# Modeling Agents in a Text-Based Fantasy Game

Prithu Dasgupta and Pavlo Lyalyutskyy

Brown University: Computational Linguistics

**Abstract.** Text Adventure games, or interactive fiction, have been a marvel of computational linguistics use in the commercial setting. In the early days, they relied on text parsers of varying complexity to take in user input in the form of natural language and progress the story-line in some way. These games suffered from a few limitations, particularly a finite amount of game states and limited number of hard-coded possible actions. Modern deep learning models such as GPT-2 and BERT have achieved success in language modeling as well as conversational AI by being able to generate an infinite number of outputs and make sense of a seemingly infinite number of inputs. In this paper, we explore using these two well-regarded models to recreate agents in an example text adventure game. The vast size of both the dataset and model parameters led to limitations, but our experiments surely point to these as being reasonable ways of modeling these game agents.

**Keywords:** Chatbot · BERT · GPT2 · Game agents.

## 1 Introduction

Much experimentation has been done using state-of-the-art neural networks to be able to learn to simulate conversation between agents. Given multiple persona and input from one of these persona, contemporary models such as GPT2 and BERT have been able to construct a reasonable response of the other persona. However, these responses as is do not necessarily pertain to the environment the agents inhabit. We investigate methods to extend this notion of a chatbot to be influenced by additional factors: environment, actions, and backstory. In addition, the typical chatbot is fairly limited in its evaluation in determining whether it can hold a reasonable conversation. Thus, we will explore fitting our chatbot to the scenes of a fantasy game to demonstrate that the trained model can talk and behave in the game reasonably well.

We will explore this sector using the LIGHT (Learning in Interactive Games with Humans and Text) dataset, similar to the originally published LIGHT paper[1]. This dataset consists of a crowd sourced game world (663 locations, 3462 objects and 1755 characters) described all in text. In short, the game itself allows multiple agents each with a unique persona to speak, act, and emote in a collection of environments containing interactive objects.

## 2 Related Work

Much of these experiments comes from the original LIGHT paper[1] as we will explore modeling the LIGHT game world using both a BERT bi-ranker and BERT cross-ranker with several small modifications. The use of BERT follows from both original BERT paper for next sentence prediction[2] as well as the next utterance retrieval task from the TransferTransfo paper[3].

## 3 Experiments

We began by reimplementing and restructuring the main models in the original LIGHT paper: the BERT Bi-ranker and the BERT Cross-Ranker. Then as an extension and with our previous knowledge, we looked at using Huggingface's GPT2 to view this problem as a language modeling.

### 3.1 BERT Bi-Ranker

The original conception of the BERT Bi-ranker begins with the original BERT paper[2], specifically its description on next sentence prediction. This boils down to utilizing BERT's power to look at context to determine the likelihood that a certain answer could be paired with an initial question. Generally, the next sentence prediction tasks entails determining the relation between two sentences. Two sentences A and B will be concatenated together with some sort of separation token in between. 50% of the time, sentence B will be the true continuation of sentence A and the other 50% of the time, sentence B will be a randomly pulled sentence from the dataset[2]. The correct sentence will be labeled with a 1, while the random sentences will be labeled with a -1. This complex task has been brought down to a binary classification problem.

In context with the LIGHT platform, this approach is mainly followed. In terms of data processing, we sample a small portion of 10% of the large dataset with some additional threshold of conversation history. We represent the current sentence as the context of the game world with the previous conversation history of the agents with the next possible utterance following a separation token. The context of the game world consists of the setting's name and description, the name and backstory of each of the agents, and the interactive objects in the scene at the current time step. To limit the window size of the model but to use the context of the conversation, at most the past three utterances are appended to the context. The context is then stored as a separate vector representation than the next utterance instead of the two being concatenated together. The responding speech, action, or emote is stored in an input vector separate from the context. Each of the context and input vectors have their own attention masks. The first token of the context is a special task token which lists whether the corresponding input is an utterance, act, or emote.

Our implementation of the next utterance task or "bi-ranker" deviates slightly from both the original BERT paper and the implementation described in the original LIGHT paper. As mentioned, the context and input are distinct vectors constructed in preprocessing. We pass each of the context and input into the same pretrained Huggingface BERT model and ideally want their embeddings to be similar if the two are a continuation and different otherwise. Thus, once passed into the BERT model, we pass each of these two outputs through a shared linear layer, and return the cosine similarity of these two outputs. This model is trained using a mean squared error loss function as the model should predict values close to one for the correct responses and close to negative one for random responses. This is accounted for by the returning the cosine similarity of the two feature vectors.

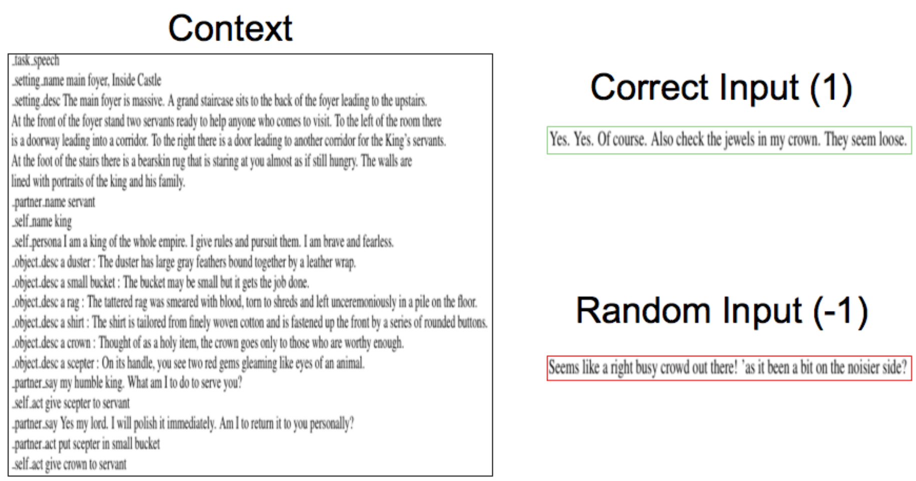


**Fig. 1.** BERT Bi-Ranker Preprocessing Example

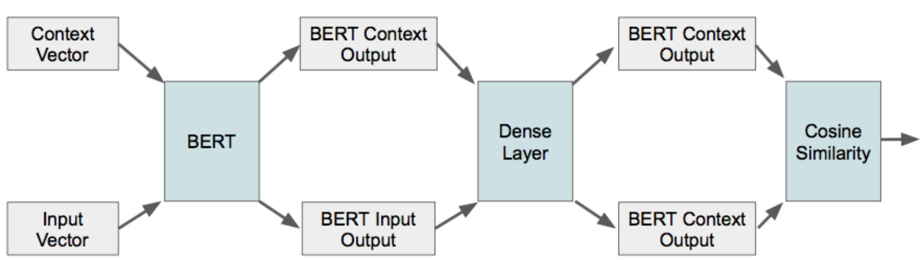


**Fig. 2.** BERT Bi-Ranker Model Architecture

### 3.2 BERT Cross-Ranker

The cross ranker differs from the bi-ranker that instead of returning a probability for a single candidate that the next utterance is likely, the cross ranker takes in a batch of candidates and returns a softmax over all of these candidates. This model follows from both the original LIGHT paper[1] as well as TransferTransfo[3].

For data processing, because of the large scope of data we have and the fact the original LIGHT paper noted that this model is far more computationally intensive than the BERT bi-ranker, we restricted our cross-ranker to be a classification task among a discrete number of emotes. There are a total of twenty two possible emotes in the game world[1], but infinitely many utterances and actions. Each batch of our data passed to the corss ranker is the same context and history appended to a separation token appended to a possible emotes. One of these emotes is the correct while the other batch size minus one are randomly selected emotes from the original twenty-two. The correct emote is labeled as one, while the incorrect random emotes are labeled as zero. Again to limit the size of this vast dataset and its window size, we set a threshold of the total episodes of the entire dataset and each episodes' past speech history.

The model we implemented mostly follows from the TransferTransfo paper[3]. Given the concatenated context and possible emote, we pass that vector into the BERT model and obtain the last hidden state. We then pass the hidden state into a set of two linear layers with a RELU in between. The first layer units is equal to the BERT embedding size, while the second linear layer has a single unit. Thus, we result in a single vector of size batch size. We then take a softmax over this vector to find the most likely emote among the batch. We use sum squared error as our loss function, as the correct emote should have a probability close to one while the random emotes should have probabilities close to zero.

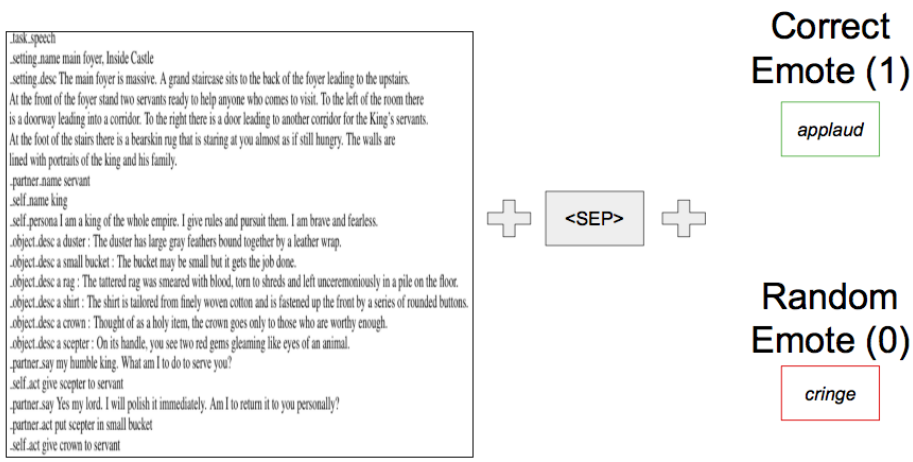


**Fig. 3.** BERT Cross-Ranker Preprocessing Example

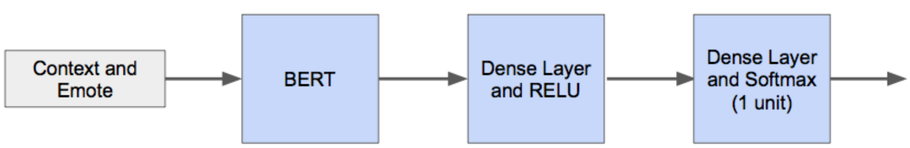


**Fig. 4.** BERT Cross-Ranker Model Architecture

### 3.3 GPT2

With our past experience of using GPT2 to represent a chatbot, we also explore using GPT2 to model our task. The difference is that when using GPT2 to represent a chatbot, typically only the utterances are input. However, in our case we also pass the context of the scene and characters which could interfere with the language model. In addition, BERT is perhaps better at exploiting context than the GPT2 language model.

Since we are now in the realm of language modeling, our labels are simply equal to our inputs as we are attempting to predict the next token. We again append all of the past context, character stories, scene objects, and part of the conversation history. Then the correct text, action, or emote follows with a task token in between. An attention mask is also returned.

For the model itself we use the GPT2LMHeadModel from Huggingface and its built in loss function. Huggingface's GPT2 implementation is fairly easy to use, so there is not much additional overhead.

## 4 Results

### 4.1 BERT Bi-Ranker

After training the BERT Bi-Ranker model on the next sentence prediction task we achieved fairly decent results. The model accuracy/recall for predicting which utterance most closely is associated with the context turned out to be 61.4%. Which is better than the baseline for random guessing (50%). This is not quite as good as the BERT Bi-Ranker from the original LIGHT paper however. In the paper, a recall of 70% was reported from a set of 20 candidates where as we were only able to achieve a recall of 61.4% from a set of 2 candidates.

Some issues that we had difficulties with was the instability of training the BERT based bi-ranker. This is in agreement with the LIGHT paper as the BERT Bi-Ranker is supposed to be a fast method for evaluating utterances compared to the Cross-Ranker. By using a dot product, or in our model, the cosine similarity, between the context embedding and the candidate utterance embedding significant information is lost. The BERT Bi-Ranker based model however proves to be more effective than the baseline at finding a fast similarity metric between our two embeddings and with more training time and experimentation the model should converge better.

### 4.2 BERT Cross-Ranker

The BERT Cross Ranker is known to be computationally heavy compared to the Bi-ranker. Thus, we trained this model solely on the discrete class of emotes. This limited sample space surely lead to an increase in reported test set accuracy. The cross ranker could correctly predict about 63.7% emotes. However, this seems to be a vast improvement from the original LIGHT paper could only reach a much lower 25.8%. Keep in mind that this likely arises from the batch size we were able to work with. With limited GPU resources, we were only able to train our model without error with a batch size of three or less. A baseline for random guessing in this case would be approximately $1/3 \approx 33.3\%$ which is already larger than the LIGHT paper's benchmark. Thus, it is likely with more resources, they were able to sample more candidates emotes for a given context, which would lead to overall accuracy decreasing since only one of these batch size many candidates is the correct emote. With limited time and resources, we were not able to approximate the cross ranker on actions and speech, but we would assume that accuracy would reduce to somewhere near the bi-ranker with the increase in sample space.

### 4.3 GPT2

We were fairly surprised by the reliability of the GPT2 model since we believed that BERT would better be able to exploit context. Using the language modeling approach with GPT2 we were able to achieve an accuracy on our seen test set of 65.1% on our seen test set across speech, actions, and emotes with a perplexity of 1.56. We believe that possibly the extra layers incorporated in the BERT bi-ranker and cross-ranker may be adding additional confusion to the model which is not accounted for in the base GPT2 model.

## 5 Conclusion

We were able to achieve our best accuracy using the GPT2 model with an accuracy of 65% per token, followed by the BERT Cross-Ranker, and then followed by the Bi-Ranker. This project was a good first step in creating language models with grounding information. Text adventure games serve as a perfect medium for studying how to create language models that not only work with a large amount of contextual information but also contain actions/emotes/dialogues are valid given the contextual history. In essence this kind of work is essential for truly intelligent conversational agents to take place.

In addition, the training time of our models were very high, even with using a small subset of the original dataset. More computational resources are likely needed to better estimate the domain. With incorporating the context of each of these scenes, our sequence lengths grew to be very long. Perhaps looking into ways to reduce this while maintaining the majority of the context could make our models much lighter.

There are many areas our model could be improved. For instance, the LIGHT paper[1], was incredibly vague in how they set up their architecture and loss functions in the model. Because of this, we had to do a lot of experimentation in figuring out what loss function was appropriate for the task and how best to create some of our models. We are sure there are better ways to setup our models so that in future works, less data and training time is needed for convergence.

## References

1. Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, Jason Weston. 2019. Learning to Speak and Act in a Fantasy Text Adventure Game. arXiv preprint arXiv:1903.03094.
2. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understand- ing. CoRR, abs/1810.04805.
3. Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. arXiv preprint arXiv:1901.08149.