Alternative Criteria For Optimization

As we have previously seen, an HMM system using the standard Baum-Welch reestimation algorithm learns to emulate the statistics of the training database. We refer to this training mode as “representation.” This is not necessarily equivalent to minimizing the recognition error rate. It depends to a great deal on the extent to which the statistics of the training database match the test database. Often, especially in open-set speaker independent cases, the test database contains data never seen in the training database.

A potential solution to this problem is to attempt to force the recognizer to learn to discriminate (reduce recognition errors explicitly). This is analogous to developing a system to make an M-way choice in an N-dimensional space by learning decision boundaries rather than clustering the data and modeling the statistics of each cluster.

One approach to modifying the HMM structure to do this is to use a different cost function. For example, we can minimize the discrimination information (or equivalently the cross-entropy) between the data and the statistical characterizations of the data implied by the model.

Recall the definition of the discrimination information:

$$J_{DI} = \int_{-\infty}^{\infty} f(\underline{\boldsymbol{y}}|y)\log\frac{f(\underline{\boldsymbol{y}}|\boldsymbol{y})}{f(\underline{\boldsymbol{y}}|M)}d\underline{\boldsymbol{y}}$$

Note that a small DI is good, because it implies a given measurement matches both the signal statistics and the model statistics (which means the relative mismatch between the two is small).

However, such approaches have proven to be intractable — leading to highly nonlinear equations. A more successful approach has been to maximize the average mutual information. Recall our definition for the average mutual information:

$$\overline{M}(\underline{y},\underline{M}) = \sum_{l=1}^{L}\sum_{r=1}^{R} P(\underline{y} = y^{l},\underline{M} = M_{r})\log\left[\frac{P(\underline{y} = y^{l},\underline{M} = M_{r})}{P(\underline{y} = y^{l})P(\underline{M} = M_{r})}\right]$$

This can be written as:

$$\bar{M}(\underline{y},\underline{M}) = \sum_{l=1}^{L}\sum_{r=1}^{R} P(\underline{y}=y^{l},\underline{M}=M_{r})\times \log[P(\underline{y}=y^{l},\underline{M}=M_{r})] - \log[P(\underline{y}=y^{l})]$$

$$= \sum_{l=1}^{L}\sum_{r=1}^{R} P(\underline{y}=y^{l},\underline{M}=M_{r})\times [\ \log P(\underline{y}=y^{l},\underline{M}=M_{r}) - \log\sum_{m=1}^{R} P(\underline{y}=y^{l}\,|\,\underline{M}=M_{m})P(\underline{M}=M_{m})\ ]$$

Note that the last term constitutes a rewrite of $P(\underline{y}=y^{l})$.

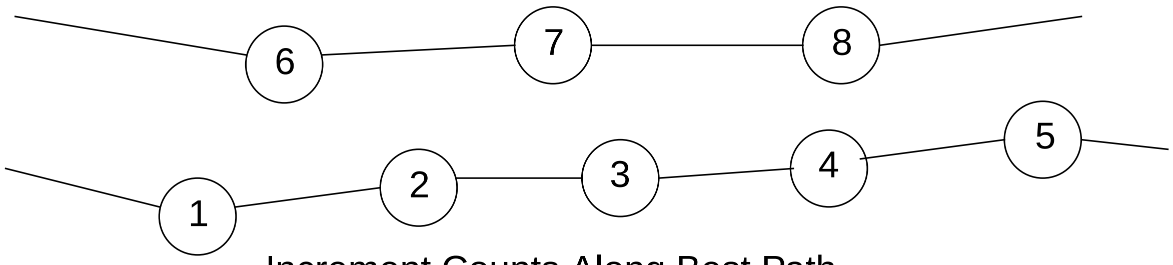
If we assume there is exactly one training string (which is the error we want to correct), and it is to be used to train M_l, then, if we assume $P(\underline{y}=y^{l}\,|\,\underline{M}=M_{r}) \approx \delta(l-r)$, we can approximate $\bar{M}(\underline{y},\underline{M})$ by:

$$\bar{M}(\underline{y},\underline{M}) \approx \sum_{l=1}^{L} P(\underline{y}=y^{l},\underline{M}=M_{l}) - \log\sum_{m=1}^{R} P(\underline{y}=y^{l}\,|\,\underline{M}=M_{m})P(\underline{M}=M_{m})$$

The first term in the summation corresponds to the probability of correct recognition, which we want to maximize. The second term corresponds to the probability of incorrect recognition, which we want to minimize.

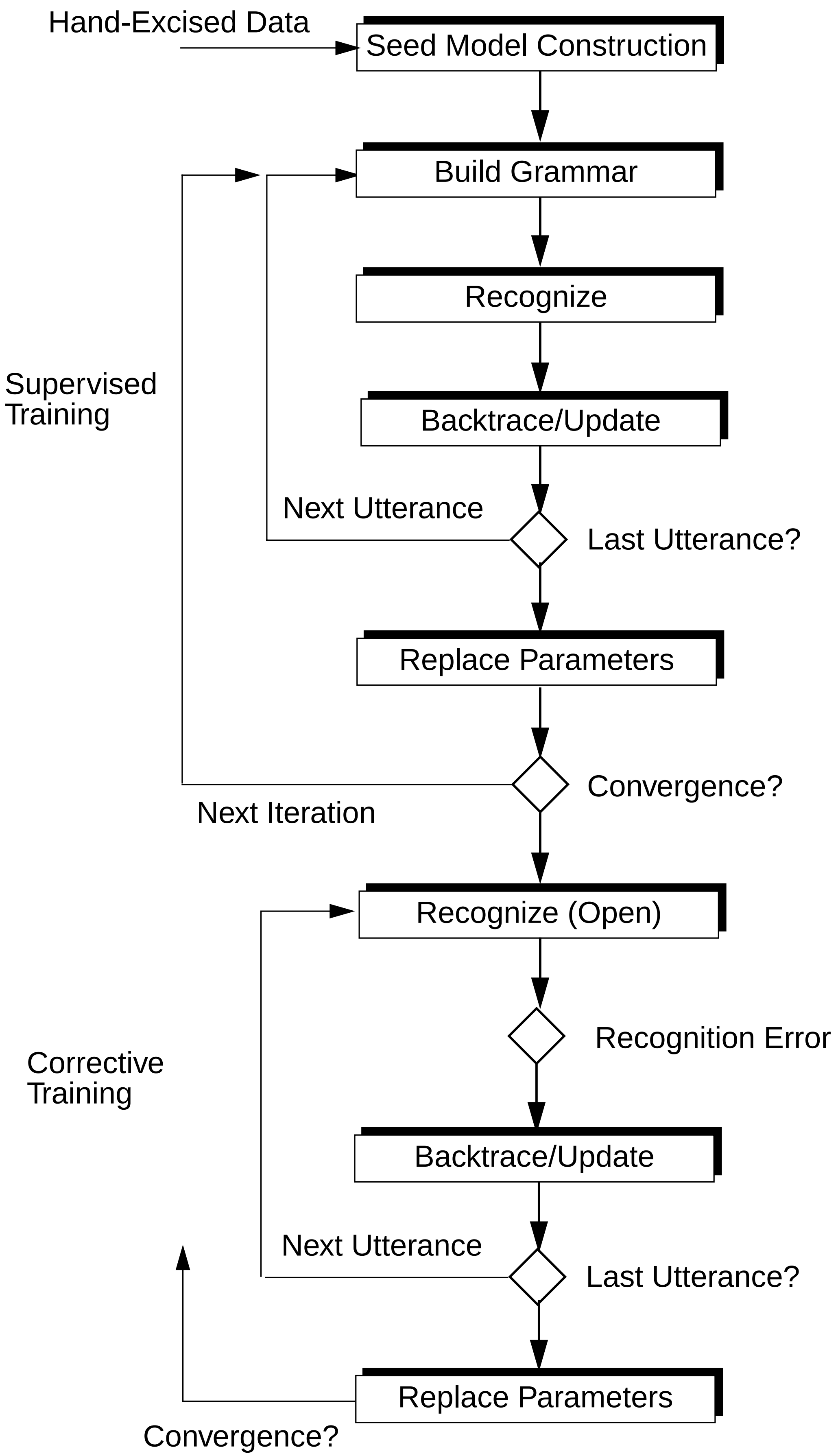
This method has a rather simple interpretation for discrete HMMs:

Decrement Counts Along Competing Incorrect Paths



Increment Counts Along Best Path

An Overview of the Corrective Training Schedule



Unfortunately, this method is not readily extensible to continuous speech, and has proved inconclusive in providing measurable reductions in error rate. However, discriminative training algorithms continues to be an important are of research in HMMs.

Later, when we study neural networks, we will observe that some of the neural network approaches are ideally suited towards implementation of discriminative training.

Distance Measures for HMMs

Recall the KMEANS clustering algorithm:

Initialization: Choose K centroids

Recursion:

1. Assign all vectors to their nearest neighbor.
2. Recompute the centroids as the average of all vectors assigned to the same centroid.
3. Check the overall distortion. Return to step 1 if some distortion criterion is not met.

Clustering of HMM models is often important in reducing the number of context-dependent phone models (which can often approach 10,000 for English) to a manageable number (typically a few thousand models are used). We can use standard clustering algorithms, but we need some way of computing the distance between two models.

A useful distance measure can be defined as:

$$D(M_1,M_2) \equiv \frac{1}{T_2}[\log P(y^2 \mid M_1) - \log P(y^2 \mid M_2)]$$

where y^2 is a sequence generated by M_2 of length T_2. Note that this distance metric is not symmetric: $D(M_1,M_2) \neq D(M_2,M_1)$

A symmetric version of this is:

$$D'(M_1,M_2) = \frac{D(M_1,M_2) + D(M_2,M_1)}{2}$$

The sequence y^2 is often taken to be the sequence of mean vectors associated with each state (typically for continuous distributions).

Often, phonological knowledge is used to cluster models. Models sharing similar phonetic contexts are merged to reduce the complexity of the model inventory. Interestingly enough, this can often be performed by inserting an additional network into the system that maps context-dependent phone models to a pool of states.