NLPDL HW4 Report

Zekai Zhang

11, 2022

1 Configurations

For all the experiments, I adopt 64 as the batch size and 5e-5 as the initial learning rate.

For the AdamW optimizer, the configuration is {lr : 5e-5, weight_decay : 0.0, adam_beta1 : 0.9, adam_beta2 : 0.999, adam_epsilon : 1e-08, max_grad_norm : 1.0}.

Datasets "restaurant_sup", "acl_sup" and "agnews_sup" employs EPOCH = 20, 15 and 10 respectively when trained without adapters. When in adapter version, model train 20 epochs on all datasets.

The source code and dataset is available at https://github.com/violets-blue/NLPDL_hw4. The experiment results are available at https://wandb.ai/justinzzk/huggingface?workspace=user-justinzzk.

2 Learning Curves

All the experiments converged stably under the above configuration. Same experiments are conducted on 5 different seeds to ensure the reliability.

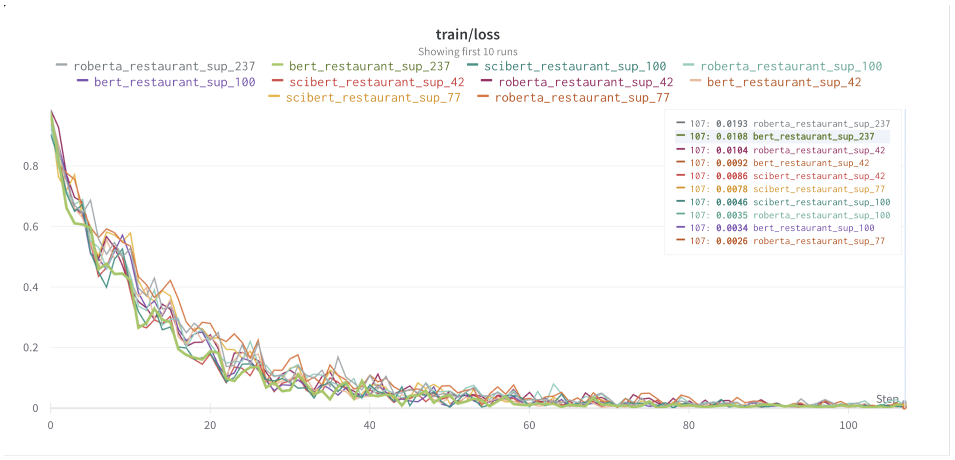


Figure 1: The result of different models on "restaurant_sup".

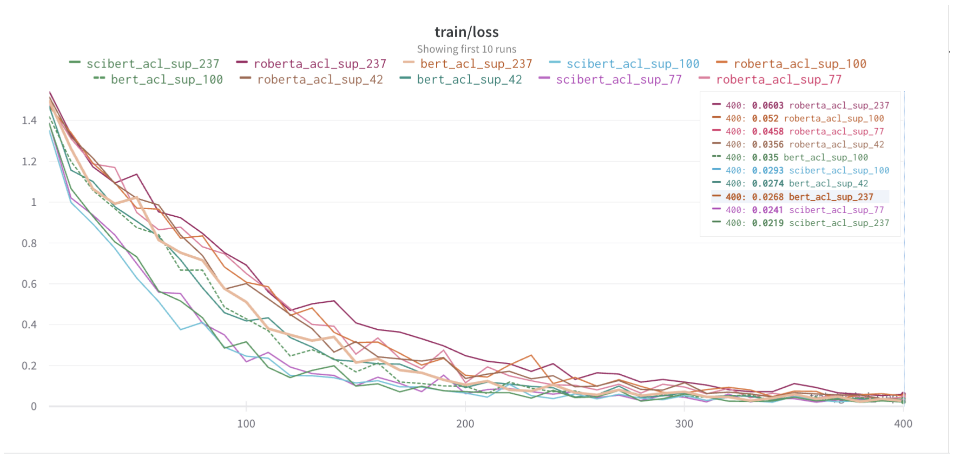


Figure 2: The result of different models on "acl_sup".

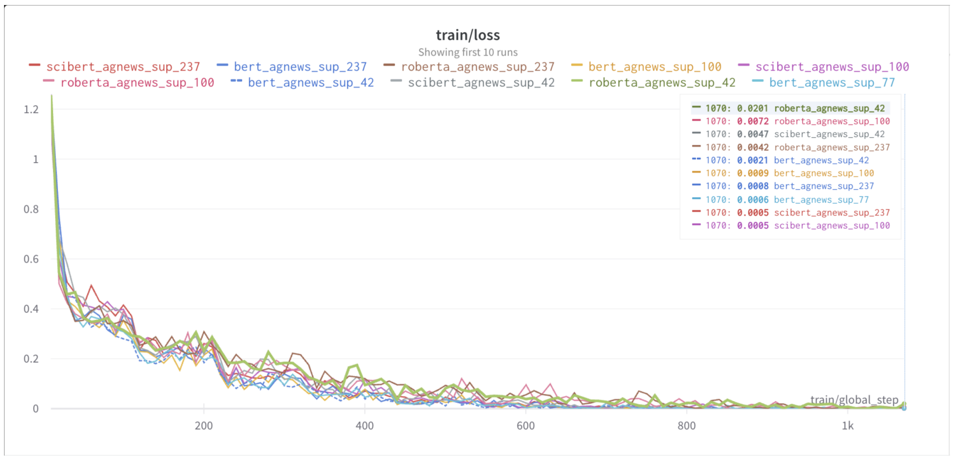
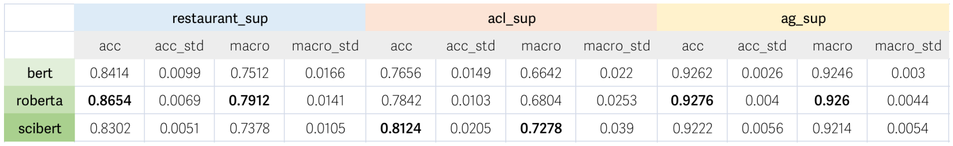


Figure 3: The result of different models on ”agnews_sup”.

3 Results and Analysis

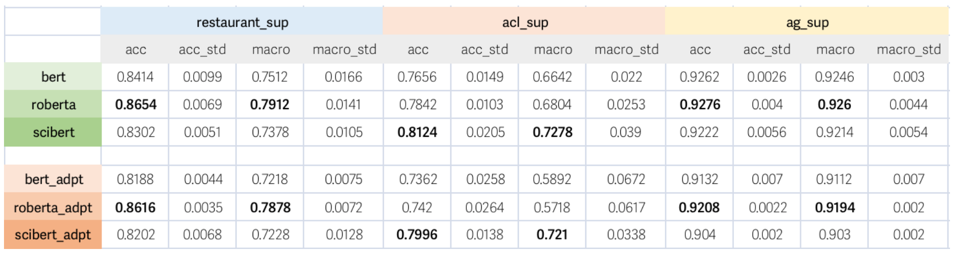


	restaurant_sup				acl_sup				ag_sup			
	acc	acc_std	macro	macro_std	acc	acc_std	macro	macro_std	acc	acc_std	macro	macro_std
bert	0.8414	0.0099	0.7512	0.0166	0.7656	0.0149	0.6642	0.022	0.9262	0.0026	0.9246	0.003
roberta	**0.8654**	0.0069	**0.7912**	0.0141	0.7842	0.0103	0.6804	0.0253	**0.9276**	0.004	**0.926**	0.0044
scibert	0.8302	0.0051	0.7378	0.0105	**0.8124**	0.0205	**0.7278**	0.039	0.9222	0.0056	0.9214	0.0054

Figure 4: Experiment results

Roberta achieves higher score than Bert on all the experiments. Scibert is the best on ”acl_sup” mainly because scibert is pretrained on science datasets, which has less domain gap with the acl dataset. But in general, Roberta preforms the best.

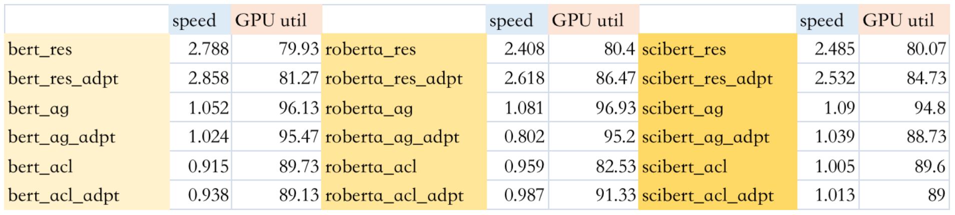
4 Adapter



	restaurant_sup				acl_sup				ag_sup			
	acc	acc_std	macro	macro_std	acc	acc_std	macro	macro_std	acc	acc_std	macro	macro_std
bert	0.8414	0.0099	0.7512	0.0166	0.7656	0.0149	0.6642	0.022	0.9262	0.0026	0.9246	0.003
roberta	**0.8654**	0.0069	**0.7912**	0.0141	0.7842	0.0103	0.6804	0.0253	**0.9276**	0.004	**0.926**	0.0044
scibert	0.8302	0.0051	0.7378	0.0105	**0.8124**	0.0205	**0.7278**	0.039	0.9222	0.0056	0.9214	0.0054
bert_adpt	0.8188	0.0044	0.7218	0.0075	0.7362	0.0258	0.5892	0.0672	0.9132	0.007	0.9112	0.007
roberta_adpt	**0.8616**	0.0035	**0.7878**	0.0072	0.742	0.0264	0.5718	0.0617	**0.9208**	0.0022	**0.9194**	0.002
scibert_adpt	0.8202	0.0068	0.7228	0.0128	**0.7996**	0.0138	**0.721**	0.0338	0.904	0.002	0.903	0.002

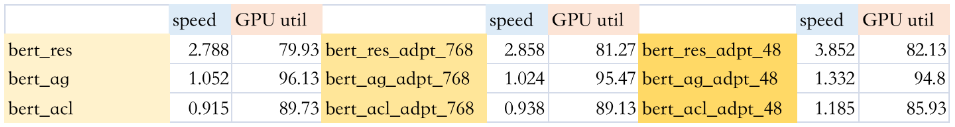
Figure 5: Adapter Experiment results

All experiments experience a slight drop in performance when using less parameters. Bert and Roberta with adapter have comparably larger drops in performance on acl_sup, while Scibert has less performance degration. A possible explanation is that there exist more domain gaps between Bert and Roberta's pretrained data and acl_sup, so they need more parameters to bridge the gaps. However, we are glad to see that the best model on a certain task still performs the best in adapter version, which demonstrates the consistency of our experiments.



	speed	GPU util		speed	GPU util		speed	GPU util
bert_res	2.788	79.93	roberta_res	2.408	80.4	scibert_res	2.485	80.07
bert_res_adpt	2.858	81.27	roberta_res_adpt	2.618	86.47	scibert_res_adpt	2.532	84.73
bert_ag	1.052	96.13	roberta_ag	1.081	96.93	scibert_ag	1.09	94.8
bert_ag_adpt	1.024	95.47	roberta_ag_adpt	0.802	95.2	scibert_ag_adpt	1.039	88.73
bert_acl	0.915	89.73	roberta_acl	0.959	82.53	scibert_acl	1.005	89.6
bert_acl_adpt	0.938	89.13	roberta_acl_adpt	0.987	91.33	scibert_acl_adpt	1.013	89

Figure 6: Adapter Efficiency results

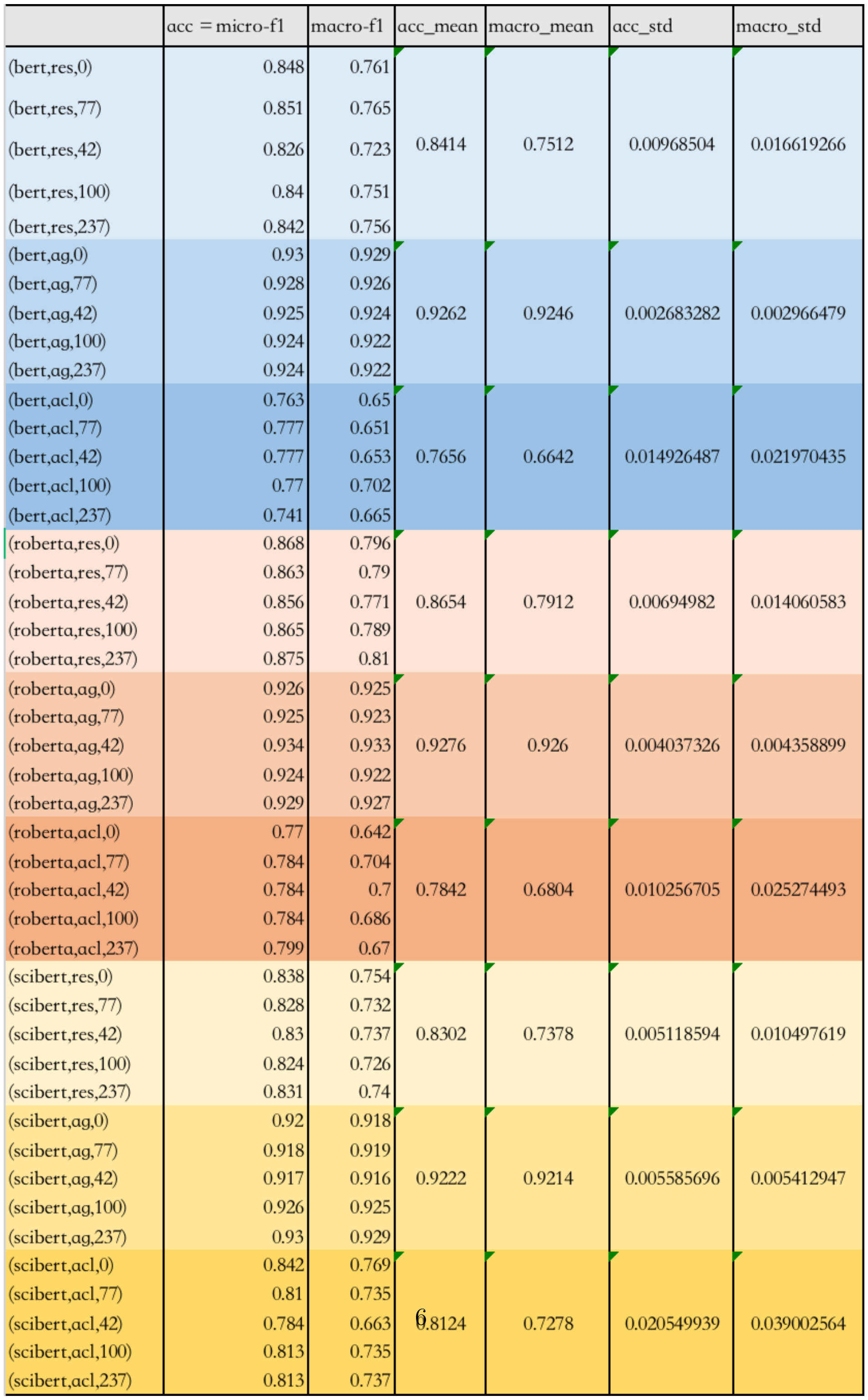


	speed	GPU util		speed	GPU util		speed	GPU util
bert_res	2.788	79.93	bert_res_adpt_768	2.858	81.27	bert_res_adpt_48	3.852	82.13
bert_ag	1.052	96.13	bert_ag_adpt_768	1.024	95.47	bert_ag_adpt_48	1.332	94.8
bert_acl	0.915	89.73	bert_acl_adpt_768	0.938	89.13	bert_acl_adpt_48	1.185	85.93

Figure 7: Adapter Efficiency results

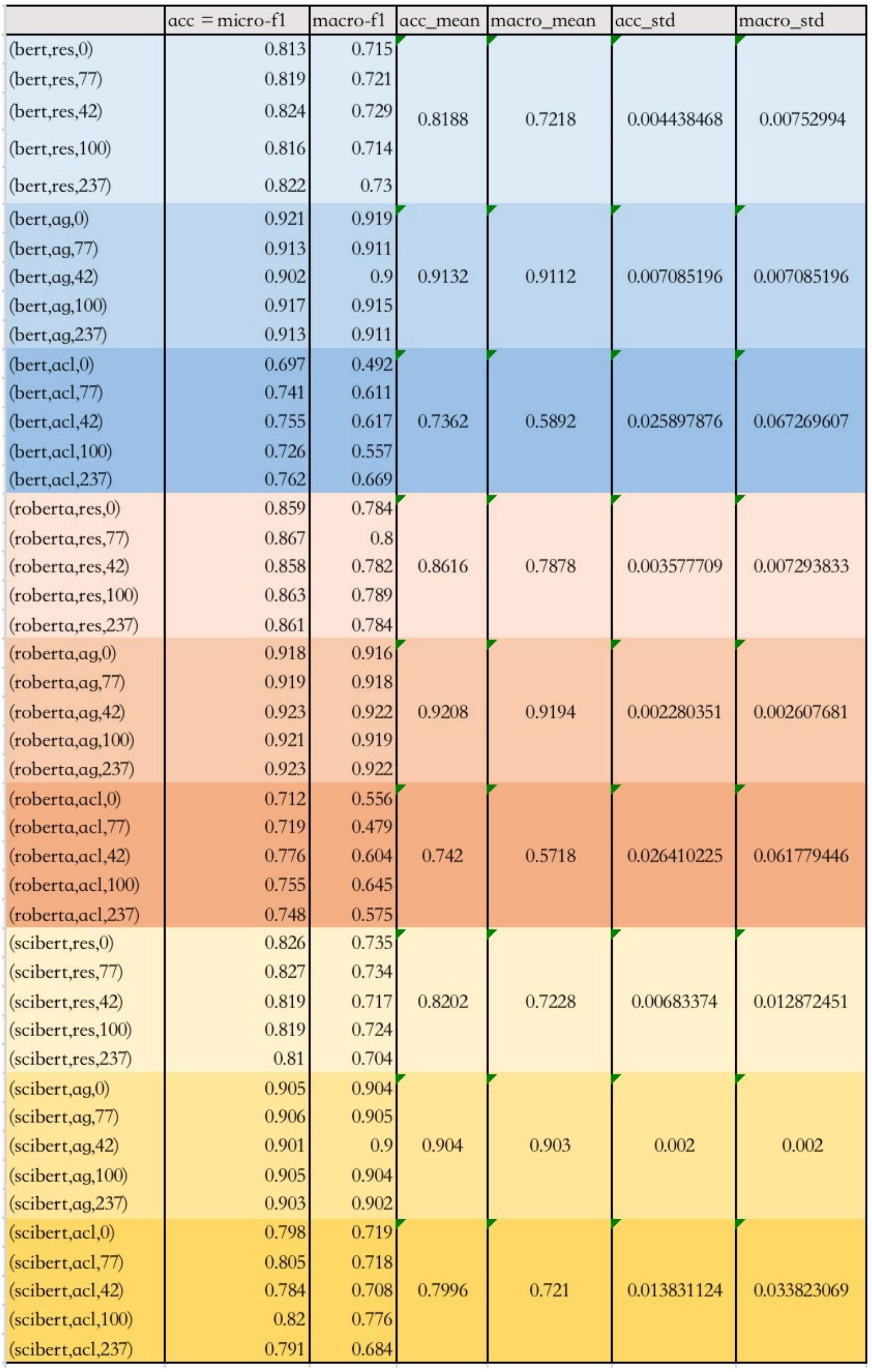
In figure 6, it seems that adapters don't improve the efficiency of training, and sometimes need more time than models without them. It is due to the fact that adapter size is set to a large value 768 during training. We also include results of experiments where adapter size is set to a small value, 48. In figure 7, we can see that model with adapter has a roughly 30% gain in training speed (num of steps per second).

5 Appendix



	acc = micro-f1	macro-f1	acc_mean	macro_mean	acc_std	macro_std
(bert,res,0)	0.848	0.761				
(bert,res,77)	0.851	0.765				
(bert,res,42)	0.826	0.723	0.8414	0.7512	0.00968504	0.016619266
(bert,res,100)	0.84	0.751				
(bert,res,237)	0.842	0.756				
(bert,ag,0)	0.93	0.929				
(bert,ag,77)	0.928	0.926				
(bert,ag,42)	0.925	0.924	0.9262	0.9246	0.002683282	0.002966479
(bert,ag,100)	0.924	0.922				
(bert,ag,237)	0.924	0.922				
(bert,acl,0)	0.763	0.65				
(bert,acl,77)	0.777	0.651				
(bert,acl,42)	0.777	0.653	0.7656	0.6642	0.014926487	0.021970435
(bert,acl,100)	0.77	0.702				
(bert,acl,237)	0.741	0.665				
(roberta,res,0)	0.868	0.796				
(roberta,res,77)	0.863	0.79				
(roberta,res,42)	0.856	0.771	0.8654	0.7912	0.00694982	0.014060583
(roberta,res,100)	0.865	0.789				
(roberta,res,237)	0.875	0.81				
(roberta,ag,0)	0.926	0.925				
(roberta,ag,77)	0.925	0.923				
(roberta,ag,42)	0.934	0.933	0.9276	0.926	0.004037326	0.004358899
(roberta,ag,100)	0.924	0.922				
(roberta,ag,237)	0.929	0.927				
(roberta,acl,0)	0.77	0.642				
(roberta,acl,77)	0.784	0.704				
(roberta,acl,42)	0.784	0.7	0.7842	0.6804	0.010256705	0.025274493
(roberta,acl,100)	0.784	0.686				
(roberta,acl,237)	0.799	0.67				
(scibert,res,0)	0.838	0.754				
(scibert,res,77)	0.828	0.732				
(scibert,res,42)	0.83	0.737	0.8302	0.7378	0.005118594	0.010497619
(scibert,res,100)	0.824	0.726				
(scibert,res,237)	0.831	0.74				
(scibert,ag,0)	0.92	0.918				
(scibert,ag,77)	0.918	0.919				
(scibert,ag,42)	0.917	0.916	0.9222	0.9214	0.005585696	0.005412947
(scibert,ag,100)	0.926	0.925				
(scibert,ag,237)	0.93	0.929				
(scibert,acl,0)	0.842	0.769				
(scibert,acl,77)	0.81	0.735				
(scibert,acl,42)	0.784	0.663	0.8124	0.7278	0.020549939	0.039002564
(scibert,acl,100)	0.813	0.735				
(scibert,acl,237)	0.813	0.737				

Figure 8: Full experiment results



	acc = micro-f1	macro-f1	acc_mean	macro_mean	acc_std	macro_std
(bert,res,0)	0.813	0.715				
(bert,res,77)	0.819	0.721				
(bert,res,42)	0.824	0.729	0.8188	0.7218	0.004438468	0.00752994
(bert,res,100)	0.816	0.714				
(bert,res,237)	0.822	0.73				
(bert,ag,0)	0.921	0.919				
(bert,ag,77)	0.913	0.911				
(bert,ag,42)	0.902	0.9	0.9132	0.9112	0.007085196	0.007085196
(bert,ag,100)	0.917	0.915				
(bert,ag,237)	0.913	0.911				
(bert,acl,0)	0.697	0.492				
(bert,acl,77)	0.741	0.611				
(bert,acl,42)	0.755	0.617	0.7362	0.5892	0.025897876	0.067269607
(bert,acl,100)	0.726	0.557				
(bert,acl,237)	0.762	0.669				
(roberta,res,0)	0.859	0.784				
(roberta,res,77)	0.867	0.8				
(roberta,res,42)	0.858	0.782	0.8616	0.7878	0.003577709	0.007293833
(roberta,res,100)	0.863	0.789				
(roberta,res,237)	0.861	0.784				
(roberta,ag,0)	0.918	0.916				
(roberta,ag,77)	0.919	0.918				
(roberta,ag,42)	0.923	0.922	0.9208	0.9194	0.002280351	0.002607681
(roberta,ag,100)	0.921	0.919				
(roberta,ag,237)	0.923	0.922				
(roberta,acl,0)	0.712	0.556				
(roberta,acl,77)	0.719	0.479				
(roberta,acl,42)	0.776	0.604	0.742	0.5718	0.026410225	0.061779446
(roberta,acl,100)	0.755	0.645				
(roberta,acl,237)	0.748	0.575				
(scibert,res,0)	0.826	0.735				
(scibert,res,77)	0.827	0.734				
(scibert,res,42)	0.819	0.717	0.8202	0.7228	0.00683374	0.012872451
(scibert,res,100)	0.819	0.724				
(scibert,res,237)	0.81	0.704				
(scibert,ag,0)	0.905	0.904				
(scibert,ag,77)	0.906	0.905				
(scibert,ag,42)	0.901	0.9	0.904	0.903	0.002	0.002
(scibert,ag,100)	0.905	0.904				
(scibert,ag,237)	0.903	0.902				
(scibert,acl,0)	0.798	0.719				
(scibert,acl,77)	0.805	0.718				
(scibert,acl,42)	0.784	0.708	0.7996	0.721	0.013831124	0.033823069
(scibert,acl,100)	0.82	0.776				
(scibert,acl,237)	0.791	0.684				

Figure 9: Adapter Full experiment results