CHATBOT

ABSTRACT:

A long-term objective of machine learning research is to create a conversation bot with intelligence. The majority of natural language comprehension research has been on learning from fixed training sets of labelled data with supervision at the word level (tagging, parsing tasks), or sentence level (question answering, machine translation). This kind of supervision is unrealistic given the way humans learn because they both learn and utilise language. In this research, we look into dialog-based language acquisition, where guidance is offered inadvertently and implicitly through the dialogue partner's response. We examine this configuration using two datasets: the bAbI dataset and the extensive question-answering dataset. On these tasks, we assess a number of standard learning strategies and demonstrate that a unique model incorporating predictive lookahead is a promising method for learning from a teacher's response. One unexpected outcome is that it can learn to appropriately respond to queries even without any reward-based monitoring.

OBJECTIVE:

The chatbot created is expected to learn from pre-made dialogues and their labelled outcomes. The training helps the bot to infer conclusions from the stories given as dialogues and the user can give his/her own story as a dialog to expect a result.

INTRODUCTION:

Supervised learning, which often entails using annotators to label significant amounts of data each task, has been credited with many of machine learning's accomplishments. However, by taking action and learning from the results of (i.e., the feedback from) that action, people can learn. When engaging in dialogues, or making speech utterances, humans receive feedback from the responses of other humans, which are therefore extraordinarily rich in information. This is arguably most apparent when a teacher and student are conversing, since the teacher will compliment good communication and criticise poor ones. However, whether a dialogue partner is a teacher or not, generally every response will contain an instructive training signal for picking up new language in later talks.

This research investigates whether dialogues can be used to train machine learning models. The ultimate goal is to create a dialogue agent that is intelligent and can pick up new information from discussions. In order to accomplish that, it must learn from input that is provided in natural language. The majority of machine learning tasks in the literature on natural language processing, however, do not take this form; instead, they are manually labelled at the word, segment, or sentence level (question answering), or at the named entity recognition or part of speech level (part of speech tagging) by labelers. Algorithms for learning have since been created to take advantage of that form of supervision. As a result, we must create evaluation datasets for the dialog-based language learning environment as well as models and learning algorithms.

CODE:

Setting up vocabulary of all words

```
vocab = set()
all_data = test_data + train_data
for story, question , answer in all_data:
        vocab = vocab.union(set(story))
        vocab = vocab.union(set(question))
vocab.add('no')
vocab.add('yes')

vocab_len = len(vocab) + 1
max_story_len = max([len(data[0]) for data in all_data])
max_question_len = max([len(data[1]) for data in all_data])
```

Vectorizing the data

```
vocab_size = len(vocab) + 1
tokenizer = Tokenizer(filters=[])
tokenizer.fit_on_texts(vocab)
train_story_text = []
train_question_text = []
train_answers = []

for story,question,answer in train_data:
        train_story_text.append(story)
        train_question_text.append(question)
train_story_seq = tokenizer.texts_to_sequences(train_story_text)
```

Functionalize vectorization

```
def vectorize_stories(data, word_index=tokenizer.word_index,
max_story_len=max_story_len,max_question_len=max_question_len):
```

```
for story, query, answer in data:

    # Grab the word index for every word in story
    x = [word_index[word.lower()] for word in story]
    # Grab the word index for every word in query
    xq = [word_index[word.lower()] for word in query]

    # Grab the Answers (either Yes/No so we don't need to use list comprehension here)
    # Index 0 is reserved so we're going to use + 1
    y = np.zeros(len(word_index) + 1)

    # Now that y is all zeros and we know its just Yes/No , we can use numpy logic to create this assignment
    #
    y[word_index[answer]] = 1

    # Append each set of story,query, and answer to their respective holding lists
    X.append(x)
    Xq.append(xq)
    Y.append(y)

# Finally, pad the sequences based on their max length so the RNN can be trained on uniformly long sequences.

# RETURN TUPLE FOR UNPACKING
return (pad_sequences(X, maxlen=max_story_len),pad_sequences(Xq, maxlen=max_question_len), np.array(Y))
```

CONCLUSION:

A collection of evaluation datasets and models for dialog-based language learning have been presented. The ultimate objective of this line of research is to develop a learner that can communicate with people, allowing people to effectively instruct it via dialogue. We think that the dialog-based language learning strategy we described is a modest step in that direction. Positive feedback and various forms of corrections are the only sorts of feedback that are studied in this research. Any response in a dialogue, however, has the potential to be viewed as feedback and should be helpful for learning. If forward prediction and the other strategies we attempted also work there, it should be further investigated.