1. Install Kaldi and OpenFST

Follow the instructions at http://kaldi.sourceforge.net/install.html to download the Kaldi and OpenFST source and then http://kaldi.sourceforge.net/build_setup.html to compile the source into libraries and binaries you can use. A Linux environment is recommended; however, the Kaldi site also has installation instructions for Windows. Note that there are Linux workstations you can access listed here (remote access) http://cims.nyu.edu/webapps/content/systems/resources/labs and here (at WWH) http://cims.nyu.edu/webapps/content/systems/resources/computeservers.

Please remember to never run large compute jobs on the login server access.cims.nyu.edu. Rather, use one of the compute servers such as mauler.cims or brawler.cims. You will need about 8G of disk space for this assignment, which you can find in /data/$USER. Note: on the CIMS compute servers, you will likely need to use ". /configure --use-cuda=no" in order to get Kaldi to work. Finally, note that compiling Kaldi and all the tools from source can take hours, so please leave ample time to complete this assignment.

```
cd /data/$USER
svn co svn://svn.code.sf.net/p/kaldi/code/trunk kaldi-trunk # 10-20 minutes
cd kaldi-trunk
cd tools
make       # Takes hours
cd ../src
./configure --use-cuda=no
make depend
make
```

2. Complete Kaldi Tutorial

Follow the steps at http://kaldi.sourceforge.net/tutorial.html to learn how to use Kaldi. Since we don’t have the full Resource Management corpus available to us, we will actually run the examples located in the eggs/rm/s4 directory instead of the ones referred to “Running the example scripts” part of the tutorial. Note that in contrast to the tutorial, which is based on eggs/rm/s5, we will be using pre-computed mel-frequency cepstral coefficient (MFCC) features in
this exercise. However, you should examine the parts of the scripts in egs/rm/s5 related to feature extraction as directed in the exercise. Note that some of the paths inside egs/rm/s4 will be different than those inside egs/rm/s5, which are referenced in the tutorial. Once the running of the example scripts is complete, answer the following questions about your resulting system. Please include all shell commands you ran to arrive at the answer.

```
cd /data/$USER/kaldi-trunk/egs/rm/s4
./getData.sh
./run.sh
```

- **What is the phone (phoneme) set of your system?**

  ```
  $ cat data/lang/phones.txt | awk '{print $1}' | grep -v "eps"
  aa
  ae
  ah
  ao
  aw
  ax
  axr
  ay
  b
  ch
  d
  dh
  dx
  eh
  er
  ey
  f
  g
  hh
  ih
  ix
  iy
  jh
  k
  l
  m
  n
  ng
  ow
  oy
  p
  r
  s
  sh
  t
  th
  ts
  uh
  uw
  v
  w
  y
  ```
Which kind of notation is being used to encode the phone set? Arpabet

How many words are in your lexicon?
```
wc -l data/local/lexicon.txt
1002
```

How many states are in your L.fst transducer?
```
$ tools/openfst/bin/fstinfo data/lang/L.fst | grep states
# of states 4658
# of final states 1
# of accessible states 4658
# of coaccessible states 4658
# of connected states 4658
```

How many utterances are used to train the monophone system? How about the triphone one?
```
$ grep -l "train monophone" run.sh # (to find monophone training dir)

# train monophone system.
steps/train_mono.sh data/train.lk data/lang exp/mono || exit 1;

$ wc -l data/train.lk/feats.scp
1000 data/train.lk/feats.scp
$ grep -l "train tri1" run.sh

# train tri1 [first triphone pass]
steps/train_deltas.sh data/train/data/lang/exp/mono ali exp/tri1 || exit 1;
$ wc -l data/train/feats.scp
1600 data/train/feats.scp
```

List some of the MFCC features for a training or test utterance.
```
$ tools/featbin/copy-feats ark:'pwd'/data/feat/mono.raw_mfcc_test.ark ark,t:- | head -2
```

Show some alignments
```
$ tools/bin/show-alignments data/lang/phones.txt exp/mono/0.mdl
```

List some of the MFCC features for a training or test utterance.
```
$ tools/featbin/copy-feats ark:'pwd'/data/feat/mono.raw_mfcc_test.ark ark,t:- | head -2
```

Show some alignments
```
$ tools/bin/show-alignments data/lang/phones.txt exp/mono/0.mdl
```
How many non-silence states are in your monophone HMMs? How about in your final triphone HMMs?

$ ./../src/bin/show-transitions data/lang/phones.txt exp/mono/30.mdl | grep state | grep -v sil | tail -1

Transition-state 132: phone = zh hmm-state = 2 pdf = 131

$ ./../src/bin/show-transitions data/lang/phones.txt exp/tri2a/25.mdl | grep state | grep -v sil | tail -1

Transition-state 967: phone = zh hmm-state = 2 pdf = 365

3. Install and test OpenGRM NGram and Thrax libraries

Download and install the OpenGRM library and the Thrax grammar compiler from opengrm.org. Complete the tutorials listed in the “Quick Tour” section of both the OpenGRM and the Thrax pages. Provide a listing of all shell commands you used and the output.

Note: to point OpenGRM at the OpenFST binaries you compiled in problem 1, you will need to issue the following flags before running ./configure (assuming you are using bash):

```
export LDFLAGS="-L/data/\${USER}/kaldi-trunk/tools/openfst/lib
-L/data/\${USER}/kaldi-trunk/tools/openfst/lib/fst"
export CPPFLAGS="-I/data/\${USER}/kaldi-trunk/tools/openfst/src/include"
```

Unfortunately, the version of OpenFST downloaded with Kaldi is not compatible with Thrax, so you’ll need to separately download OpenFST 1.3.3 or later from openfst.org and point the Thrax Autoconf at it. You should compile OpenFST with far (FST Archive) and pdt (push-down transducers) enabled. Assuming you’ve unTarred the .tar.gz in /data/\${USER}/openfst-1.3.3, you will need to do:
You may want to include the binary directories of the libraries you have installed in your path by modifying your .bashrc or similar file.

cd /data/$USER/openfst-1.3.3
./configure --enable-far --enable-pdt
make

then

export OPENFST=/data/$USER/openfst-1.3.3
export LDFLAGS="-L$OPENFST/lib -L$OPENFST/lib/fst"
export CPPFLAGS="-I$OPENFST/include"

wget http://www.openfst.org/twiki/pub/GRM/NgramDownload/opengrm-ngram-1.0.3.tar.gz
./configure
make
wget http://openfst.cs.nyu.edu/twiki/pub/GRM/NGramQuickTour/earnest.txt
export
PATH=$PATH:/data/$USER/opengrm-ngram-1.0.3/src/bin:/data/$USER/kaldi-trunk/tools/openfst/bin
ngramsymbols <earnest.txt >earnest.syms
farcompilestrings -symbols=earnest.syms -keep_symbols=1 earnest.txt >earnest.far
ngramcount -order=5 earnest.far >earnest.cnts
ngrammake earnest.cnts >earnest.mod
ngramrandgen earnest.mod | farprintstrings

mkdir /data/$USER/openfst
cd /data/$USER/openfst
wget http://www.openfst.org/twiki/pub/FST/FstDownload/openfst-1.3.3.tar.gz
tar -xzvf openfst-1.3.3.tar.gz
cd openfst-1.3.3
./configure --enable-far --enable-pdt
make

mkdir /data/$USER/thrax
cd /data/$USER/thrax
wget http://www.openfst.org/twiki/pub/GRM/ThraxDownload/thrax-1.1.0.tar.gz
tar -xzvf thrax-1.1.0.tar.gz
cd thrax-1.1.0
export OPENFST=/data/$USER/openfst/openfst-1.3.3
export LDFLAGS="-L$OPENFST/lib -L$OPENFST/lib/fst"
export CPPFLAGS="-I$OPENFST/include"
./configure
make

mkdir /data/$USER/thrax/thrax-1.1.0/src/grammars
../bin/thraxmakedep example.grm
../bin/thraxrewrite-tester --far=example.far --rules=TOKENIZER
Appendix: output of run.sh

$ ./run.sh
RM_data_prep succeeded.
fstisstochastic data/lang_test/G.fst
3.70666e-07 -3.60422e-07
fsttablecompose data/lang_test/L_disambig.fst data/lang_test/G.fst
fsttablecompose data/lang/L.fst data/lang_test/G.fst
fstisstochastic
3.70666e-07 -3.60422e-07
fstisstochastic
fsttablecompose data/lang_test/L_disambig.fst data/lang_test/G.fst
3.70666e-07 -3.60422e-07

First few lines of lexicon FST:
0 1 <eps> <eps> 0.693147182
0 1 sil <eps> 0.693147182
1 2 ah A 0.693147182
1 1 ah A 0.693147182
1 3 ey A42128
1 15 ey AAW
1 23 ae ABERDEEN
1 28 ax ABOARD
1 32 ax ABOVE
1 35 ae ADD
RM_format_data succeeded.
Succeeded creating MFCC features for train
Succeeded creating MFCC features for test
Created a 1000-utterance subset of data/train and put it in data/train.1k.
Computing cepstral mean and variance statistics
Initializing monophone system.
Compiling training graphs
Pass 0
Pass 1
Aligning data
Pass 2
Aligning data
Pass 3
Aligning data
Pass 4
Aligning data
Pass 5
Aligning data
Pass 6
Aligning data
Pass 7
Aligning data
Pass 8
Aligning data
Pass 9
Aligning data
Pass 10
Aligning data
Pass 11
Pass 12
Aligning data
Pass 13
Pass 14
Pass 15
Aligning data
Pass 16
Pass 17
Pass 18
Pass 19
Pass 20
Aligning data
Pass 21
Pass 22
Pass 23
Pass 24
Pass 25
Aligning data
Pass 26
Pass 27
Pass 28
Pass 29
fsttablecompose data/lang_test/L_disambig.fst data/lang_test/G.fst
fstdeterminizestar --use-log=true
fstminimizeencoded
fstisstochastic data/lang_test/tmp/LG.fst
0.000480175 -0.000445507
fstcomposecontext --context-size=1 --central-position=0
--read-disambig-syms=data/lang_test/tmp/disambig Phones.list
--write-disambig-syms=data/lang_test/tmp/disambig ilabels_1_0.list
data/lang_test/tmp/ilabels_1_0
fstisstochastic data/lang_test/tmp/CLG_1_0.fst
0.000480175 -0.000445507
make-h-transducer --disambig-syms-out=exp/mono/graph/disambig tid.list
--transition-scale=1.0 data/lang_test/tmp/ilabels_1_0 exp/mono/tree exp/mono/final.mdl
cfstablecompose exp/mono/graph/Ha.fst data/lang_test/tmp/CLG_1_0.fst
fstdeterminizestar --use-log=true
fstminimizeencoded
fstrmepslocal
fststrmsymbols exp/mono/graph/disambig tid.list
fstisstochastic exp/mono/graph/HCLG a.fst
0.000480175 -0.000445507
add-self-loops --self-loop-scale=0.1 --reorder=true exp/mono/final.mdl
exp/mono/decode/wer_10: %WER 13.41 [ 762 / 5681, 15 ins, 324 del, 423 sub ]
exp/mono/decode/wer_4: %WER 11.78 [ 669 / 5681, 71 ins, 151 del, 447 sub ]
exp/mono/decode/wer_5: %WER 11.81 [ 671 / 5681, 50 ins, 178 del, 443 sub ]
exp/mono/decode/wer_6: %WER 11.92 [ 677 / 5681, 35 ins, 203 del, 439 sub ]
exp/mono/decode/wer_7: %WER 12.44 [ 707 / 5681, 29 ins, 238 del, 440 sub ]
exp/mono/decode/wer_8: %WER 12.74 [ 724 / 5681, 21 ins, 267 del, 436 sub ]
exp/mono/decode/wer_9: %WER 12.90 [ 733 / 5681, 15 ins, 301 del, 417 sub ]
Computing cepstral mean and variance statistics
Aligning all training data
Done.
Accumulating tree stats
Computing questions for tree clustering
Building tree
Compiling training graphs
Pass 1
Pass 2
Pass 3
Pass 4
Pass 5
Aligning data
Pass 6
Pass 7
Pass 8
Pass 9
Pass 10
Aligning data
Pass 11
Pass 12
Pass 13
Pass 14
Pass 15
Aligning data
Pass 16
Pass 17
Pass 18
Pass 19
Pass 20
Aligning data
Pass 21
Pass 22
Pass 23
Pass 24
Done
fstcomposecontext --context-size=3 --central-position=1
--read-disambig-syms=data/lang_test/tmp/disambigPhones.list
--write-disambig-syms=data/lang_test/tmp/disambig_labels_3_1.list
data/lang_test/tmp/labels_3_1
fstisstochastic data/lang_test/tmp/CLG_3_1.fst
0.000480175 -0.000445507
make-h-transducer --disambig-syms-out=exp/tri1/graph/disambig_tid.list
--transition-scale=1.0 data/lang_test/tmp/labels_3_1 exp/tri1/tree exp/tri1/final.mdl
fstrpmeslocal
fsttablecompose exp/tri1/tree/Ha.fst data/lang_test/tmp/CLG_3_1.fst
fstminimizeencoded
fstdeterminizestar --use-log=true
fstrmsymbols exp/tri1/graph/disambig_tid.list
fstisstochastic exp/tri1/graph/HCLG.fst
0.000705957 -0.000672728
add-self-loops --self-loop-scale=0.1 --reorder=true exp/tri1/final.mdl
exp/tri1/decode/wer_10:WER 6.79 [ 386 / 5681, 47 ins, 96 del, 243 sub ]
exp/tri1/decode/wer_4:WER 6.25 [ 355 / 5681, 77 ins, 45 del, 233 sub ]
exp/tri1/decode/wer_5:WER 6.11 [ 347 / 5681, 69 ins, 50 del, 228 sub ]
exp/tri1/decode/wer_6:WER 6.11 [ 347 / 5681, 65 ins, 55 del, 227 sub ]
exp/tri1/decode/wer_7:WER 6.18 [ 351 / 5681, 59 ins, 61 del, 231 sub ]
exp/tri1/decode/wer_8:WER 6.27 [ 356 / 5681, 56 ins, 69 del, 231 sub ]
exp/tri1/decode/wer_9:WER 6.55 [ 372 / 5681, 51 ins, 83 del, 238 sub ]
Computing cepstral mean and variance statistics
Aligning all training data
Done.
Accumulating tree stats
Computing questions for tree clustering
Building tree
Compiling training graphs
Pass 1
Pass 2
Pass 3
Pass 4
Pass 5
Aligning data
Pass 6
Pass 7
Pass 8
Pass 9
Pass 10
Aligning data
Pass 11
Pass 12
Pass 13
Pass 14
Pass 15
Aligning data
Pass 16
Pass 17
Pass 18
Pass 19
Pass 20
Aligning data
Pass 21
Pass 22
Pass 23
Pass 24
Done
make-h-transducer --disambig-syms-out=exp/tri2a/graph/disambig_tid.list
--transition-scale=1.0 data/lang_test/tmp/ilabels_3_1 exp/tri2a/tree exp/tri2a/final.mdl
fstremeplocal
fstrmsymbols exp/tri2a/graph/disambig_tid.list
fstminimizeencoded
fstdeterminizestar --use-log=true
fsttablecompose exp/tri2a/graph/Ha.fst data/lang_test/tmp/CLG_3_1.fst
fstisstochastic exp/tri2a/graph/HCLGa.fst
0.000622809 -0.000656247
add-self-loops --self-loop-scale=0.1 --reorder=true exp/tri2a/final.mdl
exp/tri2a/decode/wer_10:%WER 6.97 [ 396 / 5681, 56 ins, 90 del, 250 sub ]
exp/tri2a/decode/wer_4:%WER 6.92 [ 393 / 5681, 102 ins, 50 del, 241 sub ]
exp/tri2a/decode/wer_5:%WER 6.86 [ 390 / 5681, 89 ins, 52 del, 249 sub ]
exp/tri2a/decode/wer_6:%WER 6.86 [ 390 / 5681, 84 ins, 54 del, 252 sub ]
exp/tri2a/decode/wer_7:%WER 7.02 [ 399 / 5681, 76 ins, 65 del, 258 sub ]
exp/tri2a/decode/wer_8:%WER 6.94 [ 394 / 5681, 68 ins, 74 del, 252 sub ]
exp/tri2a/decode/wer_9:%WER 6.97 [ 396 / 5681, 63 ins, 83 del, 250 sub ]