Master thesis report 8

University of Tartu

April 3, 2023

1 Preliminary Results

In this table my fine-tuning attained the highest accuracy on factual test-set. Which is a little bit strange since in all the other tables the accuracy should be on par with local disengQA fine-tuning. For counterfactual test-set we can observe stand adapters, lora outperformed my and disentQA fine-tuning. For adversarial-training row results are absent because data don't have counterfactual examples which are important. Prompt-tuning underforming as usual.

Methods(on 'f')	**f**	**cf**	**empty**	**random**
adversarial-training	-	-	-	-
prompt-tuning	0.6132	0.5538	0.0	0.0
adapters	0.737	0.6491	0.0	0.0
lora	0.7297	**0.6527**	0.0	0.0
fine-tuning	**0.7385**	0.6432	0.0	0.0
Local: DisentQA fine-tuning	0.7282	0.6308	0.0	0.0
Paper: DisentQA fine-tuning	0.7634	0.6784	0.0	0.0

In this table we can see all my methods except prompt-tuning outperformed my and disentQA fine-tuning.

Methods(on 'f+cf')	**f**	**cf**	**empty**	**random**
adversarial-training	0.7121	0.7729	0.0	0.0
prompt-tuning	0.6125	0.5553	0.0	0.0
adapters	**0.7326**	0.6425	0.0	0.0
lora	0.7253	**0.7897**	0.0	0.0
fine-tuning	0.7128	0.7678	0.0	0.0
Local: DisentQA fine-tuning	0.7194	0.7495	0.0	0.0
Paper: DisentQA fine-tuning	0.7575	0.7604	0.0	0.0

Which 'f+a' I'm unable to train adversarial training, and in case of prompt-tuning it still needs to be done. Other than that for factual test-set adapters achieved the highest accuracy. Lora slightly underperformed in contract with fine-tunings. However, for counterfactual test-set adapters, and lora outperformed fine-tunings.

Methods(on 'f+a')	**f**	**cf**	**empty**	**random**
adversarial-training	-	-	-	-
prompt-tuning	-	-	-	-
adapters	**0.7355**	**0.6484**	1.0	0.9861
lora	0.7187	0.6308	1.0	0.981
fine-tuning	0.7223	0.6249	1.0	0.9846
Local: DisentQA fine-tuning	0.7253	0.6198	1.0	0.9817

For counterfactual test-set all method outputperformed fine-tuning. Whereas in factual test-set lora and adversarial training slightly underperformed. And prompt-tuning still to be trained.

Methods(on 'f+cf+a')	**f**	**cf**	**empty**	**random**
adversarial-training	0.6938	0.7824	1.0	0.9788
prompt-tuning	-	-	-	-
adapters	**0.7201**	**0.7993**	1.0	0.9861
lora	0.704	0.7788	1.0	0.9832
fine-tuning	0.7077	0.7773	1.0	0.9751
Local: DisentQA fine-tuning	0.7011	0.7516	1.0	0.981

2 Problems

1. Couple days ago I discovered that in [Neeman et al., 2022] codebase they computing validation loss, however, for all my experiments I've computed validation accuracy, and based on highest value choose checkpoint. Do you think it would be a problem?

2. Some of my results trained more than 50K steps. And I don't have other checkpoints beside the best one.

3 Agenda

1. Bootstrap Test to obtain variation. If it is good idea should it be done on their codebase on ours?

References

[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.