TensorFlow Requirement 1.x TensorFlow 2 Not Supported ✗

Adversarial Text Classification

Code for *Adversarial Training Methods for Semi-Supervised Text Classification* and *Semi-Supervised Sequence Learning*.

Requirements

- TensorFlow >= v1.3

End-to-end IMDB Sentiment Classification

Fetch data

```
$ wget http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz \
    -O /tmp/imdb.tar.gz
$ tar -xf /tmp/imdb.tar.gz -C /tmp
```

The directory `/tmp/aclImdb` contains the raw IMDB data.

Generate vocabulary

```
$ IMDB_DATA_DIR=/tmp/imdb
$ python gen_vocab.py \
    --output_dir=$IMDB_DATA_DIR \
    --dataset=imdb \
    --imdb_input_dir=/tmp/aclImdb \
    --lowercase=False
```

Vocabulary and frequency files will be generated in `$IMDB_DATA_DIR`.

Generate training, validation, and test data

```
$ python gen_data.py \
    --output_dir=$IMDB_DATA_DIR \
    --dataset=imdb \
    --imdb_input_dir=/tmp/aclImdb \
    --lowercase=False \
    --label_gain=False
```

`$IMDB_DATA_DIR` contains TFRecords files.

Pretrain IMDB Language Model

```
$ PRETRAIN_DIR=/tmp/models/imdb_pretrain
$ python pretrain.py \
    --train_dir=$PRETRAIN_DIR \
    --data_dir=$IMDB_DATA_DIR \
    --vocab_size=87007 \
    --embedding_dims=256 \
    --rnn_cell_size=1024 \
    --num_candidate_samples=1024 \
    --batch_size=256 \
    --learning_rate=0.001 \
    --learning_rate_decay_factor=0.9999 \
    --max_steps=100000 \
    --max_grad_norm=1.0 \
    --num_timesteps=400 \
    --keep_prob_emb=0.5 \
    --normalize_embeddings
```

`$PRETRAIN_DIR` contains checkpoints of the pretrained language model.

Train classifier

Most flags stay the same, save for the removal of candidate sampling and the addition of `pretrained_model_dir`, from which the classifier will load the pretrained embedding and LSTM variables, and flags related to adversarial training and classification.

```
$ TRAIN_DIR=/tmp/models/imdb_classify
$ python train_classifier.py \
    --train_dir=$TRAIN_DIR \
    --pretrained_model_dir=$PRETRAIN_DIR \
    --data_dir=$IMDB_DATA_DIR \
    --vocab_size=87007 \
    --embedding_dims=256 \
    --rnn_cell_size=1024 \
    --cl_num_layers=1 \
    --cl_hidden_size=30 \
    --batch_size=64 \
    --learning_rate=0.0005 \
    --learning_rate_decay_factor=0.9998 \
    --max_steps=15000 \
    --max_grad_norm=1.0 \
    --num_timesteps=400 \
    --keep_prob_emb=0.5 \
    --normalize_embeddings \
    --adv_training_method=vat \
    --perturb_norm_length=5.0
```

Evaluate on test data

```
$ EVAL_DIR=/tmp/models/imdb_eval
$ python evaluate.py \
    --eval_dir=$EVAL_DIR \
    --checkpoint_dir=$TRAIN_DIR \
    --eval_data=test \
    --run_once \
    --num_examples=25000 \
    --data_dir=$IMDB_DATA_DIR \
    --vocab_size=87007 \
    --embedding_dims=256 \
    --rnn_cell_size=1024 \
    --batch_size=256 \
    --num_timesteps=400 \
    --normalize_embeddings
```

Code Overview

The main entry points are the binaries listed below. Each training binary builds a `VatxtModel`, defined in `graphs.py`, which in turn uses graph building blocks defined in `inputs.py` (defines input data reading and parsing), `layers.py` (defines core model components), and `adversarial_losses.py` (defines adversarial training losses). The training loop itself is defined in `train_utils.py`.

Binaries

- Pretraining: `pretrain.py`
- Classifier Training: `train_classifier.py`
- Evaluation: `evaluate.py`

Command-Line Flags

Flags related to distributed training and the training loop itself are defined in `train_utils.py`.

Flags related to model hyperparameters are defined in `graphs.py`.

Flags related to adversarial training are defined in `adversarial_losses.py`.

Flags particular to each job are defined in the main binary files.

Data Generation

- Vocabulary generation: `gen_vocab.py`
- Data generation: `gen_data.py`

Command-line flags defined in `document_generators.py` control which dataset is processed and how.

Contact for Issues

- Ryan Sepassi, @rsepassi
- Andrew M. Dai, @a-dai adai@google.com
- Takeru Miyato, @takerum (Original implementation)