Lab 5: First steps with Moses SMT

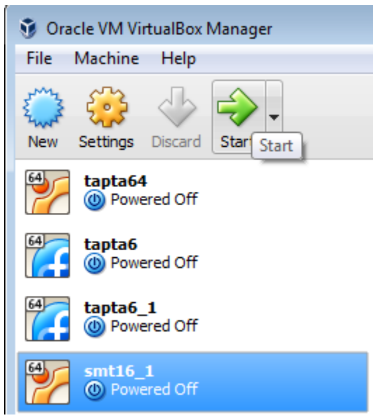
This documentation contains instructions to follow the lab n.5 GIAN course about MT

Status: 6/1/2017 BP V0.04

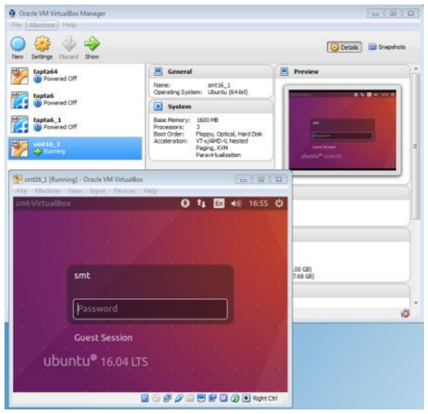
Contents

Preliminaries

1. Your host compute contains a "SMT" virtual machine provided
2. Open virtual box (on Ubuntu, click on "search", type virtual, open it)
3. The virtual machine will appear, select the machine, click "start"



➔


Oracle VM VirtualBox Manager
General
System
Preview
smt16_1 [Running] - Oracle VM VirtualBox
smt
Password
Guest Session
ubuntu 16.04 LTS

The username is “smt” and password is the same

The corpus

The “indic languages” corpus was downloaded from http://homepages.inf.ed.ac.uk/miles/data/indic/

It contains a set of sentences extracted from Wikipedia that were translated by 4 different persons using mechanical turk.

Each English sentence may have 4 different Hindi versions.

Each entry in the testset and devset has one Hindi version but 4 different English translations (hence the unique file test.hi and the 4 files test.en.0…test.4.en)

Launch a full training

Open a terminal and type:

```
bash Icon2016/labs/trainMoses.sh hi en
```

It will take ~10 minutes to run, you will observe various steps :

1) Tokenization
2) mkclass process (takes some time: it create word classes, useful for word alignments)
3) Then you will see “giza” process running, you can recognize model 1, model 2 etc.
4) Once the alignments are ready, you’ll observe the “phrase extraction”
5) The “scoring”
6) The final step will translate the 1000 sentences of the “testset” (test.hi)
7) finally you will get a BLEU score

Please read the documentation “UnixCommandsInANutshell” that was given to you yesterday

⇨ It will create a full MT model that you can now use with the following command:

```
/home/smt/mosesdecoder/bin/moses -f parallel-corpora/hi-en/trainMoses/model/moses.ini
```

(Note that if you type the command as shown above, it will wait for your input – a text in Hindi- , and will output the translation, you can finish your text by typing **CTRL-D**)

1. Use previous command to directly translate a given sentence:

e.g.

```
echo "मैं दुकान जाउंगा" | ~/mosesdecoder/bin/moses -f ~/parallel-corpora/hi-en/trainMoses/model/moses.ini
```

2. Use previous command to translate the first 5 sentences from the Hindi testset (test.hi) [your assignment]

3. Save the result in a file

   a. Check that the input file and the output file have exactly the same number of lines
      []

   b. Display them side-by side (original and machine translation)
      []

Look into the generated model

Now that you have a trained model, try to browse the various components of this model

Lexical translation table

The first step (after running word alignment - with the tool called « giza ») has created 2 files (into model dir)

lex.e2f and lex.f2e (lexical translation tables, first: Hindi→English second English→hindi)

Look at their content, try to look at some word translation (e.g. `grep university lex.e2f`)

Unix hint: There are some options in Unix command “sort” that are useful to know:

- sort -k *n* to sort according to the *n* column
- sort -n to sort a numerical value
- sort -k 4n to sort according to the 4th column using numerical sort

e.g.

to get probabilities of translation for “university”:

```
cat lex.e2f| grep ' university '
```

To get the 2 most probable translations of ‘university’, sort previous result according to 3rd column, numerically, and display only the last 2 lines

```
cat lex.e2f|  grep ' world '|sort -k 3n|tail -2
```

```
दुनिया world 0.0898204
विश्व world 0.4982036
```

Similarly, try to output the last 10 most probable Hindi translations for the word “university” (in lex.e2f)

[]

Try to get the most probable inverse translation for the word ‘university’ (in lex.f2e)

[]

Compare the 2 results: look at the probabilities, one file has probabilities that sums up to one, the other not, why?

[]

Extracted phrases

Look at the file called “extract.sorted.gz”, this is the result of the phrase extraction.

Unix hint: Note the file ends with “.gz”, this is a compressed file, to look at it use the command “zcat” that works like “cat” (usually: zcat extract.sort.gz|more)

Look at all phrases containing “book”, note that the word alignment (from Giza) is there too

Look at phrases where book is aligned with “किताब”

[]

Look at phrases where book is aligned with 'पुस्तक'

[]

Phrase table:

Look at the file called “phrasetable. gz”, this is the result of the phrase extraction.

Similar to previous exercise, find entries with ‘book→'किताब'’ and entries with book→ पुस्तक’

[]

Language model

Look at the file text.en.arpa

Look for the word ‘world’, you will see that there are probabilities for single word, bigrams adnd tri grams

Please note that here the probabilities are expressed in term of log

Read the blog: http://blog.wordtree.org/2014/01/11/arpa-lm/ for a short explanation of the meaning of these.

The exact format is

logarithm (base 10) of conditional probability *p* of that N-gram, followed by the words *w1*...*wN* making up the N-gram. Here, they are followed by the logarithm (base 10) of the backoff weight for the N-gram

When you see this line:

```
-3.1987915      world    -0.36612493
```

It means that the word ‘world’ has a probability of $10^{-3.1987915}$, which is 0.00063

Compare the probabilities of ‘world’ and ‘worldwide’:

[]

Training assignments

Train the other direction

Have a look at the training script, and try to train a model for the other direction (e.g. from English into Hindi)

Note that the script uses “variables” ($src for the source language $tgt for the target language), the only difference is on the test set files (you have 4 different testsets for English and only one for the indic language).

Adding more data to the model

Note that this assignment may be rather complicated, but this will help you building your own model in the future.

For some languages (not Hindi) you have also a bilingual dictionary (e.g. for Mayalayam: dict.ml dict.en)

Find the right commands to add this data to the training, save the previous model, and launch a new training on this (read the instructions in “Icon2002/labs/trainModel.sh”).

E.g. for Mayalam, with the “basic” training, I had a BLEU score of 11.57, after adding the dictionary, I had 17.35

Bigger Hindi-English model

Do a similar training on more data using another English-Hindi parallel corpus:

https://lindat.mff.cuni.cz/repository/xmlui/bitstream/handle/11858/00-097C-0000-0023-625F-0/hindencorp05.plaintext.gz

Solution for adding dictionary data (here in Malayalam):

```
# go in the right directory
cd ~/parallel-corpora/ml-en/
# save the previous trained model:
mv  trainMoses trainMoses.basic
# Create a new “training”  directory
mkdir trainMoses
cd trainMoses
# Copy the previous sets
cp ../trainMoses.basic/test* ../trainMoses.basic/trainset* .
# tokenize the “ml” part of the dictionary, append it to the end of the trainset
~/Icon2016/labs/tokenizer.perl -l ml < ../dict.ml >> trainset.ml
# same for English part
~/Icon2016/labs/tokenizer.perl -l en < ../dict.en >> trainset.en
# create the new language model (still with 3-grams,
# type ~/mosesdecoder/bin/lmplz without parameter to know the options)
# Here we add the  --discount_fallback option to avoid
/home/smt/mosesdecoder/bin/lmplz -o 3 --discount_fallback -S 80% -T /tmp < ./trainset.en
>text.en.arpa
# launch the Moses training
~/mosesdecoder/scripts/training/train-model.perl --parallel 3 --external-bin-dir ~/joshua/bin -root-
dir . --corpus ./trainset --f ml --e en -lm 0:3:`pwd`/text.en.arpa
# Launch the decoding
~/mosesdecoder/bin/moses -f model/moses.ini < test.ml > output.en
# output a BLEU score
/home/smt/mosesdecoder/scripts/generic/multi-bleu.perl ./test.en.[0-3] < output.en
```