

Compositional Generalization Abilities of Neural Sequence Models

This repository contains implementations of the Transformer and LSTM models that we evaluate on compositional generalization benchmarks.

Dependencies

- compatible with python 3.6
- dependencies can be installed using `code/requirements.txt`

Setup

Install VirtualEnv using the following (optional):

```
$ [sudo] pip install virtualenv
```

Create and activate your virtual environment (optional):

```
$ virtualenv -p python3 venv  
$ source venv/bin/activate
```

Install all the required packages:

at `code:`

```
$ pip install -r requirements.txt
```

To create the relevant directories, run the following command in the corresponding directory of that model:

for eg, at `code/transformer:`

```
$ sh setup.sh
```

Then transfer all the data folders to the data subdirectory of that model, which in case is `code/transformer/data/`

Models

The current repository includes implementations of 2 Models:

- Transformer in `code/transformer`
 - Sequence-to-Sequence Transformer Model
- LSTM in `code/lstm`
 - Sequence-to-Sequence LSTM Model

Datasets

We work with the following datasets:

- SCAN add jump split
 - Train Data Size: 13,204 examples
 - Test Data Size: 7,706 examples
 - Available online at: <https://github.com/brendenlake/SCAN>
- Colors
 - Train Data Size: 14 examples
 - Test Data Size: 8 examples
- COGS
 - Train Data Size: 24,155 examples
 - Test Data Size: 21,000 examples
 - Available online at: <https://github.com/najoungkim/COGS>

Usage:

The set of command line arguments available can be seen in the respective `args.py` file. Here, we illustrate running a Transformer on the SCAN add_jump dataset. Follow the same methodology for running any experiment over any model.

Running Transformer Model on SCAN add_jump

at `code/transformer:`

```
$ python -m src.main -mode train -pretrained_model_name none -  
finetune_data_voc none -dataset add_jump -epochs 150 -no-save_model -no-  
show_train_acc -embedding random -no-freeze_emb -no-freeze_transformer -d_model  
128 -d_ff 256 -decoder_layers 3 -encoder_layers 3 -heads 2 -batch_size 64 -lr  
0.0005 -emb_lr 0.0005 -dropout 0.1 -run_name run_add_jump_transformer -gpu 0
```