Deep speech build

These instructions are for those of us who wish to have a version of Mozilla's DeepSpeech where they can make changes to the `native_client` decoder.

Getting the build environment set up to enable this is a non-trivial process, and we hope that these instructions will guide the reader through the setup in as painless a way as possible.

Many modern practices in software engineering are used, some of which may appear counterintuitive. However, it is possible to use Bazel to build the software, so do not give up when you face adversity!

Set up a virtualenv for everything

Let's say you're in your home directory, make a new directory for all of the source code stuff:

```
$ mkdir Mozilla
$ cd Mozilla
$ virtualenv -p python3.7 tmp/deepspeech-venv/$ source
/home/fran/source/Mozilla/tmp/deepspeech-venv/bin/activate
```

Get the source code

Now check out your fork of DeepSpeech (We presume you have forked it and are intending to submit pull requests)

```
$ git clone git@github.com:ftyers/DeepSpeech.git
```

Replace `ftyers` here with your GitHub username.

Next check out Mozilla's `tensorflow` fork:

```
$ git clone https://github.com/mozilla/tensorflow.git
$ cd tensorflow
$ git checkout origin/r1.14
$ cd ..
```

Next check out `kenlm`:

```
$ git clone git@github.com:kpu/kenlm.git
```

Now run `ls`, you should have the following directories:

```
$ ls
DeepSpeech  kenlm  tensorflow  tmp
```

Install Bazel

In order build the `native_client` stuff, you need Google's `Bazel` build system, each version of tensorflow makes its own requirements on the version of Bazel you need. For tensorflow 1.14, you need Bazel 0.24.1, you can install it using the following commands:

```
$ wget https://github.com/bazelbuild/bazel/releases/download/0.24.1/bazel-0.24.1-installer-linux-x86_64.sh
$ bash bazel-0.24.1-installer-linux-x86_64.sh --prefix=/home/fran/local/bin
```

Note that this shell script installs a binary of `Bazel` that is embedded in a zip file inside the shell script, the --prefix give the location to install it. It should be in your `$PATH`.

Check that Bazel is working by running:

```
$ bazel version
```

Set up necessary symlinks

and build options

```
$ cd tensorflow
$ ln -s ../DeepSpeech/native_client/ .
$ ./configure
```

This is an interactive `configure` script. Make sure you are using `python3.7` if you aren't you can set the environment variable `PYTHON_BIN_PATH`.

You definitely want to change the "desired Python library path to use", make sure you change it to something like:

```
/usr/lib/python3.7/dist-packages/
```

If you are building on a laptop without a GPU you can basically say no to all the other questions. You should get something like:

```
$ ./configure WARNING: --batch mode is deprecated. Please instead explicitly shut down your
Bazel server using the command "bazel shutdown".
You have bazel 0.24.1 installed.
Found possible Python library paths:
  /home/fran/local/lib/python3.7/site-packages/
  /usr/lib/python3/dist-packages
  /usr/local/lib/python3.7/dist-packagesPlease input the desired Python library path to use.
Default is [/home/fran/local/lib/python3.7/site-packages/]
Do you wish to build TensorFlow with XLA JIT support? [Y/n]: nNo XLA JIT support will be
enabled for TensorFlow.
Do you wish to build TensorFlow with OpenCL SYCL support? [y/N]: nNo OpenCL SYCL support will
be enabled for TensorFlow.
Do you wish to build TensorFlow with ROCm support? [y/N]: nNo ROCm support will be enabled
for TensorFlow.
Do you wish to build TensorFlow with CUDA support? [y/N]: nNo CUDA support will be enabled
for TensorFlow.
Do you wish to download a fresh release of clang? (Experimental) [y/N]: nClang will not be
downloaded.
Do you wish to build TensorFlow with MPI support? [y/N]: nNo MPI support will be enabled for
TensorFlow.
```

```
Please specify optimization flags to use during compilation when bazel option "--config=opt"
is specified [Default is -march=native -Wno-sign-compare]:
Would you like to interactively configure ./WORKSPACE for Android builds? [y/N]: nNot
configuring the WORKSPACE for Android builds.
```

Build `native_client` in the tensorflow directory

```
$ bazel build --workspace_status_command="bash native_client/bazel_workspace_status_cmd.sh" --
config=monolithic -c opt --copt=-O3 --copt="-D_GLIBCXX_USE_CXX11_ABI=0" --copt=-
fvisibility=hidden //native_client:libdeepspeech.so //native_client:generate_trie
```

If you run out of memory during the build process, specify the number of jobs by adding `--jobs` at the end of the bazel build command, e.g.

```
$ bazel build --workspace_status_command="bash native_client/bazel_workspace_status_cmd.sh" --
config=monolithic -c opt --copt=-O3 --copt="-D_GLIBCXX_USE_CXX11_ABI=0" --copt=-
fvisibility=hidden //native_client:libdeepspeech.so //native_client:generate_trie --jobs 6
```

Compile the `deepspeech` binary and language bindings

First compile the `deepspeech` binary

```
$ cd ../DeepSpeech/native_client/
$ TFDIR=../../tensorflow make deepspeech
```

Then compile and install the Python bindings

```
$ cd python
$ make bindings
$ pip install dist/deepspeech*
```

And the `ctcdecode` package:

```
$ cd ../ctcdecode/
$ make bindings
$ pip install dist/*.whl
```

Compile kenlm

```
$ cd ../../../kenlm/
$ mkdir build
$ cd build
$ cmake ../
$ make
```

Make a change in the native_client and rebuild it

Make a change in the `native_client` code:

```
$ cd ../../tensorflow/
```

Now edit `native_client/ctcdecode/scorer.cpp`, you can add something like:

```
  std::cerr << "Scorer::setup()" << lm_path << " ||| " trie_path << std::endl;
```

And then try and rebuild:

First you need to run `make clean` in `native_client/ctcdecode`

```
cd native_client/ctcdecode
make clean
cd ../../
```

This is because the build leaves behind some `.h` files that are picked up by a build rule somewhere.

And then you can rebuild:

```
bazel build --workspace_status_command="bash native_client/bazel_workspace_status_cmd.sh" --
config=monolithic -c opt --copt=-O3 --copt="-D_GLIBCXX_USE_CXX11_ABI=0" --copt=-
fvisibility=hidden //native_client:libdeepspeech.so //native_client:generate_trie
```

Now set up some data and try it out

First of all, set up a language model with KenLM:

You'll need some English text, you can get it from OPUS

```
$ cd ../DeepSpeech/data/lm$ wget http://opus.nlpl.eu/download.php?f=SETIMES/v2/moses/en-
tr.txt.zip -O en-tr.txt.zip
$ unzip en-tr.txt.zip
$ ../../../kenlm/build/bin/lmplz -o 5 < SETIMES.en-tr.en > lm.arpa$
../../../kenlm/build/bin/build_binary lm.arpa lm.binary$ cat SETIMES.en-tr.en | sed
's/./&\n/g' | sort -u > alphabet.txt$ ../../../tensorflow/bazel-
bin/native_client/generate_trie alphabet.txt lm.binary trie
```

Now you can run the demo:

```
$ cd ../../
$ pip3 install -r requirements.txt
$ ./bin/run-ldc93s1.sh
```

It should finish with:

```
I0717 14:56:33.608065 139972947781440 saver.py:1280] Restoring parameters from
/home/fran/.local/share/deepspeech/ldc93s1/train-800I Restored variables from most recent
checkpoint at /home/fran/.local/share/deepspeech/ldc93s1/train-800, step 800Testing model on
data/ldc93s1/ldc93s1.csv
I Test epoch...Test on data/ldc93s1/ldc93s1.csv - WER: 1.000000, CER: 0.980769, loss: 0.005800
--------------------------------------------------------------------------------WER:
1.000000, CER: 0.980769, loss: 0.005800 - src: "she had your dark suit in greasy wash water
all year" - res: "i"
--------------------------------------------------------------------------------
```

The change-build-test loop

If you make a change to `native_client`, you need to go back to the tensorflow directory and run:

```
$ bazel build --workspace_status_command="bash native_client/bazel_workspace_status_cmd.sh" --
config=monolithic -c opt --copt=-O3 --copt="-D_GLIBCXX_USE_CXX11_ABI=0" --copt=-
fvisibility=hidden //native_client:libdeepspeech.so //native_client:generate_trie --
verbose_failures
```

Then after that you need to go back to the DeepSpeech/native_client directory

```
$ cp ../../tensorflow/bazel-bin/native_client/libdeepspeech.so /home/fran/local/lib/
```

for me this is inside my virtualenv e.g.
`~/deep-speech-env/lib/python3.6/site-packages/deepspeech/lib/libdeepspeech.so`

```
 2211  cd ../DeepSpeech/native_client/
 2212  cd ctcdecode/
 2213  make bindings
 2216  pip install -U dist/*.whl
 2220  cd native_client/
 2221  cd python/
 2223  make bindings
 2226  pip3 install -U dist/*
```

when doing this, something else is missing. e.g. something is hanging around. The rebuilt version

is still printing a ton of debug statements even though I have removed these.

Revision #3

Created Wed, Jul 17, 2019 6:09 PM by kenneth

Updated Wed, Jul 17, 2019 9:15 PM by kenneth