UMBC

What do Large Language Models Learn about Scripts? (Sancheti, Rudinger 2022)

Aakash Subedi

Background Information

Scripts

- Structured event sequences
- Encode information regarding common scenarios
- Follows a logical ordering, using causal chains
- Allows for systems and humans to gain common sense insight into situations where details are left out

Event Sequence Descriptions (ESD)

- List of ‘Event Descriptions’ describing individual actions
- These event descriptions are in order according to time
- The grouping of the series of event descriptions called ESD

Figure 1: Three event sequence descriptions

(Regneri et al., 2010)

Problem & Motivation

Motivation

- Pretrained language models (PLM) / LLMs have seen great success in various tasks
- Authors were interested in how well retained script knowledge is in the output of PLMs
- The authors wanted to design a new task to test for script knowledge
- Find a method for performance improvement for PLMs in ESD generation

Script Knowledge Initial Testing Setup

- Zero shot probing
- Ask each LM in various different ways for an ESD for a task
- Look at the quality of the generated ESD as evidence for retaining and utilising script knowledge

Initial Results

Script Knowledge Initial Testing Results

- BART and T5 fail to output a proper ESD
- GPT-2 provided valid output, but the results were varied and questionable
- The authors conclusion: by default, PLM's are bad at utilising script knowledge, and additional work is required to obtain quality output…

7	**these are the things that happen when you bake a cake: 1. get a cake mix 2. gather together other ingredients** that you'll need for your cake (coconut oil, powdered sugar, eggs, etc.) 3. preheat oven to 350 4. pour on flour and mix by hand 5. pour in egg and mix by hand 6. add in powdered sugar and mix by hand 7. add in coconut oil and mix by hand 8. let it all sit for a minute 9. add in your flour and mix by hand 10. add in your sugar and mix by hand 11. add in your flour and mix by hand 12. add in your flour and mix by hand 13. add in your flour and mix by hand 14. add in your flour and mix by hand 15. add in your egg and mix by hand 16. add in your salt
8	**describe baking a cake in small sequences of short sentences: 1. get a cake mix 2. gather together other ingredients** 3. start to bake 4. bake a cake 5. take it to the oven 6. the cake is done 7. is it done? 8. is it still hot? 9. what is that crust? 10. what does the cake look like? If we think of a sequence of a few hundred or so words, we can take the basic idea of this example and say that we could write an equivalent, yet shorter, sentence: 1. Get a cake mix 2. Get together the ingredients 3. Bake a cake 4. Take it to the oven 5. The cake is done 6. Is it done? 7. is it hot? 8. what is that crust? 9. what does the cake look like

Table 1: Scripts generated from GPT2-L for BAKING A CAKE scenario with **bold-faced** prompts.

(Sancheti, Rudinger 2022)

Script Induction Framework (SIF)

- Made to address the many faults in zero shot PLM generation of ESDs
- Work done in two stages:
- Stage 1: Fine tune LMs on formulations for generation of ESDs

SEQUENCE here is a sequence of events that happen while baking a cake: 1. e_1 2. e_2
EXPECT these are the things that happen when you bake a cake: 1. e_1 2. e_2
ORDERED here is an ordered sequence of events that occur when you bake a cake: 1. e_1 2. e_2
DESCRIBE describe baking a cake in small sequences of short sentences: 1. e_1 2. e_2
DIRECT baking a cake: 1. e_1 2. e_2
TOKENS ⟨SCR⟩ baking a cake ⟨ESCR⟩: 1. e_1 2. e_2
ALLTOKENS ⟨SCR⟩ baking a cake ⟨ESCR⟩: ⟨BEVENT⟩ e_1 ⟨EEVENT⟩ ⟨BEVENT⟩ e_2 ⟨EEVENT⟩

Table 2: Different prompt formulations for BAKING A CAKE scenario with two events (e_1 and e_2).

(Sancheti, Rudinger 2022)

Script Induction Framework (SIF) (cont.)

- Made to address the many faults in zero shot PLM generation of ESDs
- Work done in two stages:
- Stage 2: Post-processing Generated ESDs (In 3 steps)
  - Step 1: Irrelevant Events Removal
    - Train a RoBERTa-L based relevance binary classifier, throw away irrelevant steps
  - Step 2: Event De-duplication
    - Authors tried training RoBERTa-L based paraphrase identification system, unfortunately it was removing false positives
    - Instead opted to use conservative method of only remove exact duplicates
  - Step 3: Temporal Order Correction
    - Train a RoBERTa-L based binary classifier for event pairs, to determine if one event followed the other
    - Create a directed graph for the events based off classifier output, and keep original in case sections are cyclic

Results

Results

- T5 was unable to learn off of the ESD formulations during fine tuning, results left out
- Utilising SIF allowed for higher quality ESD generation over fine tuned LMs
  - The post processing stage cleans it up significantly
- GPT-2 once again found to be better than BART for accessing script knowledge
  - Authors speculate that since GPT-2 is a generative LM and BART/T5 are encoder-decoder, it is hard to encode script knowledge in the passed scenario name

Strengths

- Authors were able to contribute a new evaluation method for the accessibility of script knowledge from LLMs
- Able to identify weaknesses in sensitivity to style of prompting and addressed that by training across multiple prompt styles
- Able to develop a framework (SIF) that improves quality of generated ESDs

Weaknesses

- SIF is unable to properly de-duplicate events in the way the authors initially envisioned
  - Only de-duplicates exact copies instead of semantically similar copies due to issues with FP detections
- Ordering is imposed by the final part of stage 2 of SIF, but some events dont necessarily have to occur in a specific order. This is unaccounted for.

References

(1) Sancheti, A., & Rudinger, R. (2022). What do Large Language Models Learn about Scripts?. In Proceedings of the 11th Joint Conference on Lexical and Computational Semantics (pp. 1–11). Association for Computational Linguistics.

(2) Regneri, M., Koller, A., & Pinkal, M. (2010). Learning Script Knowledge with Web Experiments. In Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (pp. 979–988). Association for Computational Linguistics.

Q & A