Can we build a voice-recognition system that avoids overfitting by explicitly defining small supervectors, instead of implicitly defining large ones?

**METHODS:**

- **Neural Tensor Network** (n=729)
  - Format:
  - Output:
  - [PCA representation of tensor supervectors (2-class)]
  - [Normalized confusion matrix]

- **Softmax Neural Network** (n=79)
  - Format:
  - Output:
  - [PCA representation of diagonal supervectors (2-class)]
  - [Normalized confusion matrix]

- **Support Vector Machine** (n=53)
  - Format:
  - Output:
  - [PCA representation of concatenated vectors (2-class)]
  - [Normalized confusion matrix]

**RESULTS:**

<table>
<thead>
<tr>
<th>9-Class Statistics:</th>
<th>NTN</th>
<th>SNN</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train Accuracy</td>
<td>18.2%</td>
<td>38.6%</td>
<td>47.8%</td>
</tr>
<tr>
<td>Test Accuracy</td>
<td>14.8%</td>
<td>35.8%</td>
<td>44.6%</td>
</tr>
<tr>
<td>F1</td>
<td>17.3%</td>
<td>37.2%</td>
<td>44.4%</td>
</tr>
<tr>
<td>% improvement</td>
<td>4.2%</td>
<td>27.8%</td>
<td>37.7%</td>
</tr>
</tbody>
</table>

**CONCLUSION:**

Explicitly defined supervectors create highly non-convex objectives, which are too difficult to optimize using Stochastic Gradient Descent.