SCAN

Sequence-to-sequence learning with neural networks like LSTM models usually perform poorly on datasets like SCAN where it consists of commands and actions.

1)
One way to approach this problem is to use Neural Synchronous Grammars for Sequence-to-Sequence Learning. In [1] authors explore an alternative, hierarchical approach to sequence-to-sequence learning with latent neural grammars. This work shows that it performs very well on the SCAN dataset. I used this model and trained it on SCAN dataset. I got the following result:

Accuracy: 0.9918

I provided the code in the folder **train_latent-neural-grammers_from_scratch**

For train run:

```
>> python train_scan.py --train_file data/SCAN/tasks_train_length.txt --save_path scan-length.pt
```

for test run:

```
>> python predict_scan.py --data_file data/SCAN/tasks_test_length.txt --model_path scan-length.pt
```

2)
Another way to solve this challenge is to use Large Language Models. The problem is that LLMs are trained on common natural language and may not perform well on dataset like SCAN. So I finetuned an LLM on the SCAN dataset and surprisingly the performance was very good. The code and results are provided in folder

I used this starting code from huggingface and modified it to finetune an LLM (T5ForConditionalGeneration) on a dataset (SCAN). This code needs the dataset in the form of translation in .json files, so I found a .json version of SCAN dataset. This is one line of the dataset:


```
{"translation": {"en": "jump", "mentalese": "I_JUMP"}}
```


The results are inside the results folder. The results for the ‘right’ folder in the dataset are following after 60 epochs of finetuning T5:


```
{
    "epoch": 60.0,
    "test_bleu": 99.9433,
    "test_exact_match_percentage": 0.9964,
    "test_gen_len": 62.0188,
```

```
  "test_loss": 0.00021045177709311247,
  "test_mean_sequence_accuracy": 1.0,
  "test_runtime": 556.3642,
  "test_samples": 4476,
  "test_samples_per_second": 8.045,
  "test_steps_per_second": 0.403,
  "train_loss": 0.0021691048159245187,
  "train_runtime": 27628.6566,
  "train_samples": 15225,
  "train_samples_per_second": 33.063,
  "train_steps_per_second": 1.655
}
Acc = 0.9964253798033958
```


It seems that only 10 epochs are enough to reach a good performance.

As we can see the bleu score, exact_match_percentage, accuracy are all very close 100%. This shows that finetuning an LLM like T5 that has not been trained on datasets like SCAN can be very helpful in solving this kind of challenges.

To run the finetuning script run:

```
>> ./scripts/run_t5_base_scan.sh
```

This script will call run_translation.py code with the desired arguments.

I provided the code in the folder **Fine-tune-T5-LLM-on-SCAN**

[1] Kim, Yoon. "Sequence-to-sequence learning with latent neural grammars." *Advances in Neural Information Processing Systems* 34 (2021): 26302-26317.