Vinyals O, Le QV. Sequence to Sequence Learning with Neural Networks. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27. Curran Associates, Inc.; 2014. pp. 3104–3112.

5. Li J, Monroe W, Ritter A, Galley M, Gao J, Jurafsky D. Deep Reinforcement Learning for Dialogue Generation [Internet]. arXiv [cs.CL]. 2016. Available: http://arxiv.org/abs/1606.01541

6. Yu L, Zhang W, Wang J, Yu Y. SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient [Internet]. arXiv [cs.LG]. 2016. Available: http://arxiv.org/abs/1609.05473

7. Li J, Monroe W, Shi T, Jean S, Ritter A, Jurafsky D. Adversarial Learning for Neural Dialogue Generation. Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics; 2017. pp. 2157–2169.

8. Nio L, Sakti S, Neubig G, Yoshino K, Nakamura S. Neural Network Approaches to Dialog Response Retrieval and Generation. IEICE Trans Inf Syst. 2016;E99.D: 2508–2517.

9. Ritter A, Cherry C, Dolan WB. Data-driven Response Generation in Social Media. Proceedings of the Conference on Empirical Methods in Natural Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics; 2011. pp. 583–593.

10. Serban IV, Sordoni A, Bengio Y, Courville A, Pineau J. Building end-to-end dialogue systems using generative hierarchical neural network models. Thirtieth AAAI Conference on Artificial Intelligence. 2016. Available: https://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/viewPaper/11957

11. Li J, Galley M, Brockett C, Gao J, Dolan B. A Diversity-Promoting Objective Function for Neural Conversation Models [Internet]. arXiv [cs.CL]. 2015. Available: http://arxiv.org/abs/1510.03055

12. Lowe R, Noseworthy M, Serban IV, Angelard-Gontier N, Bengio Y, Pineau J. Towards an Automatic Turing Test: Learning to Evaluate Dialogue Responses [Internet]. arXiv [cs.CL].

2017. Available: http://arxiv.org/abs/1708.07149

13. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative Adversarial Nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27. Curran Associates, Inc.; 2014. pp. 2672–2680.

14. Zhou X, Li L, Dong D, Liu Y, Chen Y, Zhao WX, et al. Multi-turn response selection for chatbots with deep attention matching network. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2018. pp. 1118–1127.

15. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is All you Need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems 30. Curran Associates, Inc.; 2017. pp. 5998–6008.