# Contextualized Commonsense Inference in Dialogue

Sai Sidhardha Grandhi
*Computer Science & Engineering*
*Indian Institute of Technology*
Hyderabad, India
CS19BTECH11050

Sri Akhil Mylavarapu
*Engineering Science - AI*
*Indian Institute of Technology*
Hyderabad, India
ES19BTECH11014

***Abstract*—This report delves into the inferences of human conversations from the dataset, *CICERO*, (ContextualIzed CommonsEnse InfeRence in dialOgues). The report looks into the basic understanding of CICERO and then an analysis of a transformers model over the dataset.**

## I. Introduction

Contextualized Commonsense inference in dialogues (CCInference), which draws on both contextual data and common sense knowledge, is to predict an appropriate reaction or action in the context of the conversation. It involves understanding the dialogue's implicit objectives and meaning and making educated predictions about the dialogue's next course of action utilising background knowledge.

Recent developments in NLP, especially the creation of pre-trained language models like BERT, GPT-3, and RoBERTa, have made important strides in the field of CCInference. These models are capable of making precise predictions in conversational scenarios by efficiently learning from massive volumes of data and utilising context and common sense information

## II. Insights into the Dataset

All the conversations are dyadic conversations with five reasoning-based inferences: Cause, Subsequent event, Prerequisite, Motivation, and Emotional reaction.

```
question_set = [
    "What is or could be the cause of target?",
    "What is or could be the prerequisite of target?",
    "What is the possible emotional reaction of the listener in response to target?",
    "What is or could be the motivation of target?",
    "What subsequent event happens or could happen following the target?"
]
```

Fig. 1. Questions in the dataset

There are 5 specific questions as shown in the Fig.1 which are the questions to which we strive to get the inference.

Thanks to the Text Processing and Retrieval Course for the teaching and the guidance to this Project

TABLE I
The key-value format in the JSON files

| Key | Value |
|---|---|
| ID | Dialogue ID with dataset indicator. |
| Dialogue | Utterances of the dialogue in a list. |
| Target | Target utterance. |
| Question | One of the five questions (inference types). |
| Choices | Five possible answer choices in a list. One of the answers is human written. The other four answers are machine generated and selected through the Adversarial Filtering (AF) algorithm. |
| Human Written Answer | Index of the human written answer in a single element list. Index starts from 0. |
| Correct Answers | List of all correct answers indicated as plausible or speculatively correct by the human annotators. Includes the index of the human written answer. |

Each input data is of the form shown in Figure 1. The Key IDs are of majorly three types: Dream, Mutual, DailyDialogue.

In this project, we focused on the CICERO-MCQ task, whose objective is to choose the correct answer *at*, along with any potential correct answer from the alternate choices *Ft*, given a dialogue D, target *ut*, one of the five inference type questions *q*, and the true answer *at*. Please refer to Fig. 2 for further clarification.

## III. Pre-Processing of the data

Firstly, the data is read from its structure and formatted into a dataframe with rows: ['ID', 'context', 'choice0', 'choice1', 'choice2', 'choice3', 'choice4', 'label']. Here the context is the string which is the combination of the question, target and dialogue utterances. The choices are split accordingly and 'label' is the answer column.

Then the dataset.map() function is used to pass the data through the tokenizer. The tokenizer we used is **"bert-base-uncased"**, which is a famous pre-trained transformer tokenizer.

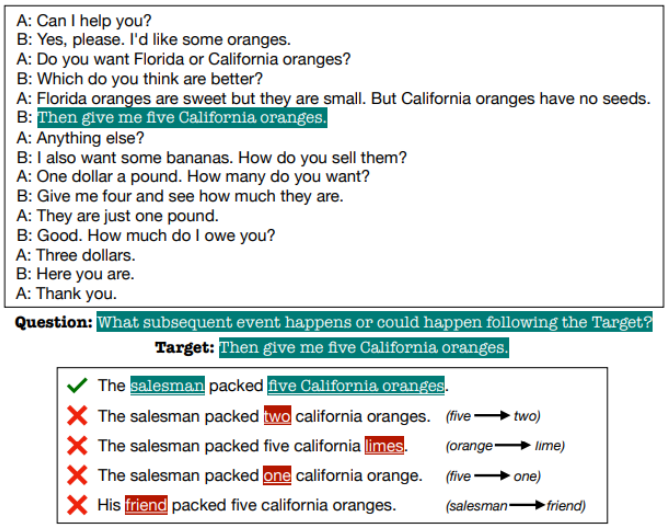


Fig. 2. An example to show the Target utterance and dependency on the answer

The dataframe after passing through the tokenizer, is with the columns: ['ID', 'context', 'choice0', 'choice1', 'choice2', 'choice3', 'choice4', 'label', 'input_ids', 'token_type_ids', 'attention_mask']

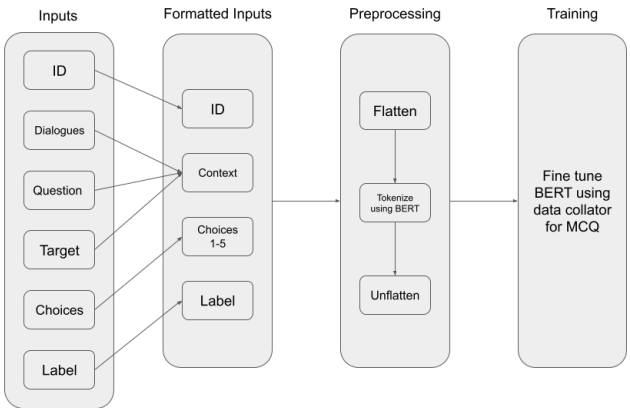


Fig. 3. The flowchart representing the implemented model.

## IV. The Transformers Model Implemented

The class, DataCollatorForMultipleChoice helps us in flattening all the model inputs (i.e, the question and all answer choices are concatenated or combined into a single sequence or tensor), apply padding (to ensure that all inputs have the same shape and can be processed by the model), and then unflatten the results (the predictions are rearranged or separated back into their original structure, where each answer choice is associated with the corresponding question or prompt).

Then we used **"bert-base-uncased"** with the help of **AutoModelForMultipleChoice** class. The input data of 456 rows was then taken to run the model with a train test validation split of 18:1:1. Some of the key parameters being: learning_rate= $3e-6$, per_device_train_batch_size=1, per_device_eval_batch_size=1, num_train_epochs=1, weight_decay=0.005.

## V. Results

- Time taken for one epoch is 1:18:13. This shows the computing power consumed by the bert-base-uncased model, which is still not the best among the existing state-of-art models.
- Validation loss for the first epoch is 1.565350. This tells that more epochs are needed to be run before the result is assessed.
- Accuracy after the first epoch is 26%.

## VI. Acknowledgements & Conclusions

Even with a dataset as small as 500 elements and a single epoch. The RAM usage is close to 11GB and is crashing on changing the epochs to a higher number or increasing the dataset size.

Efforts to make completely new models were also made as the existing ones are hard to be executed. But the padding and un-flattening issues were never being resolved.

## Link to code

Here is the code to the python notebook saved on github.

## References

[1] [CICERO: A Dataset for Contextualized Commonsense Inference in Dialogues](https://aclanthology.org/2022.acl-long.344) (Ghosal et al., ACL 2022)
[2] https://github.com/declare-lab/CICERO/tree/main/v1 [Github link for the resource paper]