**Report on hmwk1**

Yifei Huang

High level structural 1

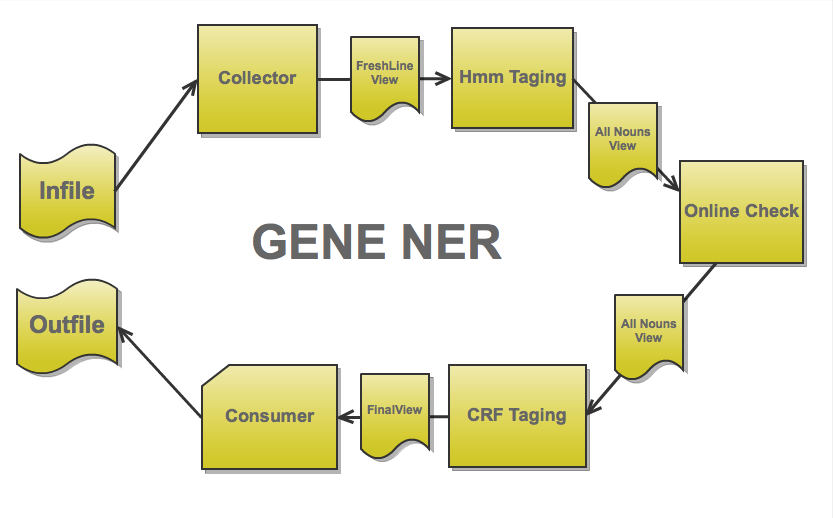
TypeSystem: 2

More work: 3

Resources: 3

High level structural

The whole system is built on one CPE. Inside the CPE it has one collector, one consumer and three analyses. The structural is showed below:



Collector:

In the collector, it will chunk the file into sentences. That is each sentence is a Cas. Then it parse it to get ID and plain text. After that an annotation:*myCpe.line* is created to save these information. Finally, each annotation will be put into a view called “freshline” for the further use.

Hmm Taging:

In this component, pre-trained hmm model for gene is involved. Basically it will grab plain text of the sentence form Freshline View, and Tag the sentence into phrases. Each phrase will have a confidence from 0-1, representing their possibility to be gene phrases. Then for each phrase, if it has a confidence larger than certain threshold, a nounPhrase annotation will be created to store its information. All the nounPhrase annotation goes into an “allNouns” view finally.

Online Checking:

In this component, all the nounPhrase annotation will be checked from “allNouns” view. If it has a confidence lower than some value, it means it’s hard to say whether it is or not. As a result, we may want to check the phrase online. Inside the component an open source online gene database is connected. If the ambiguous phrases can be found, we than set its confidence into the baseline threshold of CRF component, which will be discussed soon.

CRF Tagging:

This component takes all the nonnPhrase annotation from “allNoun” view and averages them into a score. If the score is higher than or equal to the threshold mentioned in the online checking part, we make consider this sentence is gene related. If it does, the plaintext from “freshLine” view will be load again and put into the pretrained CRF model. The CRF model is a very accurate but slow tagger. As a result, we may want to reduce the unnecessary work for it. As a result, all the previews tagging is just a way to eliminate the work for the CRF model. So that it does not need to waste time on some obvious no gene-related sentence. This approach may look useless in the sample.in file. However, if the test file do contains a lot of noise, it will be eliminated by this method.

TypeSystem:

The whole system online contains one type system:lineType. Inside it two types are involved.

lineType:

* myCpe.line
  + id
  + sentence
* myCpe.nounPhrases
  + nouns
  + confidence

myCpe.line contains the basic information of one sentence of input file. It includes the id and the plain text for the body of the sentence.

myCpe.nounPhrases contains the information for one noun phrases appear in sentence. Inside, nouns save the plaintext of the noun phrases. Confidence save the confidence this phrase is a gene phrase. The confidence is derived from trained models.

Conclusion:

The whole system runs 4mins in a 15000-line file.(sample.in).

With the Hmm + webChecker fileter only 7417 out of 15000 sentences were scanned by CFG TAGGER and get an accuracy of 0.997322 and the recall of 0.958226. The F-measure is 0.978.

If we use the pure CFG Tagger instead, on the same test file(sample.in), it takes 8 mins to finish. The accuracy is 0.991077 and the recall is 0.931837. F-measure:0.96

If we use only hmm Tagger the accuracy and recall will all be around 0.8.

As we can see with the pipelined structure, the performance of CFG tagger is increased much in running time and some improvement in F-measure.

However, due to the lack of test data, it’s hard to make this result robust. Further test is required.

Warning: Due to the high F-measure (0.97), I highly suspect the training model is built on the a larger data set which includes the sample.in file. According to the resource website information, the CFG data is trained based on BioCreative 2 gene mention task (Training)(sponsored by IBM). As a result, our project’ dataset my be covered in them.

More work:

Besides the current system, the following idea was not tried or abandoned.

1. hashmap + nearest neighbor (I only get an offline database for 8mb so I did not try it)
2. HMM tag + CFG tag + merging (worth trying not enough time.)

Resources:

Hmm model: http://alias-i.com/lingpipe/

CFG model: http://cbioc.eas.asu.edu/banner/

Gene database: http://bergmanlab.smith.man.ac.uk:8081/?help