**Gene translation system direction**

# 1 Code and examples

Directory: code\_and\_example/

## 1.1 System requirement

(1) You need install moses system. http://www.statmt.org/moses/?n=Development.GetStarted

(2) You'd better run the baseline of Moses. <http://www.statmt.org/moses/?n=Moses.Baseline>

(3) Python 2.7.

Here we use irstlm to train language model, as this baseline suggest.

## 1.2 parallel corpus

for train:

gene.6.pr : gene sequence, have been segmented, maximal word lenght 6

gene.6.go:  gene ontology sequence

for test:

gene.6.pr.test:  gene sequence

gene.6.go.test: gene ontology sequence

## 1.3 train language model of target sequence, here we train gene ontology model

file: train\_lm.sh

(attention: please revise the moses\_path and irstml\_path to your own path!)

run:

./train\_lm.sh gene.6.go

You will get gene.6.go.blm.

## 1.4 train translation model

File: train\_ts.sh

(attention: please revise the moses\_path and  giza\_path to your path!)

run:

./train\_ts.sh gene.6 gene.6.go.blm

Wait about 4 hours (1 CPU. 10+G memory requirement)

You will get a "train" directory.

The train/model/moses.ini is just your translation model file

## 1.5 test

File: get\_bleu.sh

(please revise the moses\_path)

run:

./get\_bleu.sh gene.6.go.test gene.6.pr.test

it will translate gene.6.pr.test and then compare it with gene.6.go.test. If there is no error, you will get the result:

*BLEU = 26.38,* ***47.4****/30.0/21.4/15.9 (BP=1.000, ratio=1.571, hyp\_len=48369, ref\_len=30786)*

Because we don't consider the order of go terms, 47.4 is just its BLEU score (unigram). You could still divide the test corpuses into test part and tuning part, and then run the tuning process (tuning.sh). It could bring some improvement for bleu score, but it may take several days.

# 2 Get gene data

Directory: get\_gene\_data

If you want build your own corpus from original database source, see this section:

**1 get gene data**

here we will show how to get corpus from original data source. you need install gunzip.

using human data:

File : get\_gene\_data.sh, its operation

(1). get original gene ontology data. (ftp of www.geneontology.org).

(2). get gene sequence data.(ftp of www.uniprot.org). ()

(3). gunzip

(4). convert gene sequence format. (get\_gene.py)

(5). convert gene ontology format.( get\_go.py)

(6). Get parallel corpus according to gene database id. (get\_corpus.py)

just run ./get\_gene\_data.sh

We will get parallel corpus

human.pr.filter gene sequene

human.go.filter corresponding gene ontology sequence

This corpus contain about 1,8000 data.

You could also download the big corpus:

Gene ontology files (UniProt [multispecies]):

<http://www.geneontology.org/gene-associations/submission/gene_association.goa_uniprot.gz>

contain about 40 millions data.

Related sequence data could be found at:

UniProtKB/Swiss-Prot,

<ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_sprot.fasta.gz>

UniProtKB/TrEMBL,

<ftp://ftp.uniprot.org/pub/databases/uniprot/current_release/knowledgebase/complete/uniprot_trembl.fasta.gz>

You could download these two files then cat them together. Then revise the ./get\_gene\_data.sh. (refer to get\_big\_gene\_data.sh). Finally, you could get about 20 million parallel corpus.

# 3 segment gene sequence and clean the corpus

Directory: pre\_process\_gene\_data

If you want to build your own corpus, you will need segment the gene sequence and clean the corpus.

## 2.1 data files

(1) mix.go original gene ontology sequence

(2) mix.pr original gene sequence

(Here we give these data. You could also copy the human.pr.filter in previous to mix.go, and human.go.filter to mix.go to run your own experiment, also for the big data set)

(3) protein.dict dict file

## 2.2 preprocess the file, get the parallel corpus

File : pre\_process.sh

just run:

./pre\_process.h

Its operation:

step 1:segment the gene sequence. (segment\_gene\_sequence.py)

step 2:clean the corpus.( clean\_corpus.py)

step 3:divide the corpus into train and test parts.( divide\_corpus.py)

finally, you will get 4 files

gene.pr : gene sequence, have been segmented, maximal word length 6

gene.go: gene ontology sequence

gene.pr.test: gene sequence for test

gene.go.test: gene ontology sequence for test

These files could be used to train the translation model. These could be used for section 1.

# 3 Build gene dictionary

Directory: build\_dict

<oreover, if you want to build your own protein.dict for different maximal word length, prepare protein sequences and see segment part of this project. Here we also give an example:

## Step 1, install SRILM

A simple method to build dictionary is to use language model. Here use SRILM. You should install it first. ( <http://www.speech.sri.com/projects/srilm/> )

Then cp the executable file “ngram-count” to this directory. Normally in your install dir. There have been an “ngram-count”, but it could only run in a special Linux version, so just overwrite it.

## Step 2, you should make 2 file

1. In ./get\_gene\_word : run make
2. In ./get\_gene\_word\_prob : you should set the srilm install path and MACHINE\_TYPE in Makefile, then run make.

## Sept 3, uncompress the data file,

Run: tar -xzvf protein.fa.tar.gz

## Step 4, train n-gram language model, n=1-5

Run: ./ build\_all\_lm\_model.sh

(Attention, you must cp “ngram-count” to this directory. See step1)

## Step 5, Build dictionary with different maximal word length

Run: ./build\_dict\_all.sh

Its operation :

1. Get all possible gene word, filter them by frequency. (get\_word)
2. Get probability of all gene words. (get\_prob)
3. Filter the gene words in dictionary by MI method. (mi\_filter.py)

Finally, you will get protein.dict.\*.mi , (\* is 1,2,…,5) is the dictionary with maximal word length \*. Then you could use these dictionary file to segment the gene sequence (in section 3).

If you have map/reduce cluster, you could use EM method to build the gene word. See DNA segment part of this project. To train an 8 maximal word length dictionary, you need at least 4G protein sequence data. More gene sequence could also find in section 1.