TensorFlow Requirement 1.x TensorFlow 2 Not Supported ×

LFADS - Latent Factor Analysis via Dynamical Systems

This code implements the model from the paper "LFADS - Latent Factor Analysis via Dynamical Systems". It is a sequential variational auto-encoder designed specifically for investigating neuroscience data, but can be applied widely to any time series data. In an unsupervised setting, LFADS is able to decompose time series data into various factors, such as an initial condition, a generative dynamical system, control inputs to that generator, and a low dimensional description of the observed data, called the factors. Additionally, the observation model is a loss on a probability distribution, so when LFADS processes a dataset, a denoised version of the dataset is also created. For example, if the dataset is raw spike counts, then under the negative log-likelihood loss under a Poisson distribution, the denoised data would be the inferred Poisson rates.

Prerequisites

The code is written in Python 2.7.6. You will also need:

- **TensorFlow** version 1.5 (install) -
- **NumPy, SciPy, Matplotlib** (install SciPy stack, contains all of them)
- **h5py** (install)

Getting started

Before starting, run the following:

```
$ export PYTHONPATH=$PYTHONPATH:/path/to/your/directory/lfads/
```

where "path/to/your/directory" is replaced with the path to the LFADS repository (you can get this path by using the `pwd` command). This allows the nested directories to access modules from their parent directory.

Generate synthetic data

In order to generate the synthetic datasets first, from the top-level lfads directory, run:

```
$ cd synth_data
$ ./run_generate_synth_data.sh
$ cd ..
```

These synthetic datasets are provided 1. to gain insight into how the LFADS algorithm operates, and 2. to give reasonable starting points for analyses you might be interested for your own data.

Train an LFADS model

Now that we have our example datasets, we can train some models! To spin up an LFADS model on the synthetic data, run any of the following commands. For the examples that are in the paper, the important hyperparameters are roughly replicated. Most hyperparameters are insensitive to small changes or won't ever be changed unless you want a very fine level of control. In the first example, all hyperparameter flags are enumerated for easy copy-pasting, but for the rest of the examples only the most important flags (~the first 9) are specified for brevity. For a full list of flags, their descriptions, and their default values, refer to the top of `run_lfads.py`. Please see Table 1 in the Online Methods of the associated paper for definitions of the most important hyperparameters.

```
# Run LFADS on chaotic rnn data with no input pulses (g = 1.5) with spiking noise
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_no_inputs \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_no_inputs \
--co_dim=0 \
--factors_dim=20 \
--ext_input_dim=0 \
--controller_input_lag=1 \
--output_dist=poisson \
--do_causal_controller=false \
--batch_size=128 \
--learning_rate_init=0.01 \
--learning_rate_stop=1e-05 \
--learning_rate_decay_factor=0.95 \
--learning_rate_n_to_compare=6 \
--do_reset_learning_rate=false \
--keep_prob=0.95 \
--con_dim=128 \
--gen_dim=200 \
--ci_enc_dim=128 \
--ic_dim=64 \
--ic_enc_dim=128 \
--ic_prior_var_min=0.1 \
--gen_cell_input_weight_scale=1.0 \
--cell_weight_scale=1.0 \
--do_feed_factors_to_controller=true \
--kl_start_step=0 \
--kl_increase_steps=2000 \
--kl_ic_weight=1.0 \
--l2_con_scale=0.0 \
--l2_gen_scale=2000.0 \
--l2_start_step=0 \
--l2_increase_steps=2000 \
--ic_prior_var_scale=0.1 \
--ic_post_var_min=0.0001 \
--kl_co_weight=1.0 \
--prior_ar_nvar=0.1 \
--cell_clip_value=5.0 \
--max_ckpt_to_keep_lve=5 \
--do_train_prior_ar_atau=true \
--co_prior_var_scale=0.1 \
--csv_log=fitlog \
--feedback_factors_or_rates=factors \
--do_train_prior_ar_nvar=true \
--max_grad_norm=200.0 \
--device=gpu:0 \
--num_steps_for_gen_ic=100000000 \
--ps_nexamples_to_process=100000000 \
--checkpoint_name=lfads_vae \
--temporal_spike_jitter_width=0 \
--checkpoint_pb_load_name=checkpoint \
--inject_ext_input_to_gen=false \
--co_mean_corr_scale=0.0 \
--gen_cell_rec_weight_scale=1.0 \
--max_ckpt_to_keep=5 \
--output_filename_stem="" \
--ic_prior_var_max=0.1 \
--prior_ar_atau=10.0 \
--do_train_io_only=false \
--do_train_encoder_only=false

# Run LFADS on chaotic rnn data with no input pulses (g = 1.5) with Gaussian noise
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=gaussian_chaotic_rnn_no_inputs \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_inputs_g2p5 \
--co_dim=1 \
--factors_dim=20 \
--output_dist=gaussian


# Run LFADS on chaotic rnn data with input pulses (g = 2.5)
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_inputs_g2p5 \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_inputs_g2p5 \
--co_dim=1 \
--factors_dim=20 \
--output_dist=poisson

# Run LFADS on multi-session RNN data
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_multisession \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_multisession \
--factors_dim=10 \
--output_dist=poisson

# Run LFADS on integration to bound model data
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=itb_rnn \
--lfads_save_dir=/tmp/lfads_itb_rnn \
--co_dim=1 \
--factors_dim=20 \
--controller_input_lag=0 \
--output_dist=poisson

# Run LFADS on chaotic RNN data with labels
```

```
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnns_labeled \
--lfads_save_dir=/tmp/lfads_chaotic_rnns_labeled \
--co_dim=0 \
--factors_dim=20 \
--controller_input_lag=0 \
--ext_input_dim=1 \
--output_dist=poisson

# Run LFADS on chaotic rnn data with no input pulses (g = 1.5) with Gaussian noise
$ python run_lfads.py --kind=train \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_no_inputs \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_no_inputs \
--co_dim=0 \
--factors_dim=20 \
--ext_input_dim=0 \
--controller_input_lag=1 \
--output_dist=gaussian \
```

Tip: If you are running LFADS on GPU and would like to run more than one model concurrently, set the `--allow_gpu_growth=True` flag on each job, otherwise one model will take up the entire GPU for performance purposes. Also, one needs to install the TensorFlow libraries with GPU support.

Visualize a training model

To visualize training curves and various other metrics while training and LFADS model, run the following command on your model directory. To launch a tensorboard on the chaotic RNN data with input pulses, for example:

```
tensorboard --logdir=/tmp/lfads_chaotic_rnn_inputs_g2p5
```

Evaluate a trained model

Once your model is finished training, there are multiple ways you can evaluate it. Below are some sample commands to evaluate an LFADS model trained on the chaotic rnn data with input pulses (g = 2.5). The key differences here are setting the `--kind` flag to the appropriate mode, as well as the `--checkpoint_pb_load_name` flag to `checkpoint_lve` and the `--batch_size` flag (if you'd like to make it larger or smaller). All other flags should be the same as used in training, so that the same model architecture is built.

```
# Take samples from posterior then average (denoising operation)
$ python run_lfads.py --kind=posterior_sample_and_average \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_inputs_g2p5 \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_inputs_g2p5 \
--co_dim=1 \
--factors_dim=20 \
--batch_size=1024 \
--checkpoint_pb_load_name=checkpoint_lve
```

```
# Sample from prior (generation of completely new samples)
$ python run_lfads.py --kind=prior_sample \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_inputs_g2p5 \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_inputs_g2p5 \
--co_dim=1 \
--factors_dim=20 \
--batch_size=50 \
--checkpoint_pb_load_name=checkpoint_lve

# Write down model parameters
$ python run_lfads.py --kind=write_model_params \
--data_dir=/tmp/rnn_synth_data_v1.0/ \
--data_filename_stem=chaotic_rnn_inputs_g2p5 \
--lfads_save_dir=/tmp/lfads_chaotic_rnn_inputs_g2p5 \
--co_dim=1 \
--factors_dim=20 \
--checkpoint_pb_load_name=checkpoint_lve
```

Contact

File any issues with the issue tracker. For any questions or problems, this code is maintained by @sussillo and @jazcollins.