CSCI 6908 Project

Author: Yixiao Yuan B00785417

Building a QA system

This is a PyTorch implementation of character-level word embeddings, Self-attention and QANet for SQuAD 2.0. This repository is built upon the starter code for Stanford's CS224N Default Project.

Setup

1. Make sure you have Miniconda installed
    1. Conda is a package manager that sandboxes your project's dependencies in a virtual environment
    2. Miniconda contains Conda and its dependencies with no extra packages by default (as opposed to Anaconda, which installs some extra packages)
2. Creates a Conda environment called `squad`

    Filename: env_squad_create.bat

```
REM # The conda virtual environment for CSCI 6908 Project Building a QA system
REM # Note: This environment uses GPU acceleration

call conda create -n squad python=3.8.16 --yes

call conda activate squad

call conda install --yes -c conda-forge ujson numpy pip tqdm urllib3
call conda install --yes -c conda-forge spacy=2.3.7

REM call conda update --all --yes -c conda-forge

REM # install tensorflow
pip install --upgrade --no-cache-dir tensorflow==2.10.1

REM # install pytorch
REM call conda install pytorch torchvision torchaudio pytorch-cuda=11.7 -c pytorch -c
nvidia --yes
pip install --upgrade --no-cache-dir torch torchvision torchaudio --extra-index-url
https://download.pytorch.org/whl/cu117


REM # install tensorboardX https://github.com/lanpa/tensorboardX
pip install --upgrade --no-cache-dir torch-tb-profiler
pip install --upgrade --no-cache-dir tensorboardX
REM # You can optionally install crc32c to speed up.
REM pip install --upgrade --no-cache-dir crc32c
```

```
REM # Starting from tensorboardX 2.1, You need to install soundfile for the
add_audio() function (200x speedup).
REM # pip install --upgrade --no-cache-dir soundfile

REM # Removed unused packages, temp files
call conda clean --all --yes

pause
```

3. Run `conda activate squad`
    1. This activates the `squad` environment
    2. Do this each time you want to write/test your code
4. Run `python setup.py`
    1. This downloads SQuAD 2.0 training and dev sets, as well as the GloVe 300-dimensional word vectors (840B)
    2. This also pre-processes the dataset for efficient data loading
    3. For a MacBook Pro on the Stanford network, `setup.py` takes around 30 minutes total
5. Browse the code in `train.py`
    1. The `train.py` script is the entry point for training a model. It reads command-line arguments, loads the SQuAD dataset, and trains a model.
    2. You may find it helpful to browse the arguments provided by the starter code. Either look directly at the `parser.add_argument` lines in the source code, or run `python train.py -h`.

Train and test

1. Train and test the Baseline BiDAF model.

```
# Train the model.
python train.py --name BiDAF --hidden_size 128 --model BiDAF

# Test the model.
python test.py --split dev --load_path PATH --name BiDAF --model BiDAF
# PATH is a path to a checkpoint (e.g., save/train/model-01/best.pth.tar)
```

2. Train and test the BiDAF model with character-level word embeddings.

```
# Train the model.
python train.py --name BiDAFwithChar --hidden_size 128 --model BiDAFwithChar

# Test the model.
python test.py --split dev --load_path PATH --name BiDAFwithChar --model BiDAFwithChar
# PATH is a path to a checkpoint (e.g., save/train/model-01/best.pth.tar)
```

3. Train and test the BiDAF + Self-attention model with character-level word embeddings.

```
# Train the model.
python train.py --name BiDAFwithCharSelfAtt --hidden_size 128 --model BiDAFwithCharSelfAtt

# Test the model.
python test.py --split dev --load_path PATH --name BiDAFwithCharSelfAtt --model
BiDAFwithCharSelfAtt
# PATH is a path to a checkpoint (e.g., save/train/model-01/best.pth.tar)
```

4. Train and test the QANET model.

```
# Train the model.
python train.py --name QANET --batch_size 16 --model QANET --lr 0.001 --drop_prob 0.1

# Test the model.
python test.py --split dev --load_path PATH --name QANET --model QANET
# PATH is a path to a checkpoint (e.g., save/train/model-01/best.pth.tar)
```

Performance

Model	HiddenSize	BatchSize	NLL	F1	EM	AvNA
Baseline BiDAF model.	128	64	3.26	61.07	57.69	68.22
BiDAF model with character-level word embeddings.	128	64	3.16	62.51	59.05	69.70
BiDAF + Self-attention model with character-level word embeddings.	128	64	2.73	66.48	62.90	72.41
QANET model	128	16	2.62	69.25	65.70	75.53

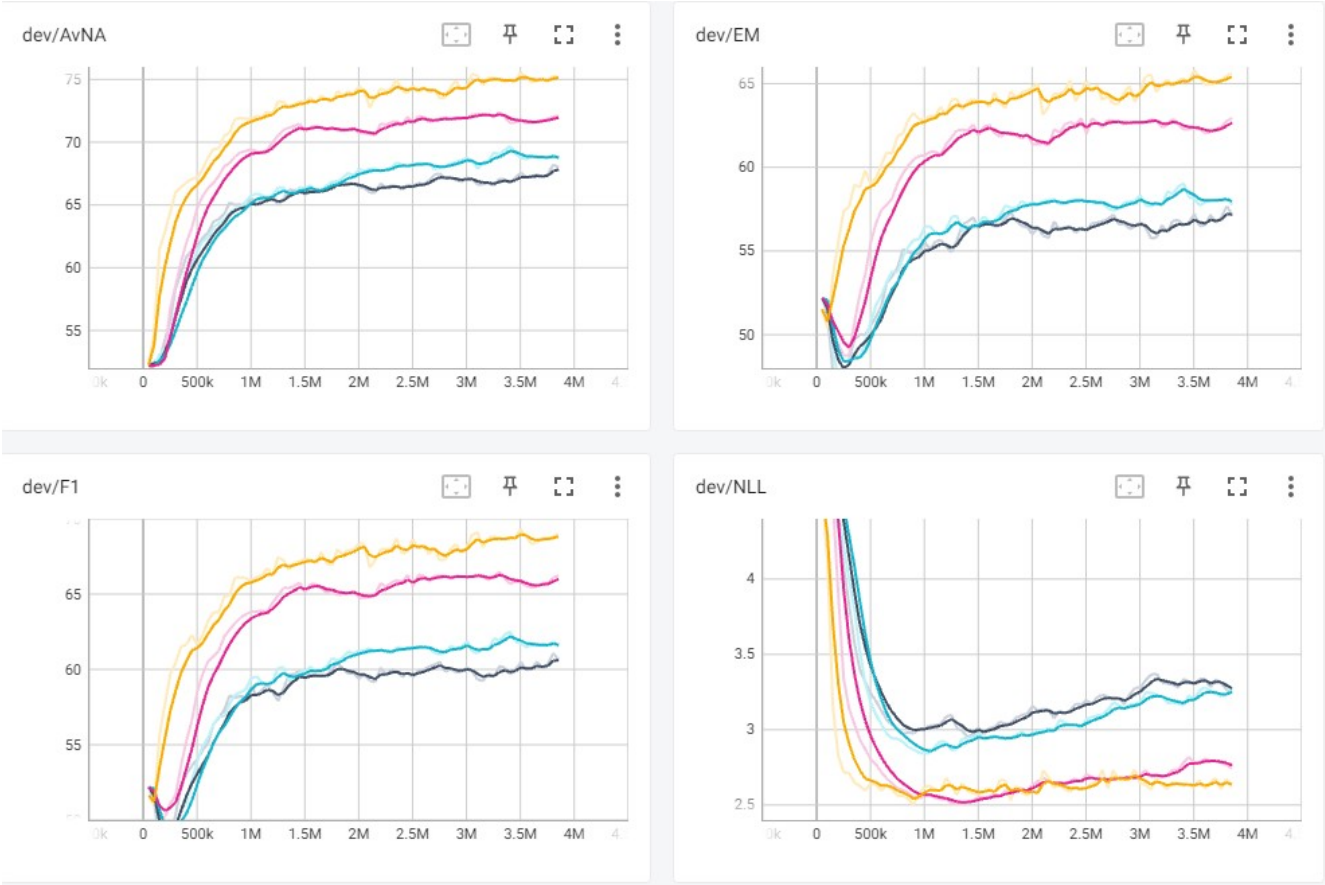
On the Nvidia 1060 6GB graphics card, our training the BiDAF model took about 6 hours to complete, and the QANET model took about 24 hours to complete.

Tensorboard

1. Run tensorboard

```
# Run tensorboard for visualisation
tensorboard --logdir ./log/
```

2. Results



BiDAF model.

BiDAF model with character-level embeddings.

BiDAF + Self-attention with character-level embeddings.

QANET model.