Cambridge University Engineering Department

April 24th 2003

Dan Povey & Phil Woodland

Recent work on Discriminative Training
* SAT for discriminative training (relates to MLR)

MAP

- Adaptation, e.g. Geneder adaptation with discriminative training (MPE)
  
  - Also w/ MPE?
  
  - Minimum Phone Error (MPE)

- Work on implementing MMI for LCSR (using lattices)

Recent work at Cambridge on discriminative training includes:

- of recognition of train-data

  - But maximizing some other criterion (e.g. MMI) which reflects goodness...

- Discriminative training is training HMM parameters not using ML

Discriminative Training
Review of implementation issues

Typical results for MPE vs MMI vs ML

MPE objective function

Review
are more accurate than average

When maximizing criterion, we try to increase likelihood of sentences which

errors

where \( \text{Raw\ Phone\ Accuracy}(s, s') \) is #phones in reference, minus #phone

\( \text{sentence likelihood} \)

\[
\left( \sum_{s} \sum_{s'} \sum_{R} P_{\text{MPE}}(s, s') \right) ^{t}
\]

where #sentence likelihood

\( \text{MPE} \) phone

Maximise the following function:

Minimum Phone Error (MPE)
Maximum Mutual Information (MMI)
Comparison of objective functions (for 2 sentences)
Improvement over ML

Without I-smoothing, MPE is worse than WMI and gives only small

(in MAP, evidence is speaker-dependent ML objective function)

—in I-smoothing, evidence is discriminative objective function
—We use a prior over the parameter values, center of prior is at ML estimate
—Mathematically, I-smoothing is like MAP

values where there is not enough training data for a Gaussian

We use a technique we call I-smoothing, to back off parameters to the ML

This is especially true for MPE

model parameters (overtraining)

Discriminative objective functions make it difficult to get robustly estimated

Prior Information for Robust Parameter Estimates

Prove: Minimum Phone Error
Improvement vs. MLL
Comparison of MPE with MMI, I-smoothed MMI
Although the winning result was a combination of results from two other sites.

This year we improved our system further, but were slightly beaten.

<table>
<thead>
<tr>
<th></th>
<th>MLIR</th>
<th>No MLIR</th>
<th>(% WERS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28.5%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30.1%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30.7%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33.3%</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>% Rel Impr</td>
<td>%PE</td>
<td></td>
<td>ML</td>
</tr>
</tbody>
</table>

Our system was the best.

From 2002 NIST evaluation, tested on subset of 2001 development data.

E.g. of MPE for an evaluation switchboard system.
backward algorithm

This can be calculated efficiently using an algorithm similar to the forward-

the lattice

Need to find this differentially without enumerating each possible sentence in

data-likelihood of each phone in the lattice

To construct the auxiliary function, need differential of objective function w.r.t.

Uses a "weak-sense" auxiliary function (see next slide)

Optimised in a number of iterations, on each iteration, optimise an auxiliary

Optimisation of MPE
Weak-sense aux: has same differential around local point $x = x'$

at a local point $x = x'$, but $\nabla f \neq 0$ elsewhere else

Strong-sense auxiliary function: has the same value as real objective function

Use of (a) strong-sense and (b) weak-sense auxiliary functions for function

Auxiliary Functions
What if a problem has no simple solution?

Traditional science expects simple solutions (e.g., physics)

Is there a "simple" solution?

Clearly simple is better if we have the choice, but...

Should we be looking for complex or simple solutions to the speech recognition problem?

Interesting question:

On another topic...
as \( \text{log(length)} \) of a particular description length.

- Goodness of solution for the best solution with a particular description length.

```
<table>
<thead>
<tr>
<th>Program complexity</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>2</td>
<td>0.03</td>
</tr>
<tr>
<td>3</td>
<td>0.06</td>
</tr>
<tr>
<td>4</td>
<td>0.10</td>
</tr>
<tr>
<td>5</td>
<td>0.15</td>
</tr>
<tr>
<td>6</td>
<td>0.20</td>
</tr>
<tr>
<td>7</td>
<td>0.25</td>
</tr>
<tr>
<td>8</td>
<td>0.30</td>
</tr>
<tr>
<td>9</td>
<td>0.36</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>Program complexity</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.01</td>
</tr>
<tr>
<td>1</td>
<td>0.02</td>
</tr>
<tr>
<td>2</td>
<td>0.03</td>
</tr>
<tr>
<td>3</td>
<td>0.04</td>
</tr>
<tr>
<td>4</td>
<td>0.05</td>
</tr>
<tr>
<td>5</td>
<td>0.06</td>
</tr>
<tr>
<td>6</td>
<td>0.07</td>
</tr>
<tr>
<td>7</td>
<td>0.08</td>
</tr>
<tr>
<td>8</td>
<td>0.09</td>
</tr>
<tr>
<td>9</td>
<td>0.10</td>
</tr>
</tbody>
</table>

```

"SOLUBLE" vs "INSOLUBLE" problems.
- Use evolution (not match research) as a model for how to solve the problem
- Swap code (and write programs so this is possible)

Language:
- Find new representations of the solution (e.g., weird new programming
problem
- Find convenient ways of creating and transmitting complex solutions to the

Some ideas:

... so what can we do (other than give up)?

If this is right, we can't find a "solution" that can be written in a few pages.

"Soluble" vs "unsolvable" problems cont'd