600.468 Machine Translation **Spring 2015**
Juan Marron

Evaluation Metrics

1 Basic METEOR

The first step I took towards improving the evaluation metric was to implement METEOR. This was a fairly straightforward implementation. METEOR simply uses the harmonic mean of precision and recall:

$$\ell(h,e) = \frac{P(h,e)\cdot R(h,e)}{(1-\alpha)R(h,e)+\alpha P(h,e)}$$

Implementing this did not involve much modification to the existing baseline evaluator, but it resulted in drastic improvements in evaluation accuracy. I had to make functions for both precision and recall, which are $P(h,e) = \frac{|h \cap e|}{|h|}$ and $R(h,e) = \frac{|h \cap e|}{|e|}$ respectively. Then, you plug in the values in the harmonic mean formula above for each pair of hypothesis and reference sentences, and for each reference output the sentence with the highest harmonic mean. I tested multiple values of alpha to see which value yielded the highest accuracy:

Baseline	.4493
alpha	Accuracy
0	.4540
.1	.4979
.2	.4984
.3	.4996
.4	.5008
.5	.5016
.6	.5028
.7	.5038
.8	.5033
.9	.5042
1.0	.5022

When alpha is zero, the harmonic mean simply reduces to just being the precision since $\ell(h,e) = \frac{P(h,e)\cdot R(h,e)}{(1-0)R(h,e)+0*P(h,e)} = \frac{P(h,e)\cdot R(h,e)}{R(h,e)} = P(h,e)$. This is almost as bad as the baseline accuracy of .4493, and that makes sense since it does almost exactly the same thing, except normalized over the length of the hypothesis sentence. As alpha increases, the results get better and better until an optimal of about $alpha = .9$ and accuracy of .5042. When alpha is .9, recall is weighted about 9 times more than precision. With a little more tuning, I found that .87 was the most accurate alpha I could find, with an accuracy of .5045.

2 METEOR with penalties

A modification to the METEOR metric allows us to consider not only individual unigrams, but also contiguous stretches that match in both the hypothesis and the reference sentence. We assign a penalty if there are not many n-gram matches. This improves the metric a little bit because incorrect word orders are penalized while having long contiguous n-gram matches (which likely means the sentence is a good translation) are rewarded and weighted more heavily. The expression for METEOR with the chunking penalty is the following:

$$\ell(h,e) = \left(1 - \gamma \left(\frac{c}{m}\right)^{\beta}\right) \frac{P(h,e) \cdot R(h,e)}{(1-\alpha)R(h,e) + \alpha P(h,e)}$$

Using a beta of 3.0, as Wikipedia suggested and maintaining my optimal alpha of .87, I played around with the gamma setting and got the following results:

gamma	Accuracy
0	.5045
.1	.5046
.2	.5045
.3	.5049
.4	.5031
.5	.4998
.6	.4910
.7	.4755
.8	.4486
.9	.4210
1.0	.5022

As you can see, the best accuracy is when gamma is around .3. With a little more fine tuning, I managed to push the accuracy to .5052 when gamma was .32. This is an improvement, but a fairly marginal one since it's only .0007 higher than METEOR without any chunking penalty.

3 WordNet

The final thing I implemented was using wordnet in order to match synonyms. I used the nltk python library to include the wordnet corpus and use the methods to get synonyms. Although this took longer to implement than either METEOR metric, it sadly did not yield as much of an improvement as I would have hoped. I generated all the synonyms for words in the reference sentence, then determined which words in the hypothesis matched with either a word in the reference sentence or any of its synonyms. After I had those matches, I calculated the precision and recall and used those numbers in the METEOR with chunking penalty formula. Using the synonyms, I achieved a final accuracy score of .5081.