ViVi_NMT: An Open Source Toolkit for Neural Machine Translation

Center for Speech and Language Technologies, Tsinghua University

August, 2017

1 Introduction

ViVi_NMT(v1.0) is a neural machine translation toolkit developed by Center for Speech and Language Technologies.

On top of Tensorflow, this toolkit mainly reproduced the RNNsearch model (Bahdanau et al., 2015).

Note that, this code is modified on the ViVi_NMT(v0.10) code（upgrade code to make it support tensorflow 1.0 and later version）, and the 'multi_bleu.perl' script is downloaded from Moses.

2 Installation

2.1 System Requirements

ViVi_NMT supports Linux or MacOS, and requires python 2.7. Besides, we highly recommand running ViVi_NMT on GPU servers.

2.2 Installing Prerequisites

2.2.1 CUDA Environment

If you choose to use CPU, please skip this step.

Assume you want to run ViVi_NMT on NVIDIA GPUs and the CUDA toolkit version 8.0 has been installed in "/usr/local/cuda-8.0/", then environment variables need to be set:

```
export PATH=/usr/local/cuda-8.0/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda-8.0/lib64:$LD_LIBRARY_PATH
```

2.2.2 Tensorflow 1.0

To have tensorflow 1.0 installed, serval methods can be applied. Here, we only introduce the installation through virtualenv. And we install the tensorflow-gpu, if you choose to use CPU, please install tensorflow of cpu.

```
pip install virtualenv --user
virtualenv --system-site-packages tf1.0
source tf1.0/bin/activate
export
TF_BINARY_URL=https://mirrors.tuna.tsinghua.edu.cn/tensorflow/linux/gpu/tensorflow_gpu-1.0.
0-cp27-none-linux_x86_64.whl
pip install --upgrade $TF_BINARY_URL
```

Subsequently,get into python console, and import tensorflow. If no error is encountered, the installation is successful.

2.2.3 Installing ViVi_NMT

The source code of ViVi_NMT is available at GitHub.If want to use ViVi_NMT, you just need download and unpack it.

Entering the ViVi_NMT folder,you will find two folders(data,train) and several files:

- data:stores training, validation, and test datasets.
- train:saves models.
- LICENSE:license statement.
- Manual.pdf:this document.
- Other files:the source code and README file.

3 Running ViVi_NMT

3.1 Train

To train the model, run "translate.py" directly with default settings.

```
python translate.py
```

Model parameters and training settings can be set by command-line arguments, as follows:

```
--learning_rate: The initial learning rate of optimizer, default is 0.0005.
--learning_rate_decay_factor: Learning rate decays by this value, default is 0.99
--max_gradient_norm: Clip gradients to this norm, default is 1.0.
--batch_size: Batch size to use during training, default is 80.
```

--hidden_units: Size of hidden units for each layer, default is 1000.

--hidden_edim: Dimension of word embedding, default is 620.

--num_layers: Number of layers of RNN, default is 1.

--keep_prob: The keep probability used for dropout, default is 0.8.

--src_vocab_size: Vocabulary size of source language, default is 30000.

--trg_vocab_size: Vocabulary size of target language, default is 30000.

--data_dir: Data directory, default is './data'.

--train_dir: Training directory, default is './train/.

--max_train_data_size: Limit on the size of training data (0: no limit), default is 0.

--steps_per_checkpoint: How many training steps to do per checkpoint, default is 1000.

Note that, we provide a sampled Chinese-English dataset in './data', with 10000 sentences in training set, 400 sentences in development set, and another 400 sentences in testing set. We sample the training data from LDA corpora, and sample development and testing from NIST2005 and NIST 2003 respectively.

3.2 Test

To test a trained model,for example, the 10000th checkpoint, run the command below.

```
python ./translate.py --model translate.ckpt-10000 --decode --beam_size 10 < data/test.src > test.trans
perl ./multi-bleu.perl data/test.trg < test.trans
```

Note that,if there are multiple target files such as test.trg0, test.trg1, test.trg2, and test.trg3, users need to set the value to the shared prefix test.trg.

Model parameters should be the same as settings when training, and other parameters for decoding are as follows.

--decode: True or False. Set to True for interactive decoding, default is False.

--model: The checkpoint model to load.

--beam_size: The size of beam search, default is 1, which represents a greedy search.

Reference

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In Proceedings of ICLR.