Classifier Combination

Kuncheva Ch. 3
Motivation

Classifiers

Are functions that map feature vectors to classes.

- Any single classifier selected will define this function subject to certain biases due to the space of models defined, the training algorithm used, and the training data used.

- **Idea:** use a ‘committee’ of base classifiers, map a vector of **class outputs** or **discriminant values** for each base classifier to the output classes. (**Parallel Combination**)
  - **Fusion:** base classifiers cover feature space; **Selection:** classifiers are **assigned** to produce classes for a region in feature space

- **Another Idea:** organize classifiers in a **cascade** (list) or hierarchy, to allow progressively more specialized classifiers to refine intermediate classification decisions, e.g. to classify low-confidence or rejected samples. (**Hierarchical/Sequential Combination**)
Effectivene Combinations: Statistical Reason

Dietterich [106] suggests three types of reasons why a classifier ensemble might be better than a single classifier.

3.1.1 Statistical Reason

Suppose we have a labeled data set $Z$ and a number of different classifiers with a good performance on $Z$. We can pick a single classifier as the solution, running onto the risk of making a bad choice for the problem. For example, suppose that we run the 1-nn classifier or a decision tree classifier for $L$ different subsets of features thereby obtaining $L$ classifiers with zero resubstitution error. Although these classifiers are indistinguishable with respect to their (resubstitution) training error, they may have different generalization performances. Instead of picking just one classifier, a safer option would be to use them all and “average” their outputs. The new classifier might not be better than the single best classifier but will diminish or eliminate the risk of picking an inadequate single classifier.

Dietterich gives a graphical illustration of this argument as shown in Figure 3.1. The outer circle denotes the space of all classifiers. The shaded inner region contains all classifiers with good performances on the training data. The best classifier for the problem (supposedly with a good performance on the training data too) is denoted by $D_*/C^3$. The hope is that some form of aggregating of the $L$ classifiers will bring the resultant classifier closer to $D_*/C^3$ than a classifier randomly chosen from the classifier space would be.

**Fig. 3.1** The statistical reason for combining classifiers. $D^*$ is the best classifier for the problem, the outer curve shows the space of all classifiers; the shaded area is the space of classifiers with good performances on the data set.

“Average” classifier outputs to produce better estimates.

D1-D5 produced with 0 resubstitution error for different feature subsets (e.g. for 1-NN or decision tree)
Effective Combinations: Computational Reason

Aggregation of local-search optimizations may improve over individual (local) error minima.

D1-D4: classifiers trained using hill climbing (e.g. gradient descent), random search.

**Fig. 3.2** The computational reason for combining classifiers. $D^*$ is the best classifier for the problem, the closed space shows the space of all classifiers, the dashed lines are the hypothetical trajectories for the classifiers during training.
Effective Combinations: Representational Reason

Combination allowing decision boundaries not expressible in original classifier parameter space to be represented

Fig. 3.3 The representational reason for combining classifiers. $D^*$ is the best classifier for the problem; the closed shape shows the chosen space of classifiers.
Classifier Output Types
(Xu, Krzyzak, Suen)

Type 1: Abstract Level
Chosen class label for each base classifier

Type 2: Rank Level
List of ranked class labels for each base classifier

Type 3: Measurement Level
Real values (e.g. [0, 1]) for each class (discriminant function outputs)
Fusion Combinations ($k$ base classifiers)

For Type 1 (Single Label per Base Classifier)

$$D_l(x) = C(B(x))$$

$B : \mathbb{R}^n \rightarrow \Omega^k$, $C : \Omega^k \rightarrow \Omega$

Combination example: voting

For Type 2 (Ranked list of $r$ classes)

$$D_r(x) = C(B(x))$$

$B : \mathbb{R}^n \rightarrow \Omega^{rk}$, $C : \Omega^{rk} \rightarrow \Omega$

Combination example: weighted voting (e.g. Borda Count)

For Type 3 (Discriminant Values)

$$D_m(x) = C(B(x))$$

$B : \mathbb{R}^n \rightarrow \mathbb{R}^{\Omega^k}$, $C : \mathbb{R}^{\Omega^k} \rightarrow \Omega$

Combination example: min, max, product rules
Classifier Ensembles (Fusion): Combination Techniques

A similar variety of terminology can be observed in the toolbox of methods for classifier combination. A starting point for grouping ensemble methods can be sought in the ways of building the ensemble. The diagram in Figure 3.4 illustrates four approaches aiming at building ensembles of diverse classifiers.

This book is mainly focused on Approach A. Chapters 4, 5, and 6 contain details on different ways of combining the classifier decisions. The base classifiers (Approach B) can be any of the models discussed in Chapter 2 along with classifiers not discussed in this book. Many ensemble paradigms employ the same classification model, for example, a decision tree or a neural network, but there is no evidence that this strategy is better than using different models. The design of the base classifiers for the ensemble is partly specified within the bagging and boosting models (Chapter 7) while designing the combiner is not coupled with a specific base classifier. At feature level (Approach C) different feature subsets can be used for the classifiers. This topic is included in Chapter 8. Finally, the data sets can be modified so that each classifier in the ensemble is trained on its own data set (Approach D). This approach has proven to be extremely successful owing to the bagging and boosting methods described in Chapter 7.

Although many of the existing streams in classifier combination are captured in the four-approaches classification, there are many more that are left outside. For example, a remarkably successful ensemble building heuristic is manipulating the output labels by using error correcting codes (ECOC) (Chapter 8). Other topics of interest include clustering ensembles (Chapter 8) and diversity in classifier ensembles (Chapter 10). Developing a general theory, as impossible as it sounds.

Fig. 3.4 Approaches to building classifier ensembles.
Cascade Architecture


Here a set of classifiers is obtained using AdaBoost, then partitioned using dynamic programming to produce a cascade of binary classifiers (detectors) (Viola and Jones face detector (2001, 2004))
ECOC: Another Label-Based Combiner

Error-Correcting Output Codes (ECOC)

Classifier ensemble comprised of binary classifiers (each distinguishing a subset of the class labels: dichotomizers)

- Represent base classifier output as a bit string
- Learn/associate bit string sequences with concrete labels; classify by Hamming distance to bit string (‘code’) for each class
  \[
  \mu_j(x) = -\sum_{i=1}^{L} |s_i - C(j, i)|
  \]
  (‘support’/discriminant value)
- Details provided in Ch. 8 (Kuncheva)

(s₁, ..., s₇) = (0, 1, 1, 0, 1, 0, 1).

Hamming Distances:
- class 1: 5
- class 2: 3
- class 3: 1
- class 4: 5

<table>
<thead>
<tr>
<th></th>
<th>D₁</th>
<th>D₂</th>
<th>D₃</th>
<th>D₄</th>
<th>D₅</th>
<th>D₆</th>
<th>D₇</th>
</tr>
</thead>
<tbody>
<tr>
<td>ω₁</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ω₂</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ω₃</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>ω₄</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Training Combiners: Stacked Generalization

**Protocol:**

Train base classifiers using cross-fold validation

Then train combiner on all N points by using the class labels output by the base classifiers for each fold (train/test partition)

*Fig. 3.5 Standard four-fold cross-validation set-up.*
There is nothing new under the sun...

Sebeysten (1962)

Idea of using classifier outputs as input features, classifier cascade architectures

Dasarthy and Sheila (1975)

Classifier selection using two classifiers

Rastrigin and Erenstein (1981 - Russian)

Dynamic classifier selection

Barabash (1983)

Theoretical results on majority vote classifier combination