Improving Dialog Systems with Pre-trained Models

Rajat Gupta 19BM6JP17 • Karthikeya R 19BM6JP32
Vineet Kumar 19BM6JP46 • Paturu Harish 19BM6JP55

Presented at Final End Term Project Evaluation
for
CS60078 Complex Networks

Indian Institute of Technology Kharagpur

June 12, 2020

Slide Outline

1. Introduction & Problem Description
2. Methodology & Our Contribution
3. Evaluation & Discussion

Goal

Exploring how **Dialog Systems** can benefit from **Pretrained Models**

Dialog systems better known as interactive chat bots, are used in a wide set of applications ranging from technical support services to language learning tools

Pretrained Models Transfer Learning is where what has been learned in one setting is exploited to improve generalization in another setting. We use pre-trained HRED model to improve our dialog system through fine-tuning.

Related Works

- Sordoni et al. (2015) proposed a novel **h**ierarchical **r**ecurrent **e**ncoder-**d**ecoder architecture **(HRED)** For generating **Context-Aware Query Suggestions**

Two key desirable property of any Query Suggestion engine is

- Query co-occurrence $\implies$ Query **Relatedness**, Can be used to produce suggestions. However, query co-occurrence are prone to data sparsity & perform poorly on unseen data.
- Previous submitted queries provide useful **context**. Order in which past queries are submitted is also crucial[1].
- However, *Key challenge* is dealing with the **growth of diverse contexts**, since it induces sparsity, and classical count-based models become unreliable [2]

[1] Huang et al. CIKM 2009
[2] Cao et al. SIGKDD 2008

Motivation

Training dialog system needs extremely large amount of data, which is often not available $\rightarrow$ undesirable result $\times$

Our Contribution

We are trying to explore if we can improve our dialog system through fine-tuning on a pre-trained model.

Dataset

Daily Dialog Dataset

Li, Yanran, et al. "DailyDialog: A Manually Labelled Multi-turn Dialogue Dataset." IJCNLP 2017.

- High quality, less noisy dialogue dataset
- Multiturn – suitable to train compact conversational models
- topic and physical context focused conversations

SET	NUMBER OF SENTENCES	NUMBER OF DIALOGS
Training	87,170	11,119
Validation	8,069	1,001
Testing	7,740	1,001

Table: Dataset Subdivison

Data Preprocessing

- Raw dataset is _eou_ delimited, we converted it into .csv
- As can be seen from the Previous Table 1, the number of dialogs in the training data set is not much. Hence, we created additional samples for each dialog by keeping $<U_1....U_{t-1}>$ as context and $<U_t>$ (where U_t is the utterance) as the corresponding response, for t varying from 3 to the length of each dialog.

Slide Outline

Evaluation Metrics

Perplexity

In general, perplexity is a measurement of how well a probability model predicts a sample.
low perplexity is good and high perplexity is bad since the perplexity is the **exponentiation of the entropy**.

BLEU

The Bilingual Evaluation Understudy Score, or BLEU for short, is a metric for evaluating a generated sentence to a reference sentence.
A perfect match results in a score of 1.0, whereas a perfect mismatch results in a score of 0.0.

Workflow

1. Training a Vanilla (standard) Seq2Seq RNN Auto-encoder in PyTorch to be used as the pre-training model for HRED model
2. Evaluating the model through Perplexity Score and BLUE Score metrics
3. Modifying the original architecture by adding LSTM context encoder to make it a Hierarchical Recurrent Encoder Decoder (HRED) model
4. Comparing the HRED model trained with and without pre-trained weights using the metrics, mentioned above

Training vanilla RNN Auto-Encoder

encoder function *maps* the input space to a different latent space, followed by a decoder function that *maps* the latent space to a different target space

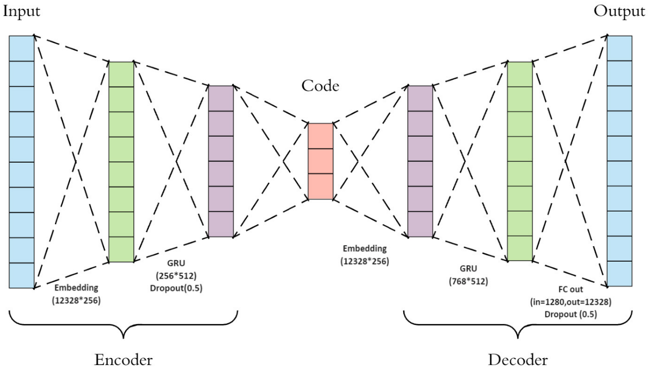
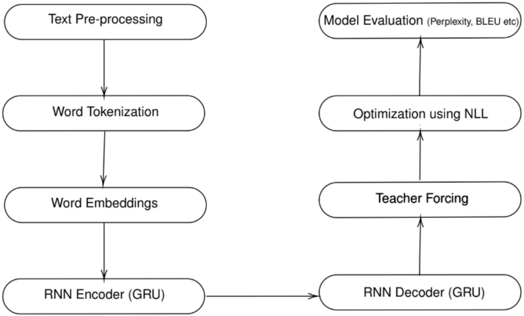


Figure: Model Architecture

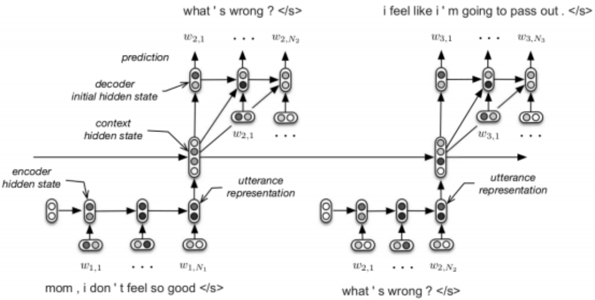
Teacher Forcing

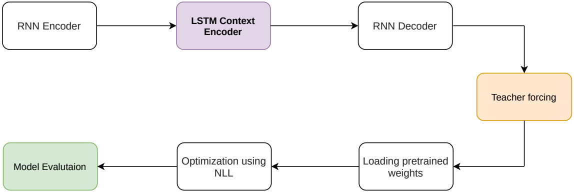
Teacher forcing is a strategy for training recurrent neural networks that uses model output from a prior time step as an input. This is implemented in the architecture using Teacher Forcing Ratio, some probability set in prior, we use the current target word as the decoder's next input rather than using the decoder's current guess. Teacher forcing ratio of 0.5 is used in our model.

Workflow


Text Pre-processing
Word Tokenization
Word Embeddings
RNN Encoder (GRU)
RNN Decoder (GRU)
Teacher Forcing
Optimization using NLL
Model Evaluation (Perplexity, BLEU etc)

HRED Model


what ' s wrong ? </s>
i feel like i ' m going to pass out . </s>
prediction
decoder
initial hidden state
context
hidden state
encoder
hidden state
utterance
representation
utterance
representation
mom , i don ' t feel so good </s>
what ' s wrong ? </s>


RNN Encoder
LSTM Context Encoder
RNN Decoder
Teacher forcing
Loading pretrained weights
Optimization using NLL
Model Evalutaion

Our HRED Architecture & Specifications

- Encoding all utterances in context using encoder to get utterance vectors
- These utterance vectors are fed to LSTM to obtain a single context vector
- The context vector is then fed to the decoder to generate the dialog response

```
Seq2Seq(
  (encoder): Encoder(
    (embedding): Embedding(12328, 256)
    (rnn): GRU(256, 512)
    (dropout): Dropout(p=0.5, inplace=False)
  )
  (con_enc): Context_Encoder(
    (rnn): LSTM(512, 512)
    (dropout): Dropout(p=0.2, inplace=False)
  )
  (decoder): Decoder(
    (embedding): Embedding(12328, 256)
    (rnn): GRU(768, 512)
    (fc_out): Linear(in_features=1280, out_features=12328, bias=True)
    (dropout): Dropout(p=0.5, inplace=False)
  )
)
```

Slide Outline

Model Evaluation – RNN

EPOCH #	Training Loss	Validation Loss	Training PPL	Validation PPL
1	1.571	2.049	4.812	7.762
2	1.326	1.978	3.767	7.231
3	1.160	1.909	3.191	6.743
4	1.031	1.813	2.804	6.128
5	0.931	1.768	2.536	5.860

Table 2: Training and Validation Results for RNN Auto-Encoder

Test Loss	Test PPL
1.839	6.292

Table 3: Test Results for RNN Auto-Encoder

Description	Score
Corpus Bleu	45.27
Sentence Bleu	59.11
Sentence Bleu with Smoothing	53.96

Table 4: BLEU Scores using RNN Auto-Encoder

Train

- **Input:** [' ', 'really', '?', 'i', 'think', 'that', '"s', 'impossible', '!']
- **Prediction:** [' ', 'really', '?', 'i', 'think', 'that', '"s', 'absolutely', '!', '<eos>']

Test

- **Input:** [' ', 'mainly', 'because', 'we', "'ve", 'invested', 'in', 'a', 'heat', 'recovery', 'system', '.']
- **Prediction:** [' ', 'because', 'because', 'we', "'ve", 'lived', 'in', 'a', 'few', 'or', 'system', '.', '<eos>']

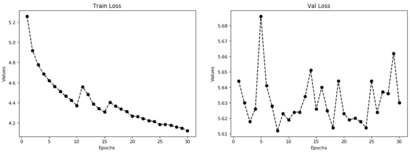
Results – HRED without pre-trained weights

EPOCH #	Training Loss	Validation Loss	Training PPL	Validation PPL
1	4.186	5.624	65.738	276.998
2	4.175	5.637	65.056	280.667
3	4.159	5.636	64.031	280.204
4	4.148	5.662	63.287	287.615
5	4.122	5.630	61.674	278.755

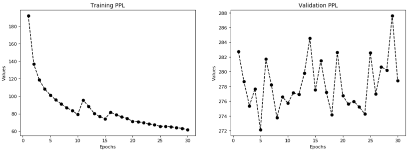
Table 5: Training and Validation Set Results for HRED model (without pre-trained weights)

Test Loss	Test PPL
5.649	284.113

Table 6: Testing Set Results for HRED model (without pre-trained weights)



Training Loss



Perplexity Scores

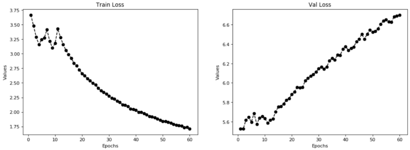
Results – Fine-tuned HRED (with pre-trained weights)

EPOCH #	Training Loss	Validation Loss	Training PPL	Validation PPL
51	1.843	6.53	6.315	685.392
52	1.825	6.558	6.205	705.171
53	1.808	6.603	6.097	737.017
54	1.793	6.639	6.009	764.6
55	1.779	6.65	5.922	773.006
56	1.766	6.63	5.85	757.498
57	1.759	6.625	5.808	774.124
58	1.732	6.683	5.654	798.809
59	1.742	6.689	5.707	803.461
60	1.711	6.701	5.536	813.497

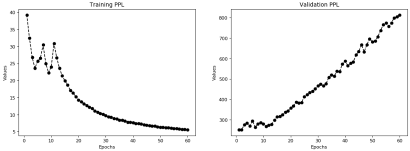
Table 7: Training and Validation Set Results for Fine-tuned HRED model with Pre-trained Weights)

Test Loss	Test PPL
6.385	593.114

Table 8: Testing Set Results for Fine-tuned HRED model with Pre-trained Weights)



Training Loss



Perplexity Scores

Predicted Outputs by non pre-trained HRED model

- **Context:** ['[', '"', 'we', "'ve", 'managed', 'to', 'reduce', 'our', 'energy', 'consumption', 'in', 'our', 'factory', 'by', 'about', '15', 'per', 'cent', 'in', 'the', 'last', 'two', 'years', '.', '"', ',', '"', 'that', "'s", 'excellent', '.', 'how', 'have', 'you', 'managed', 'that', '?', '"', ']']
- **Predicted Response:** ['i', 'dressed', 'i', "'m", 'be', 'pleased', 'to', 'gamble', 'the', 'regulations', 'within', 'september', 'bureau']

Predicted Outputs by pre-trained HRED model – I

- **Context:**
  "excuse me , sir , i 'm afraid you ca n't park your car here . " , " why not ? it 's my parking space . "
- **Ground Truth Response:**
  i 'm afraid not , sir .
- **Predicted Response:**
  perhaps , i ca n't on that i could i on my book . <eos>

Predicted Outputs by pre-trained HRED model – II

- **Context:**
  ' believe it or not , tea is the most popular beverage in the world after water . ' , ' well , people from asia to europe all enjoy tea . ' , ' right . and china is the homeland of tea . '
- **Ground Truth Response:**
  yes , chinese people love drinking tea so much . some even claim they ca n't live without tea .
- **Predicted Response:**
  the hard need more the traditional of china is more more more more more more expensive . <eos>

BLEU Score

Description	Non Pre-trained HRED	Pre-trained HRED
BLEU	0.03	0.22

Table: Comparison of BLEU Scores

Summary

Following is the comparison of performance between pre-trained and non-pre-trained HRED model.

Description	Non Pre-trained HRED	Pre-trained HRED
Epochs	30	60
Training PPL	61.674	5.536
Validation PPL	278.755	813.497
Test PPL	284.113	593.114
BLEU	0.03	0.22

Table: Non Pre-trained vs. Pre-trained Model Summary

Key Takeaway, Discussions and Future Directions

Takeaway

- Non pre-trained HRED model produces repetitive generic responses.
- However, after pre-training, this problem of natural language generation goes away, but context relevance still remains an issue.

Discussions

Accounting for temporal structure of context can overcome problem of NLG but not of NLU context. $\rightarrow$ chatbot blurting out non-generic, meaningful sentences but irrelevant to the user it is talking to.

Extensions to our work

- Use of Bi-directional RNNs to overcome the problem with long utterances
- Using pre-trained sentence embeddings like BERT with transformers which are purely Attention-based, neglecting RNNs altogether

References

1. Li, Y., Su, H., Shen, X., Li, W., Cao, Z., and Niu, S. Dailydialog: A manually labelled multi-turn dialogue dataset. arXiv preprint arXiv:1710.03957, 2017.
2. Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. Building end-to-end dialogue systems using generative hierarchical neural network models. In Thirtieth AAAI Conference on Artificial Intelligence, 2016.
3. Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Grue Simonsen, J., and Nie, J.-Y. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, pp. 553–562, 2015.

Code associated with experiment, report and this presentation is available
`https://github.com/rajatguptakgp/pretrained_dialog_system`