Master thesis report 6

University of Tartu

March 21, 2023

1 Instructive in-context-learning

Instruction: 'Using context: "⟨context⟩". Answer following question: "⟨question⟩" ⟨/s⟩'.

First I applied T5-XXL [Raffel et al., 2020] model. However, the model failed to predict at least one correct answer i.e. em=0. In [Lester et al., 2021] they were emphasizing that T5 models was trained of objective of span corruption i.e. the model nor received nor produces natural language. Unlearning span corruption objective can be done by fine-tuning, however, we don't have such a luxury. Anyway, [Lester et al., 2021] provides T5 variations that were adapted to LM objective by continue training original T5 with span corruption. I tried T5-XXL-LM-adapt, but still have got em=0.

2 Prefix-tuning of different length

In Figure 1 we can see following trend longer length worser performance. The best length was one, but it still significantly underperform in contrast with fine-tuning. However, later I realized that maybe bad performance comes from the fact that I'm not following experiment setup of [Li and Liang, 2021]. In their case optimal length of the prefix was 10.

3 Prompt-tuning implementation

I found two approaches how it can be implemented. (1) make changes on level of transformer library. What needs to be done on encoder level is to obtain prompt, then concat it with an input tokens, also we need to extend attention mask. On decoder level we need to extend attention mask because tries to use old attention mask. (2) substitute embedding layer in T5. Embedding

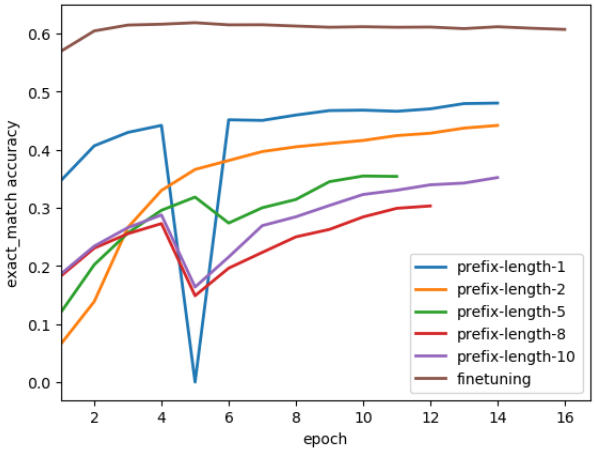

Figure 1: Various prefix-turning length

layer receives only input tokens, but we also need to change attention mask. To account for that in data loader preprocessing function(i.e. collate) we need to insert fake prompt filled with ones, so to say book the space for real prompt that would be inserted layer. Also we need to insert prompt only if fake prompt present in the input of embedding layer. Basically we want to avoid adding prompt for target tokens.

None of the approaches worked with T5 models by [Raffel et al., 2020]. The loss wasn't changing at all. Since [Lester et al., 2021] were pointing out that for prompt-tuning T5's span corruption objective make it hard to unlearn. Thus I used T5-Large-LM-adapt, and the second approached started to work(Figure 2), I'm pretty sure that the first approach wasn't working with that model, but I don't remember.

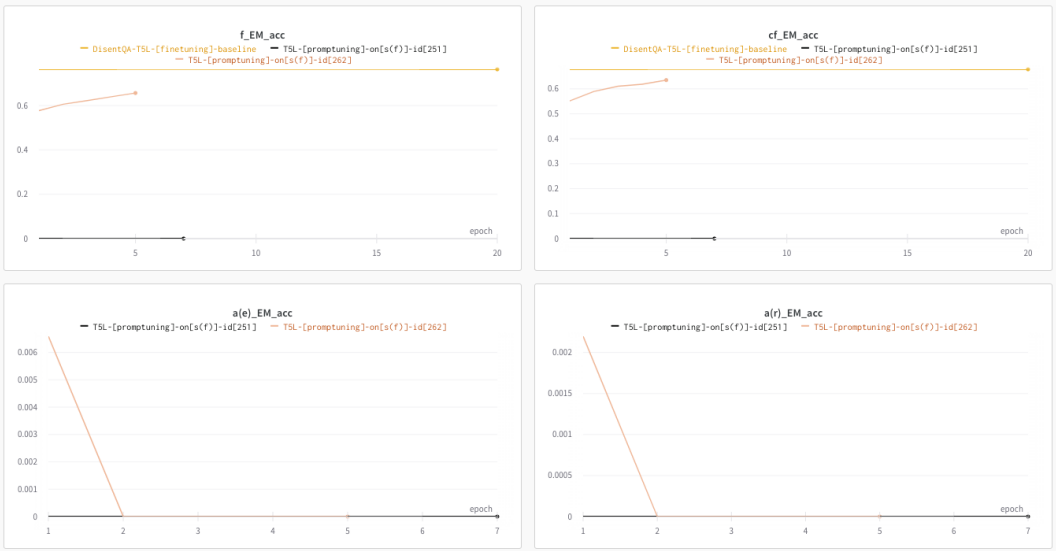

Figure 2: Prompt-tuning: second approach with original T5 and T5-Large-LM-adapt

4 Adversarial-training implementation

This was easy I just changed loss by adding additional regularization term, which represent loss on counterfactual knowledge. The strength of regularization was controlled by λ which I set to 0.25(took from paper). In terms of dataset I was using '(s) f+cf', then one loss term was computing a loss against only factual knowledge, and the second term against counterfactual knowledge. The results were promising Figure 3. The accuracy for factuals was on par with fine-tuned T5 on '(s) f', however accuracy for counterfactuals was on par with fine-tuned models on '(s) f+cf+a'.

5 Hyperparameters

Initially I started with shared hyperparameters across various approaches. The results in Figure 4 shows that for factual knowledge only prefix-tuning with length 1 underperformed, whereas all the other were on par with fine-tuning results from [Neeman et al., 2022](not from start by eventually). The same trend holds for counterfactual, except that adversarial training outperformed the baseline, but it had unfair advantage of using counterfactuals(not from test set, but from training set. To get that I've used '(s) f+cf' dataset) during

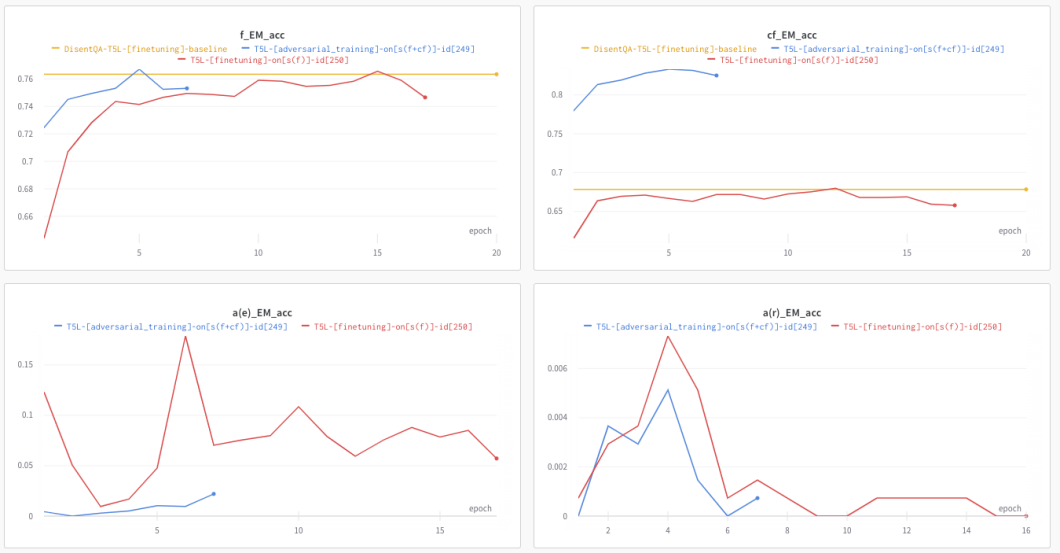

Figure 3: Adversarial training

the training.

Then I realized that maybe performance would be better if I would use hyperparameters suggested in corresponding papers. It is early to say whether it was helpful.

6 Experiments design

I have four datasets [*f, f+cf, f+a, f+cf+a*], and I have following methodologies [*prompt-tuning, prefix-tuning, adapters, lora, adversarial training*]. Thus we are going to have in total five experiments, where each experiment would consist of four sub-experiments. We are going to pick a method, and training with be done against all four datasets. This can be expressed as two nested loops. The first loop over methods and the second one over datasets.

Couple of additional points:

1. I excluded fine-tuning because [Neeman et al., 2022] already provided baseline to compare against.

2. I think that we need to predicts multiple answers as well or only. Because [Neeman et al., 2022] doesn't provided all the results for T5-Large. And single answer prediction participating only in two experiments. However, maybe we could train those baselines ourselves, but this would weaken our connection to [Neeman et al., 2022].

Figure 4: Adversarial training

7 Results summary

method(dataset)	Factual	Counterfactual
adversarial training(f+cf)	75.09	**83.15**
fine-tuning(f)	**76.78**	66.67
lora(f)	73.7	68.13
adapter(f)	74.21	67.54
DisentQA fine-tuning(f)	76.34	67.84

Table 1: Exact-match accuracy for the T5-Large models (in percent)

8 Agenda

1. **Instructive in-context-learning** wasn't successfull at all.
2. **Prefix-tuning of different length.** Smaller implies better accuracy.
3. **Prompt-tuning implementation.** Implemented two version, one didn't work with two models, and another one worked but only with model with unlearned span corruption.

Note: Considering that for prompt tuning to work we need to use different model. But this model won't be consistent with [Neeman et al., 2022]. Do we even need to be consistent?

4. **Adversarial-training implementation.** Long to train, however results are really good.

5. **Hyperparameters**. Shared ones gives performance on par with [Neeman et al., 2022]. However, the hope is that maybe if I use suggested hyperparameters from corresponding papers. It would give better results.

6. **Experiment design.** On our last meeting we decided to use only models that predict single answer. However, it will reduce the number of experiments from 5 to 2. And on top of that, they don't provided all the results for T5-Large model even in the remaining two experiments. So my proposal is to obtain those baselines ourselves by fine-tuning. Because our main point not to beat them, but rather show that performance parameter-efficient tunings should increase.

   Question: Should we use multiple answer model instead of single answer model?

9 This week plan:

1. Finish evaluation script.

2. Start real training.

References

[Lester et al., 2021] Lester, B., Al-Rfou, R., and Constant, N. (2021). The power of scale for parameter-efficient prompt tuning.

[Li and Liang, 2021] Li, X. L. and Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation.

[Neeman et al., 2022] Neeman, E., Aharoni, R., Honovich, O., Choshen, L., Szpektor, I., and Abend, O. (2022). Disentqa: Disentangling parametric and contextual knowledge with counterfactual question answering.

[Raffel et al., 2020] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer.