# Dancing Seq2Seq: An Adversarial Approach to Transformer Generative Dialog

**Alan Salimov**


## Abstract

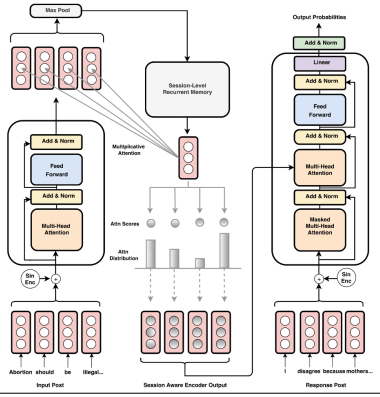
Generative dialogue systems use RNNs, attention mechanisms, and adversarial training to create coherent dialogue on both closed and open topic corpora. While previous state of the art approaches used recurrent encoder-decoder(HRED) approaches, the transformer approach has shown to provider faster and better results across NLP applications. By using stacked layers of attention instead of RNNs, transformers have shown that a language model can be learned using solely attention. However, all of these approaches suffer from a lack of diverse responses. HRED trained adversarially has shown to outperform vanilla HRED; in this paper, we propose an adversarialy trained generative dialog system, mirroring the utterance level discriminator proposed in We generate dialog using a multi-gpu adapted version of the transformer seq2seq system.We train this using the Ubuntu Help Forum Dialog Corpus, a closed-topic corpus.


## Corpus

Ubuntu Dialogue Corpus ((UDC) dataset. This dataset was extracted from the Ubuntu Relay Chat Channel. , The dataset is very large compared to the Internet Argument Corpus, and is closed topic; all dialogs are related specifically to Ubuntu Forum topics. The UDC contains about 1.85 million conversations with an average of 5 utterances per conversation, with a maximum per-utterance sequence length of 40. The maximum number of utterances is 25.

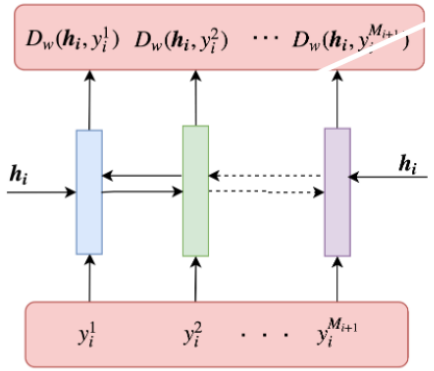
**Generator**

Slightly modified version of the transformer seq2seq model developed by Adjenji, Lee, and Liu, trained and updated step-wise.




**Discriminator**

Slightly modified version of the discriminator presented in the hred-gan implementation by Olabiyi, Khazane, and Salimov




## Model Training

- Used DataParallelModel and DataParallelCriterion loss developed by Zhang, et al
- Increase batch size 4->1800, training set size 11800->1.6 million,
- Generator only training used CrossEntropyLoss
- Adversarial used BinaryCrossEntropyLoss

## Results and Conclusion

- While we were able to realistically fit a much larger dataset and allow both vanilla and adversarial versions to be trained with state of the PyTorch Multi-GPU training implementations, we were unable to derive useful results.
- Training time for both vanilla and adversarial models went from over 24 hours to roughly 3 hours per epoch.