SoRA/Adalora Experiments

Reproducing SoRA Results (2 Runs)

Results	**Epoch**	**Accuracy**	**Loss**	**Runtime**	**Samples**
Train	50/50	-	0.083/ 0.09	2870.99/ 3753	2490/ 2490
Evaluation Train	50/ 50	1/ 1	0.0009/ 0.0009	9.98/ 12.54	1000/ 1000
Evaluation	50/ 50	0.877/ 0.877	0.839/ 0.943	1.38/ 1.73	138/138
Test	15/15	0.55/0.55 **(Paper Result: 0.877)**	0.69/0.69	1.40/1.74	138/138

SoRA Datasets Preparation

@sborse3 **@marcocst** Sorry for not making it clear earlier. The results in the paper are from the test set. But this test set differs from the test part of the original dataset from Huggingface. We partition the dataset as follows:

For small datasets (n_samples < 10K), we divide validation set to half, use one half as test set and one half as validation set. For larger datasets (n_samples > 10K), we divide training set into 1K as validation and the rest as training set, keeping the original validation set as the test set. You can find the specific implementation in the `get` function within the `SoRA/src/processor.py` file (Lines 87-106).

Problems Identified/ Roadblocks

Issues Faced in Reproducing Results for SoRA

- **Backbone Model Compatibility:**
  - LLaMA-2-7B not supported by OpenDelta's LoRA configuration
  - OpenDelta supports only models from the Transformers module
- **Roberta-large Implementation:**
  - Tested on BoolQ task (SuperGLUE) in Alora paper
  - No pre-implemented SuperGLUE script available
  - Custom implementation not compatible with original preprocessing script

Problems Identified/ Roadblocks

Issues with Adalora

- **Complex Structure:**
  - Modified Transformer trainer file for fine-tuning with Adalora
  - Multiple supporting files also modified, difficult to identify changes
- **Alternative Method:**
  - Using PEFT/AdaConfig from Hugging Face
  - SFTTrainer does not support custom compute metrics functionality, limiting accuracy calculation
- **Callback Functions:**
  - Need to run evaluation step manually
  - Forward pass, gather logits and labels, then calculate accuracy
  - Evaluation called twice on each run

Problems Identified/ Roadblocks

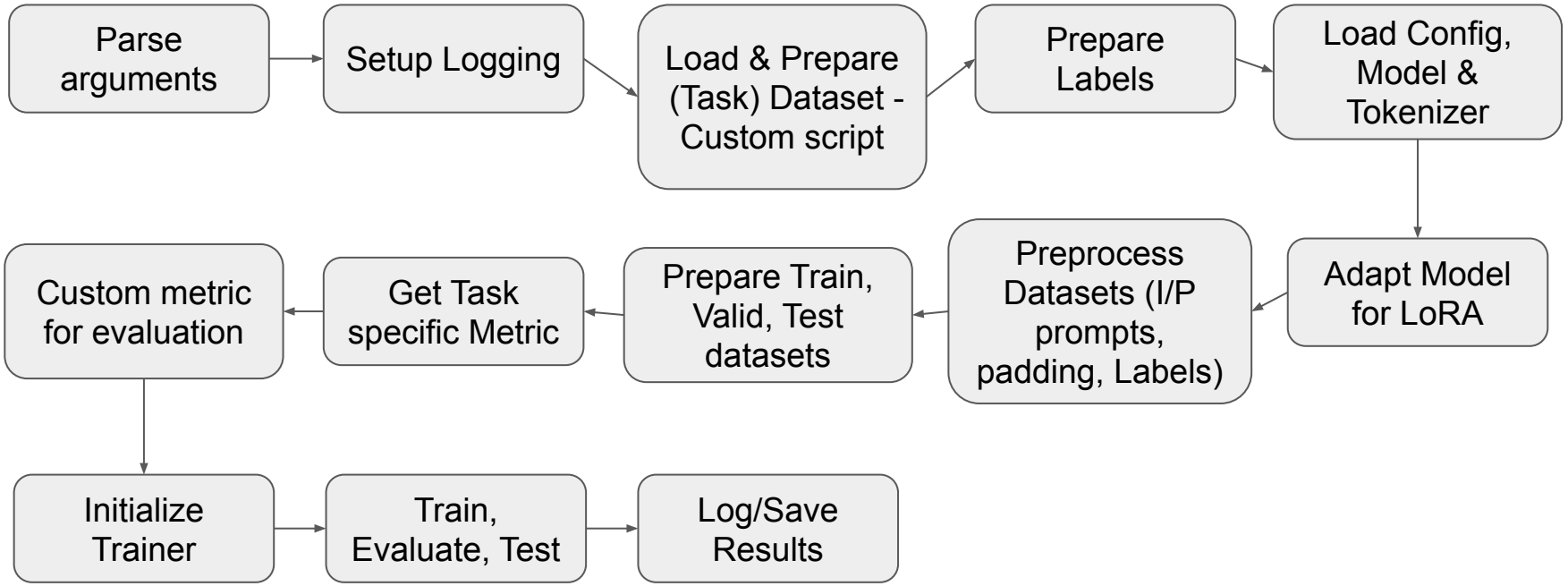
Modularity:

- Different preprocessing steps for each method
- Resulting in long and complicated code files

Potential Steps

1. Implement custom Superglue script - Adapt preprocessing and data loading for SuperGLUE tasks. Ensure compatibility with Sora and adalora
2. Experiment with the original Adalora code

General Idea of Fine tuning using Glue



Limitations

SoRA

- Different logic for data splits
- Hyperparameters were different in original experiment and on github
- Had to run experiments without lambda schedule, was running with schedule
- Had to write custom logic for super glue tasks for reproducing results as per Alora

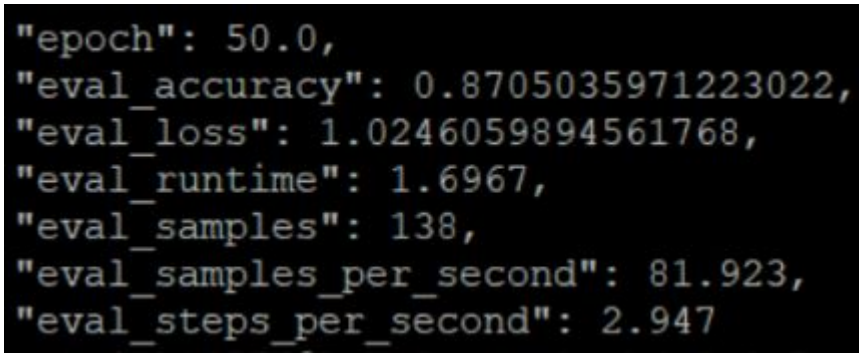
AdaLoRA

- Llama model not compatible with open delta configuration (reproducing results as per Alora)
- Alternate method of using PEFT: Function not supporting metric computation functionality

Reproduction of SoRA Results

Method	#Params	CoLA	SST-2	MRPC	QQP	STS-B	MNLI	QNLI	RTE	Avg.
Fine-Tune	184M	69.21	95.64	89.22	92.05/89.31	91.59	89.98/89.95	93.78	82.49	87.82
Adapter	1.41M	69.00	95.16	89.90	91.45/88.88	92.21	90.11/90.11	93.79	82.44	87.85
Bitfit	0.1M	68.70	94.38	87.16	87.86/84.20	89.71	87.45/87.45	91.90	76.12	85.18
LoRA (r=8)	1.33M	69.73	95.57	89.71	91.95/89.26	91.86	**90.47/90.46**	93.76	85.32	88.38
AdaLoRA	1.27M	70.86	**95.95**	90.22	92.13/88.41	91.39	90.27/90.30	**94.28**	87.36	88.83
SoRA	0.91M	**71.48**	95.64	**91.98**	**92.39/89.87**	**92.22**	90.35/90.38	**94.28**	**87.77**	**89.36**

Original Test results of SoRA and other baselines on the GLUE benchmark
(Avg of 5 runs on different seeds)



Reproduce Test results of SoRA on the GLUE benchmark (RTE task)

Reproduction of AdaLoRA Results

Method	# Params	MNLI m/mm	SST-2 Acc	CoLA Mcc	QQP Acc/F1	QNLI Acc	RTE Acc	MRPC Acc	STS-B Corr	All Ave.
Full FT	184M	89.90/90.12	95.63	69.19	**92.40/89.80**	94.03	83.75	89.46	91.60	88.09
BitFit	0.1M	89.37/89.91	94.84	66.96	88.41/84.95	92.24	78.70	87.75	91.35	86.02
HAdapter	1.22M	90.13/90.17	95.53	68.64	91.91/89.27	94.11	84.48	89.95	91.48	88.12
PAdapter	1.18M	90.33/90.39	95.61	68.77	92.04/89.40	94.29	85.20	89.46	91.54	88.24
LoRA$_{r=8}$	1.33M	90.65/90.69	94.95	69.82	91.99/89.38	93.87	85.20	89.95	91.60	88.34
AdaLoRA	1.27M	**90.76/90.79**	**96.10**	**71.45**	**92.23/89.74**	**94.55**	**88.09**	**90.69**	**91.84**	**89.31**
HAdapter	0.61M	90.12/90.23	95.30	67.87	91.65/88.95	93.76	85.56	89.22	91.30	87.93
PAdapter	0.60M	90.15/90.28	95.53	69.48	91.62/88.86	93.98	84.12	89.22	91.52	88.04
HAdapter	0.31M	90.10/90.02	95.41	67.65	91.54/88.81	93.52	83.39	89.25	91.31	87.60
PAdapter	0.30M	89.89/90.06	94.72	69.06	91.40/88.62	93.87	84.48	89.71	91.38	87.90
LoRA$_{r=2}$	0.33M	90.30/90.38	94.95	68.71	91.61/88.91	94.03	85.56	89.71	**91.68**	88.15
AdaLoRA	0.32M	**90.66/90.70**	**95.80**	**70.04**	**91.78/89.16**	**94.49**	**87.36**	**90.44**	91.63	**88.86**

Original Test results of AdaLoRA and other baselines on the GLUE benchmark (Avg of 5 runs on different seeds)

```
epoch                       =    50.0
eval_accuracy               = 0.8845
eval_loss                   = 1.1843
eval_mem_cpu_alloc_delta    =     0MB
eval_mem_cpu_peaked_delta   =     4MB
eval_mem_gpu_alloc_delta    =     0MB
eval_mem_gpu_peaked_delta   =   377MB
eval_runtime                = 4.7838
eval_samples                =    277
eval_samples_per_second     = 57.904
```

Reproduce Test results of AdaLoRA on the GLUE benchmark (RTE task)

Reproduction of SoRA Results as per Alora

Method	Additional Params		BoolQ	ReCoRD	Squad
	Initial	Final	(acc)	(f1-em)	(f1-em)
Results for RoBERTa-large					
Learned-Adapter	366M	354M	86.8	90.2	88.7
LoRA	3.54M	3.54M	86.9	90.0	88.6
SoRA	708M	3.53M	87.2	90.1	88.7
ALoRA	3.54M	3.42M	87.6	90.7	89.4

Original Test results of SoRA on the SuperGlue benchmark (Avg of 5 runs on different seeds)


```
"epoch": 10.0,
"eval_accuracy": 0.8525993883792049,
"eval_loss": 0.8919866681098938,
"eval_runtime": 42.6053,
"eval_samples": 1635,
"eval_samples_per_second": 38.376,
"eval_steps_per_second": 2.418
```


Reproduce Test results of SoRA on the Superglue benchmark (BoolQ task)