Speech to Speech Translation

09.05.2016 - 08.07.2016

Tejesh Raut

Email: tejesh22.raut@gmail.com

B Tech Second year
Computer Science and Engineering
IIT Bombay, Powai - 400076

Overview

This is an offline application by which user can translate voice from English to Hindi. This project is done using basically three parts. Speech from English is converted to text using a java project made using Sphinx4 java libraries. Then the text from English is converted to text in Hindi using Moses. And finally the text from Hindi is read out i.e., converted from Hindi text to speech using Festival. This is just a broad overview of working of the application. Many other softwares are used which will be referred later.

Goals

1. Speech to speech translation between any two languages without using any online API.
2. A continuous running application with good interface so that people with different languages can talk with each other freely.

User's Manual

Before starting this application you need to set up the environment in your PC. Details of setting up the environment is given on the next page.

Once you have setup everything, you need to go to this directory using terminal and run the run.sh file.

```
./run.sh
```

Start speaking after 0.5 seconds and when recording is over press ctrl + C to stop. The recording of the first 0.5 seconds will be considered and will be used to create the noise profile which will be further used for getting a clean audio.

After few seconds you will hear back the speech in Hindi.

You can see the text it has interpreted in English by opening the file Text_English/t3.txt and its translation done to Hindi by opening the file Text_Hindi/t3.txt.

Setting up the environment

For running this application you will first need to install some softwares and configure your PC to run it. Follow the given steps and proceed only if there are no errors. While following this tutorial you will also understand each software used in detail and the way they are used which might be useful for other projects.

Recording noise free audio or remove noise from a recorded audio file

When you record an audio it is usually very unclear and full of background in Ubuntu. Following steps will help you remove the noise from audio to a great extent:

- First of all you will need to install SOX package. To install "sox" open the terminal and run the following command:
  sudo apt-get install sox
- Record the audio in a file original.wav :
  rec -c 1 -r 16000 -b 16 original.wav
  - This command records audio until ctrl+C is pressed.
  - -c defines that sound is recorded in a single channel, -r defines 16000 sample-rate and -b defines 16-bit sample encoding.
- Amplify the audio if it is not audible.
  sox -v 256.0 original.wav amplified.wav
  - This command amplifies 256 times (amplification works on logarithmic scale)
  - Amplified audio is saved in other file with name amplified.wav
- Trim out a noise sample from it :
  sox amplified.wav noise.wav trim 0 0.5
  - This command will trim out noise of length 0.5 seconds starting from 0 seconds and save it to the file noise.wav
- Create the noise profile :
  sox noise.wav -n noiseprof noise.prof
- Create cleaned audio after removing noise :
  sox amplified.wav cleaned.wav noisered noise.prof 0.2
  - You can set the noise removing parameter to any value. Usually 0.2 to 0.3 gives better results.

You can play the cleaned audio from terminal itself using the command :
play cleaned.wav

Installing moses and its components for text translation between two languages

Moses is a statistical machine translation system that allows you to automatically train translation models for any language pair. All you need is a collection of translated texts (parallel corpus). Once you have a trained model, an efficient search algorithm quickly finds the highest probability translation among the exponential number of choices.

Before installing moses install the following packages :

g++, git, subversion, automake, libtool, zlib1g-dev, libboost-all-dev, libbz2-dev, liblzma-dev, python-dev, libtcmalloc-minimal4

```
sudo apt-get install g++ git subversion automake libtool zlib1g-dev
libboost-all-dev libbz2-dev liblzma-dev python-dev
libtcmalloc-minimal4
```

Installing Boost :

Make a directory where all work related to Machine Translation will be present:

```
mkdir MachineTranslation
```

Download Boost from Sourceforge and extract it to this folder using file manager.

If you download boost_1_61_0 you must be able to see the folder boost_1_61_0 in the folder MachineTranslation. Then enter that folder using terminal.

```
cd boost_1_61_0
```

Then run the following commands:

```
./bootstrap.sh
```

```
./b2 -j5 --prefix=$PWD -- libdir=$PWD/lib64 --layout=tagged
link=static threading =multi,single install || echo FAILURE
```

In the above command **-j5** indicates my PC is 5 Core machine (i.e. i5 processor). Use **-j2** for dual core or **-j3** for i3.

Installing Moses :

Download Moses decoder, Giza++ from github and extract to the directory MachineTranslation

In terminal enter the directory mosesdecoder-master

```
cd ../mosesdecoder-master/
```

Install moses:

```
./bjam -j5
```

If you installed moses successfully, you will be able to see the options available with bjam. Run:

./bjam --help

./bjam --with-boost=/home1/MachineTranslation/boost_1_61_0/ -j5

Enter the path of the directory MachineTranslation according to your PC in the above command

Then enter the directory giza-pp-master:

cd ../giza-pp-master/

Install giza-pp-master:

make

Go to the directory mosesdecoder-master:

cd ../mosesdecoder-master/

Make tools directory:

mkdir tools

Copy components to the tools directory:

cp ../giza-pp-master/GIZA++-v2/GIZA++
../giza-pp-master/GIZA++-v2/snt2cooc.out
../giza-pp-master/mkcls-v2/mkcls tools/

Installing IRSTLM :

Download the latest version of IRSTLM from Sourceforge and extract it to the MachineTranslation directory.

Now enter the directory trunk in irstlm in the terminal

cd ../irstlm-5.80.08/trunk/

Install IRSTLM:

./regenerate-makefiles.sh

./configure --prefix=/home1/MachineTranslation/irstlm-5.80.08/

make install

Enter the path of the directory MachineTranslation according to your PC in the above command.

Sample example running on moses :

Go to the directory mosesdecoder-master

cd ../../mosesdecoder-master/

Download samples:

wget http://www.statmt.org/moses/download/sample-models.tgz

tar xzf sample-models.tgz

rm sample-models.tgz

cd sample-models/

Run the decoder :

/home1/MachineTranslation/mosesdecoder-master/bin/moses -f phrase-model/moses.ini < phrase-model/in > out

If everything worked out right, this should translate the sentence "das ist ein kleine haus" (in the file in) as "this is a small house" (in the file out).

Train Moses for English-Hindi translation

In this tutorial we will be training moses to translate text from English to Hindi.

You need corpus to train moses for translation from English to Hindi.

Keep the files corpora.en, corpora.hi and the folder indic_nlp_library-master ready.

Corpus Input and making directory :

Make a new directory "corpus" in the MachineTranslation directory. Make a directory "training" inside the directory corpus and put the two files corpora.en and corpora.hi in it.

Place the folder indic_nlp_library-master into the MachineTranslation directory.

Pre-process corpora :

Hindi Normalisation :

```
cd corpus

python ../indic_nlp_library-master/src/indicnlp/normalize/indic_normalize.py training/corpora.hi corpora.norm.hi [True]
```

Hindi Tokenization :

```
python ../indic_nlp_library-master/src/indicnlp/tokenize/indic_tokenize.py corpora.norm.hi corpora.tok.hi hi
```

English Tokenization :

```
../mosesdecoder-master/scripts/tokenizer/tokenizer.perl -l en < training/corpora.en > corpora.tok.en
```

Create Truecase model for English :

```
../mosesdecoder-master/scripts/recaser/train-truecaser.perl --model truecase-model.en --corpus corpora.tok.en
```

Create Truecase model for Hindi :

```
../mosesdecoder-master/scripts/recaser/train-truecaser.perl --model truecase-model.hi --corpus corpora.tok.hi
```

Truecasing English :

```
../mosesdecoder-master/scripts/recaser/truecase.perl --model truecase-model.en < corpora.tok.en > corpora.true.en
```

Truecasing Hindi :

../mosesdecoder-master/scripts/recaser/truecase.perl --model truecase-model.hi < corpora.tok.hi > corpora.true.hi

Cleaning of English and Hindi :

../mosesdecoder-master/scripts/training/clean-corpus-n.perl corpora.true en hi corpora.clean 1 80

LM Building :

cd ../

mkdir lm

cd lm/

../irstlm-5.80.08/bin/add-start-end.sh < ../corpus/corpora.clean.hi > corpora.sb.hi

export IRSTLM=/home1/MachineTranslation/irstlm-5.80.08/

../irstlm-5.80.08/bin/build-lm.sh -i corpora.sb.hi -t ./tmp -p -s improved-kneser-ney -o corpora.lm.hi

../irstlm-5.80.08/bin/compile-lm corpora.lm.hi.gz --text=yes corpora.arpa.hi

../mosesdecoder-master/bin/build_binary -i corpora.arpa.hi corpora.blm.hi

Training :

cd ../

mkdir working

cd working

nohup nice ../mosesdecoder-master/scripts/training/train-model.perl -root-dir train -corpus ../corpus/corpora.clean -f en -e hi -alignment grow-diag-final-and -reordering msd-bidirectional-fe -lm 0:3:/home1/MachineTranslation/lm/corpora.blm.hi:8 -external-bin-dir ../mosesdecoder-master/tools >& training.out &

Enter the paths according to your PC. The last command takes a very long time. The time taken depends on the size of the corpora used. The corpora I have attached with this blog are of around 7 lakh lines and it took 7-8 hours for me for completing the training. You can see the operations going on by opening the file training.out in the directory working as whole output from this command is transferred to this file.

For checking whether the process is completed or not, enter the command ps.

When the operation is complete it will display “[1]+ Done” output.

Binarize Phrase Table and Lexical Reordering Table :

After the training is completed, the translation works but it is tooo slow. For large corpus as in this example it won't work at all. For working at good speeds i.e. translation within a second you need to binarize some files

Binarize Phrase Table :

```
/home1/MachineTranslation/mosesdecoder-master/bin/CreateOnDiskPt 1 1 4 100 2
/home1/MachineTranslation/working/train/model/phrase-table.gz
/home1/MachineTranslation/working/phrase-table.1.folder
```

Enter the paths as per your PC. After the above command a folder with name phrase-table.1.folder will be created in the working directory which contains the data required for binarized phrase table.

After binarizing phrase table you will need to make some changes in the moses.ini file present in working/train/model directory.

Open the moses.ini file in a editor and search for the line:

```
PhraseDictionaryMemory name=TranslationModel0 num-features=4 path=/home1/MachineTranslation/working/train/model/phrase-table.gz input-factor=0 output-factor=0
```

Comment the line by inserting '#' mark before it. And then enter the following line after it:

```
PhraseDictionaryOnDisk name=TranslationModel0 num-features=4 path=/home1/MachineTranslation/working/phrase-table.1.folder input-factor=0 output-factor=0
```

Note the colored changes that we have made. Path will depend on your PC.

Binarize Lexical Reordering Table :

```
/home1/MachineTranslation/mosesdecoder-master/bin/CreateOnDiskPt 1 1 6 100 2
/home1/MachineTranslation/working/train/model/reordering-table.wbe-msd-bidirectional-fe.gz
/home1/MachineTranslation/working/reordering-table.wbe-msd-bidirectional-fe.1.gz
```

Enter the paths as per your PC. After the above command a folder with name reordering-table.wbe-msd-bidirectional-fe.1.gz will be created in the working directory which contains the data required for binarized lexical reordering table.

After binarizing lexical reordering table you will need to make some changes in the moses.ini file present in working/train/model directory.

Open the moses.ini file in a editor and search for the line:

LexicalReordering name=LexicalReordering0 num-features=6 type=wbe-msd-bidirectional-fe-allff input-factor=0 output-factor=0 path=/home1/MachineTranslation/working/train/model/reordering-table.wbe-msd-bidirectional-fe.gz

Comment the line by inserting '#' mark before it. And then enter the following line after it:

LexicalReordering name=LexicalReordering0 num-features=6 type=wbe-msd-bidirectional-fe-allff input-factor=0 output-factor=0 path=/home1/MachineTranslation/working/reordering-table.wbe-msd-bidirectional-fe.1.gz

Note the colored changes that we have made. Path will depend on your PC.

Testing :

At this stage we are ready for testing. We can test moses now for translating from English to Hindi:

Run the following command from MachineTranslation directory to start moses:

mosesdecoder-master/bin/moses -f working/train/model/moses.ini

Type 'hello' and it should be translated to 'नमस्ते'

Now you can translate anything from English to Hindi using moses

Basic syntax for running moses is:

<Location of moses binary file> -f <Location of moses.ini>

In case you wish to translate after taking input from a file (You may need this for translating multiple lines) :

mosesdecoder-master/bin/moses -f working/train/model/moses.ini < inputEn.txt > outputHi.txt

Hindi Text to Speech conversion using festival

Festival is a general multi-lingual speech synthesis system. It can be used for conversion of text to speech. In this article we will discuss how to use festival for conversion of hindi text to speech.

Installing Festival tool for Hindi :

Enter the following command in terminal :

sudo apt-get install festival-hi

This will also install the main festival packages automatically along with the hindi support.

Converting English Text to speech using Festival :

To read out English text "Hello" to speech, open the festival interface enter the command (SayText "<text>") to read out the text :

festival

(SayText "Hello")

Or you can also hear the text directly from command-line as :

echo "Hello" | festival --tts

You will hear the "Hello" from your speakers.

You can read out English text from any file using the command:

festival --tts input.txt

Or go inside festival and then read out the file as :

festival

(tts "input.txt")

Converting Hindi Text to speech using Festival :

To read out Hindi text "नमस्ते" to speech, open the festival interface, change the voice and then enter the command to read out the text :

festival

Then inside festival :

(voice_hindi_NSK_diphone)

(SayText "नमस्ते")

To view what all voices you can use enter the command :

(voice.list)

And for selecting any voice use :

(voice_<voice name>)

But what if you need to read out Hindi from a file using some shell script. You will need command-line to read Hind. Festival itself provides the command text2wave to do this.

text2wave -o outputHindi.wav inputHindi.txt -eval "(voice_hindi_NSK_diphone)"

This will read the Hindi text from the file inputHindi.txt and store the output audio in the file outputHindi.wav.

Milestones

I. Understanding Moses and its installation

I spent the first few days in understanding what is moses and how does it work. Then I learnt how moses is trained using different corpus. Searched over internet to get the best corpus. Tried to integrate other Indian languages.

II. Working with Festival

Learnt how festival tool is used to read out a text. How to change the voices in festival to read different languages.

III. Speech Recognition

Searched for various tools for speech to text translation. Tried many different softwares and finally settled on Sphinx4 which works on java platform. Tried hard to adapt Sphinx to recognize Indian accent of English.

IV. Removing noise from audio recorded

Tried various methods to remove noise from the recorded audio and get a specific bit rate and encoding so as to give it as input to Sphinx to get converted into text.

V. Integrating Speech-Recognition, Translation and Text-To-Speech

Generated a bash script to run all the three projects simultaneously. Prevented the log from being printed while using Sphinx for speech recognition, so that only the recognized text would be forwarded for translation. Learnt command-line for running festival while changing the voice at the same time.

References :

https://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/

https://github.com/cmusphinx

http://cmusphinx.sourceforge.net/

https://sourceforge.net/projects/cmusphinx/files/

http://cmusphinx.sourceforge.net/wiki/download

https://github.com/cmusphinx/sphinxbase

https://github.com/cmusphinx/sphinxtrain

http://www.sphinx-doc.org/en/stable/install.html

http://cmusphinx.sourceforge.net/wiki/tutorialsphinx4

http://cmusphinx.sourceforge.net/wiki/sphinx4:webhome

http://sphinxsearch.com/docs/current/installing-debian.html

http://stackoverflow.com/questions/33453691/cmu-sphinx-for-indian-english

http://www.keithv.com/software/sphinx/

https://github.com/romanows/Sphinx-4-Acoustic-Model-Adaptation-Data

http://www.speech.cs.cmu.edu/tools/lmtool.html

http://www.cfilt.iitb.ac.in/~moses/shata_anuvaadak/register.html

https://en.wikipedia.org/wiki/ASR

https://sourceforge.net/projects/cmusphinx/

https://sourceforge.net/projects/espeak/?source=typ_redirect

http://www.cstr.ed.ac.uk/projects/festival/

https://oss.sonatype.org/content/repositories/snapshots/

https://github.com/joshua-decoder/indian-parallel-corpora

https://oss.sonatype.org/

https://sourceforge.net/p/cmusphinx/discussion/help/thread/051151d0/