Neural Monkey

Dušan Variš

Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University

September 4, 2018

Neural Monkey

- Toolkit for training neural models for sequence-to-sequence tasks
- Python 3
- Tensorflow 1.4
- GPU support using CUDA, cuDNN
- Modularity of parts of the computational graphs
  - Easy composition of new models
  - Fast prototyping of new experiments

Installation

1. Install the prerequisites:

```
$ git clone https://github.com/ufal/neuralmonkey -b mtm18
$ cd neuralmonkey

$ source path/to/virtualenv/bin/activate

# For CPU-only version:
(virtualenv)$ pip install numpy
(virtualenv)$ pip install --upgrade -r requirements.txt

# For GPU-enabled version:
(virtualenv)$ pip install numpy
(virtualenv)$ pip install --upgrade -r requirements-gpu.txt
```

2. Download the data:

```
$ ./tutorial/get_data.sh
```

Task 1: Language Model

1. Run `bin/neuralmonkey-train tutorial/01-language_model.ini`
2. In a separate terminal, run `tensorboard --logdir=tutorial/ --port=6006`
3. Open `localhost:6006` in your web browser

Experiment Directory structure

```
[main]
...
output="tutorial/language_model"
...
```

- args - Executed command
- original.ini - Original experiment configuration
- experiment.ini - Executed experiment configuration
- experiment.log - Experiment logfile
- variables.data - Model variables
- events.out.tfevents - TensorFlow events file for TensorBoard

Sidestep: Model Workflow

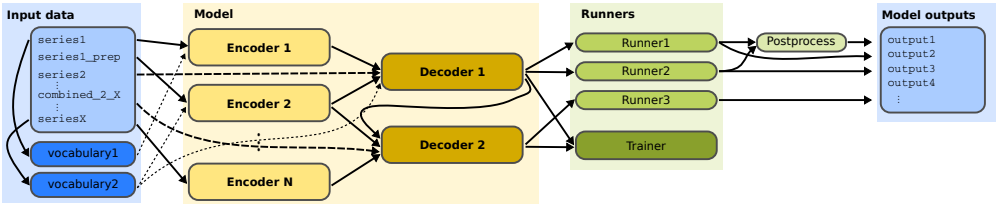

Figure 1: Model workflow.

Each step of the workflow can be modified/expanded by changing a corresponding section in the model configuration file.

Configuration Files

INI file syntax

- Sections defining separate Python objects
- Key-value pairs separated by '='
- Values can be atomic (int, boolean, string) or composite (list, objects)
- Sections are interpreted as Python dictionaries
- Variable substitution:
  - Defined in the [*vars*] section
  - `$variable` - for standalone reference
  - {`variable`} - for reference inside a string

Task 1: Configuration - Main Section

```
[main]
name="Language Model"
output="{exp_prefix}/language_model"

batch_size=64
epochs=100

tf_manager=<tf_manager>
train_dataset=<train_data>
val_dataset=<val_data>
trainer=<trainer>
runners=[<greedy_runner>, <perplexity_runner>]

evaluation=[..., ("perplexity", "target", <perplexity>)]

logging_period=50
validation_period=500
```

Task 1: Configuration - Data Specification

```
[train_data]
class=dataset.load_dataset_from_files
s_target="{data_prefix}/ar2en-train.tgt.txt"
preprocessors=[("target", "target_char", ...)]
lazy=True

[val_data]
...

[vocabulary]
class=vocabulary.from_dataset
datasets=[<train_data>]
series_ids=["target_char"]
max_size=500
save_file="{exp_prefix}/language_model/vocabulary.txt"
```

Task 1: Configuration - Model Definition

```
[decoder]
class=decoders.Decoder
name="decoder"

encoders=[]
vocabulary=<vocabulary>
data_id="target_char"

embedding_size=128
rnn_size=256
rnn_cell="LSTM"
max_output_len=30
dropout_keep_prob=$dropout
```

Task 1: Configuration - Trainers and Runners

```
[trainer]
class=trainers.cross_entropy_trainer.CrossEntropyTrainer
decoders=[<decoder>]
optimizer=<adam>
clip_norm=1.0

[adam]
class=tf.contrib.opt.LazyAdamOptimizer
...

[greedy_runner]
class=runners.runner.GreedyRunner
decoder=<decoder>
output_series="target"
postprocess=processors.helpers.postprocess_char_based

[perplexity_runner]
...
```

Task 2: Machine Translation

1. Run `bin/neuralmonkey-train tutorial/02b-seq2seq-attention.ini`
2. Check the tensorboard again

Task 2: Configuration - Extending Task 1

```
[train_data]
...
s_source="{data_prefix}/ar2en-train.src.txt"
...
preprocessors=[("target", "target_char", ...), ("source", "source_char", ...)]
lazy=True

[val_data]
...

[vocabulary]
...
series_ids=["source_char", "target_char"]
...
```

Task 2: Configuration - Adding Encoder

```
[input_sequence]
class=model.sequence.EmbeddedSequence
max_length=30
embedding_size=128
data_id="source_char"
vocabulary=<vocabulary>

[encoder]
class=encoders.recurrent.RecurrentEncoder
input_sequence=<input_sequence>
rnn_size=64
rnn_cell="LSTM"
rnn_direction="bidirectional"
dropout_keep_prob=$dropout
```

Task 2: Configuration - Adding Encoder

```
[decoder]
...
encoders=[<encoder>]
attentions=[<encoder_attention>]
...
#embedding_size=128
embeddings_source=<input_sequence>

[encoder_attention]
class=attention.Attention
encoder=<encoder>
```

Task 2: Using the Trained Model

- `bin/neuralmonkey-run tutorial/02b-seq2seq-attention.ini tutorial/02-seq2seq_data.ini`
- tutorial/02b-seq2seq-attention.ini - Model definition
- tutorial/02-seq2seq_data.ini - Inference-time data definition

Task 2: Using the Model - Data Configuration

```
[vars]
proj_prefix="."
exp_prefix="{proj_prefix}/tutorial"
data_prefix="{proj_prefix}/tutorial-data"

[main]
test_datasets=[<val_data>]

[vocabulary]
class=vocabulary.from_t2t_vocabulary
path="{exp_prefix}/seq2seq_attention/vocabulary.txt"

[val_data]
...
s_target_out="seq2seq.val.out.txt"
```

Task 3: Multimodal Translation

1. Run `bin/neuralmonkey-train tutorial/03-multimodal.ini`

Task 3: Configuration - Extending Task 2

```
[train_data]
...
s_images=("{data_prefix}/dummy-train.txt", <imagenet_reader>)
...

[imagenet_reader]
class=readers.numpy_reader.from_file_list
prefix="{data_prefix}/images"
suffix=".npz"
```

Task 3: Configuration - Adding Image Encoder

```
[image_encoder]
class=encoders.numpy_stateful_filler.SpatialFiller
input_shape=[8, 8, 2048]
data_id="images"
ff_hidden_dim=256
projection_dim=128

[decoder]
...
encoders=[<encoder>, <image_encoder>]
attentions=[<encoder_attention>, <image_attention>]
...

[image_attention]
class=attention.Attention
encoder=<image_encoder>
```

Exercises

1. Replace the RNN in the MT scenario with a Transformer architecture (use the `tutorial/task01-transformer.ini` config and fill in the TODO sections).
2. Replace the greedy decoding in the MT scenario with a beamsearch decoding (use the `tutorial/task02-beamsearch.ini` config and fill in the TODO sections).