Using Conflicts Among Multiple Base Classifiers to Measure the Performance of Stacking

We analyze the machine learning bias of stacking and point out the conflict problem. Conflicts are defined as base data with different class labels that produced the same predictions by a set of base classifiers. Based on conflicts, we propose conflict-based accuracy estimate to determine the overall accuracy of a stacked classifier and conflict-based accuracy improvement estimate to determine the overall accuracy improvement over base classifiers. We discuss some popular metrics for comparing and evaluating a set of classifiers: coverage, correlated error, diversity and specialty, and show that these metrics do not accurately estimate the overall accuracy of a stacked classifier system. From experimental results, we demonstrate that conflict-based accuracy estimate is an effective measure to predict overall performance and compare different stacked systems, and conflict-based accuracy improvement estimate is a good measure to project the overall accuracy improvement.

1 Introduction

Stacking [16] is a widely known technique to combine classifiers [7]. Empirical studies have shown that stacking helps increase accuracy. Many papers in recent years have concentrated on using various metrics, coverage [2], diversity [2, 4], correlated error [1] and specialty [4], to explain stacking and choose classifiers for combining. Choosing the best base classifiers is an important issue to increase accuracy and efficiency of a stacked classifier system. In this paper, we will analyze the machine learning bias of stacking and discuss a problem called conflicts that prevents the accuracy improvement of stacking. Based on conflicts, we propose a direct measure, conflict-based accuracy estimate, to determine the overall accuracy of a stacked system and conflict-based accuracy improvement estimate to predict the accuracy improvement. We will show that, except for coverage, all the other metrics mentioned above are indirect measures of stacking and it is hard to use them for predicting overall accuracy. Experimental results demonstrate that conflict-based accuracy estimate is an effective measure to predict performance and compare different stacked systems.

The paper is organized as follows. First, we analyze the machine learning bias of stacking and propose quantitative measures for it. The paper follows with a comparison to previously proposed metrics. We conclude by a discussion of the results of experiments and future work.

2 Conflict and Measures

2.1 Stacking and Its Conflict Problem

Following Wolpert[16], the general scheme for stacking works as follows. There are $t$ different algorithms $\{A_1, \ldots, A_t\}$ and a set of training examples $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$. $S$ is CV (or cross-validation) partitioned into $n$ pairs $\{(T_1, V_1), \ldots, (T_n, V_n)\}$ ($2 \leq n \leq m$). We train $A_1$ to $A_t$ on $T_k$ to produce classifiers $C_1$ to $C_t$ and apply these classifiers to predict on $V_k$ to obtain predicted class labels. $\forall (x_i, y_i) \in V_k$, we form a new training item: $\left( (C_1(x_i), C_2(x_i), \ldots, C_t(x_i)), y_i \right)$. The process is repeated for all $n$ pairs of $(T_k, V_k)$ to generate a new data set, $M_{training}$. $A_1$ to $A_t$ are re-applied on the complete training set $S$ to produce base classifiers $C_1$ to $C_t$. We may use any algorithm to
learn the new meta-level training set, $M_{\text{train}}$, to generate the meta-classifier, $M_C$. During testing, $C_1, \ldots, C_t$ first generate predictions. Their predictions form a meta-level testing item $(C_1(x), \ldots, C_t(x))$ that is given to the meta-classifier. The meta-classifier’s prediction $M_C((C_1(x), \ldots, C_t(x))$ is the final outcome.

For simplicity, we consider binary problems in which $y \in \{0, 1\}$. The prediction $C_i(x)$ of classifier $C_i$ on data item $(x, y)$ is abbreviated in lowercase $c_i$. $(c_1, c_2, \ldots, c_t)$ is thus a meta-level training data item. The feature vector $(c_1, c_2, \ldots, c_t)$ is further abbreviated as $c$. Each training data item $(x, y)$ will fall into exactly one particular $c$. For binary problems, there are at most $2^t$ different kinds or combinations of $c$’s. Figure 1 shows the 8 combinations of two boolean data sets (see Section 3). To understand how stacking works, for a fixed combination $c$, we count all the occurrences of the same instance $(c, 1)$ in the meta-level training data (denoted $|c, 1|$), and all occurrences of $(c, 0)$ (denoted $|c, 0|$). We define accuracy $\alpha$ and conflict ratio $\epsilon$ on the combination $c$ as:

$$\alpha = \frac{\max(|c, 1|, |c, 0|)}{|c, 1| + |c, 0|}, \quad \epsilon = 1 - \alpha$$

For conflict $c = (0, 1, 1)$ of BOOLEAN45 in Figure 1, $|(0, 1, 1), 1| = 1247$ and $|(0, 1, 1), 0| = 1445$, so $\alpha = 1445/(1247 + 1445) = 0.537$ and $\epsilon = 1247/(1247 + 1445) = 0.463$.

Since the feature vector $c = (c_1, c_2, \ldots, c_t)$ is the same for both $(c, 0)$ and $(c, 1)$, any machine learning algorithm has no way to distinguish between them. The best it can do is to pick the label of the majority class, and the minority occurrences will always be labelled incorrectly. For the $(0,1,1)$ type discussed above, during testing, whenever the meta-classifier $M_C$ is given a vector of $(0,1,1)$, its prediction will always be 0, because $|(0,1,1), 0|$ is the majority in training the meta-classifier.

Stacking gives a final prediction according to the value of $c$. We call each $c$ a type. A data item is mapped into exactly one type. In each type, we call one classification the majority label if its count is the majority. Stacking chooses the majority label. For data items of type $c$, stacking will have accuracy of $\alpha$ as defined previously.

For data of each type, there is always an $\epsilon$ portion that are labelled wrong. We assume that the training data and testing data are of the same distribution; this means that the $\epsilon$ portion will always be misclassified. We call the $\epsilon$ portion data conflicts. Conflicts were first discussed in [8, 14]. In this paper, we concentrate on measuring performance of a stacked classifier system based on conflicts. The selection of majority class for each conflict type is the machine learning bias of stacking. We have used $\alpha$ and $\epsilon$ to quantitatively measure the accuracy of each conflict types in previous discussion, we can further consider the effects of conflicts on the overall performance.

Considering the effects of conflicts on a complete training set, we define conflict-based accuracy estimate $A$ and conflict-based error estimate $E$ as:

$$A = \frac{\sum_{c=0}^{(1,1,\ldots,1)} \max(|c, 1|, |c, 0|)}{m}, \quad E = 1 - A$$

For brevity, we call conflict-based accuracy estimate as CB-accuracy estimate and conflict-based error estimate as CB-error estimate. $A$ is the portion of all training data with majority labels. This portion is labelled correctly by stacking. On the contrary, $E$ is the portion of the training data with minority labels and they are labelled incorrectly. $A$ and $E$ are predictions for the stacked classifier system’s performance in testing data. $A$ and $E$ are not training accuracy and training error because when generating the meta-level data, the base classifiers were trained from $T_k$ and tested against a disjoint $V_k$. $A$ and $E$ may not be accurate when conflict ratio $\epsilon$ of some conflict types are close to 0.5. In these cases, the chance that the majority label of these conflict types is not the best choice (or becomes the minority label for testing data) is very high.

To see the effects of conflicts on overall performance, consider Figure 1. In the left table, the accuracy $\alpha$ on $(0,0,1)$ to $(1,1,0)$ are very low: 0.537 to 0.808. There are in total 5647 conflicts out of 29491 training data items. So the CB-accuracy estimate is $A = 23844/29491 = 0.809$ and the CB-error rate estimate is $E = 5647/29491 = 0.191$. The second table is even worse. The accuracy of types $(0,0,1)$ to $(1,1,0)$ are from 0.526 to 0.768. The type $(1,0,0)$ is very tricky, since $\epsilon$ is nearly 0.50. The chance that the majority prediction 0 will be wrong for the testing data is high.

### 2.2 Measuring Performance of Stacking

We first consider different performance measurements. We propose to use CB-accuracy estimate to predict stacking’s performance. We then discuss some popular metrics and show by example that it is relatively hard to use them to predict the overall accuracy.

$\sum_{c=0}^{(1,1,\ldots,1)}$ iterates from $(0,0,\ldots,0)$ to $(1,1,\ldots,1)$
Recent work on stacking has concentrated on improving accuracy. The improved accuracy estimate we call CB-accuracy. For brevity, we meta-level training data by counting how many predictions our classifiers are fixed. However, it is problematic to use these metrics to compare different structures, since these measures are defined over the average performance accuracy of all base classifiers. A stacked system with inaccurate base classifiers may have a very high accuracy improvement, but may not necessarily have a good overall performance accuracy. To compare different structures, choose classifiers for stacking and prune classifiers, we claim that using accuracy is more appropriate. Assuming the training and testing data have the same distribution, we hypothesize CB-accuracy estimate, derived from conflicts, is a good and direct estimation of accuracy.

We also define conflict-based accuracy improvement estimate to project accuracy improvement. It is defined as \( \frac{(A-B)}{B} \). \( B \) is the average of the base classifier accuracy estimate, which is easily calculated from the meta-level training data by counting how many predictions one classifier has correct labels. For brevity, we call conflict-based accuracy improvement estimate as CB-accuracy improvement estimate.

**Metrics:** Recent work on stacking has concentrated effort on using coverage [2], diversity [2, 4], specialty [4] and correlated error [1] to choose classifiers for combining and to explain how stacking increases accuracy. For a detailed description and formal definition of these metrics, please refer the cited papers. These metrics, except for coverage, are indirect measurements of stacking. On the other hand, conflicts explains what actually contributes to accuracy improvement and what prevents it. Conflicts is not a metric. It is the cause of poor performance of stacking. CB-accuracy estimate \( A \), derived from conflicts, can be used to directly predict the accuracy of a stacked classifier system.

It has been reported that there is either an increasing or a decreasing trend of accuracy improvement (not overall accuracy) when these metrics increase. As we shall see, it is relatively hard to use these metrics to estimate how well the overall stacked system will be.

**Coverage**, introduced by Brodley and Lane [2], measures the fraction of instances for which at least one of the base classifiers produces the correct predictions. Coverage actually measures 2 extreme cases of conflicts, \((0,0,\ldots,0),1\) and \((1,1,\ldots,1),0\). In terms of conflicts, we can define coverage as: \( \text{coverage} = \frac{1 - \left(\frac{m-1}{m}\right)^{\#\text{of conflicts}}}{m} \). But conflict is more general than coverage. It is reported that high coverage will increase stacking’s accuracy improvement [2, 4]. This is natural since high coverage reduces the occurrences of the two extreme cases of conflicts.

**Diversity** [2, 4] measures how different the base classifiers are, based on their predictions. When the value of diversity grows, the predictions from the base classifiers are more evenly distributed and, therefore, more diverse. It has been observed that accuracy improvement increases with diversity. But it is not necessarily true that high diversity will increase overall accuracy. From the example in Figure 2, we see that when diversity increases, overall accuracy \( \sigma \) can either increase or decrease. The numbers in Figure 2 are calculated from the formula of each metric.

**Correlated Error**, introduced by Ali and Pazzani [1], measures the fraction of instances for which a pair of base classifiers make the same incorrect prediction. It has been observed that there is a decreasing trend in

<table>
<thead>
<tr>
<th>BOOLEANS Meta Level Data</th>
<th>BOOLEANS/8 Meta Level Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c )</td>
<td>(</td>
</tr>
<tr>
<td>(0,0,0)</td>
<td>523</td>
</tr>
<tr>
<td>(0,0,1)</td>
<td>972</td>
</tr>
<tr>
<td>(0,1,0)</td>
<td>754</td>
</tr>
<tr>
<td>(1,0,0)</td>
<td>1247</td>
</tr>
<tr>
<td>(1,0,1)</td>
<td>886</td>
</tr>
<tr>
<td>(1,1,0)</td>
<td>1318</td>
</tr>
<tr>
<td>(1,1,1)</td>
<td>2210</td>
</tr>
<tr>
<td>CB-accuracy estimate ( A = 0.809 )</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BOOLEANS/8 Meta Level Data</th>
<th>BOOLEANS/8 Meta Level Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>( c )</td>
<td>(</td>
</tr>
<tr>
<td>(0,0,0)</td>
<td>367</td>
</tr>
<tr>
<td>(0,0,1)</td>
<td>956</td>
</tr>
<tr>
<td>(0,1,0)</td>
<td>0</td>
</tr>
<tr>
<td>(1,0,0)</td>
<td>0</td>
</tr>
<tr>
<td>(1,0,1)</td>
<td>1249</td>
</tr>
<tr>
<td>(1,1,0)</td>
<td>1419</td>
</tr>
<tr>
<td>(1,1,1)</td>
<td>0</td>
</tr>
<tr>
<td>CB-accuracy estimate ( A = 0.883 )</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: Sample of Conflicts with RIPPER, CART and ID3 as base classifiers.
3 Experiments and Results

We wish to determine whether or not the proposed CB-accuracy estimate accurately estimates the overall performance $\sigma$ and if CB-accuracy improvement estimate is effective towards projecting the true accuracy improvement. We compare them with those previously defined.

**Experimental Set-up:** In our experiments, four data sets were used, two artificial boolean data sets and two real world credit card transaction data sets. We generated two artificial data sets of 15 boolean variables. For BOOLEAN5678, the data item is true if 5, 6, 7 or 8 variables out of 15 are 1 otherwise it is false. Since there are 15 variables, there are $2^{15} = 32768$ data items in total. The percentage of data items with label true are about 64%. Another similar artificial data set, BOOLEAN45 was generated in a similar way: a data item is true if 4 or 5 variables out of 15 are 1. The percentage of data with label true is about 13.3%. We used 10-fold CV in our test and did not replicate any data items, so the training and test sets were disjoint. These two datasets are noise-free and of significant size, it is relatively easy for us to analyze results under such conditions.

We also ran experiments on two real world data sets, First Union Credit Card Transaction Data and Chase Credit Card Transaction Data. The target classification of the data is either legitimate or fraudulent. For a description of the data schema, refer to [14]. From each bank, we obtained 0.5 million data spanning a whole year. The First Union data was not uniformly sampled for each month. The percentage of fraud ranges from 4% to over 20% over each month and the size of data for each month varies. In our experiment, we removed all fields that are not available at authorization time and then culled all transactions into one large data set. We partitioned the data into 10 folds, used one fold for training and another fold for testing for a total of 5 runs. Each fold is disjoint. The Chase Credit Card data was more uniformly sampled than the First Union data. The fraud percentage of each month ranges from 17% to 23%. The data set size of each month varies from 28k to 50k. Although the training and testing data are not of the same distribution, we think that this data set is more appropriate to test these measures in a real world situation. We used data of one month for training and data of 2 months later for testing. (In a real world content, there is a one month billing cycle and a one month investigation to ultimately determine if a transaction is fraudulent.) Since our data set is 12 months, only 10 experiments are feasible.

We used Ripper [6], CART, ID3 and C4.5 [3] as the base learners. We combined 2 and 3 out of the 4 base classifiers trained by these programs and formed 10 different stacked systems. 2-fold CV was used to generate meta-level training data. We did not use any meta-learner here. We have shown that the meta-level data is actually easy to learn. In place of the meta-
classifier, we used a rote table to record all the conflict
types and their majority labels. The use of a rote ta-
ble has exactly the same effect as an un-pruned full
decision tree. Each conflict type is a leaf of such a
tree. However, a rote table may be different from a
pruned/generalized classifier. In the Section 4, we will
discuss this issue.

For each combination of base classifiers, we calculated
the value of all the metrics and the corresponding over-
all accuracy $\sigma$ and accuracy improvement. We plot all
the original data points in the figures that follow. We
didn’t use their average in order to show the full spread
of the points. We used polynomials to fit the data
points to uncover any trend. We used the Marquardt-
Levenberg algorithm [12] (a non-linear least square fit-
ing procedure, available in the GNUFIT package [9])
for this purpose. The sum of residuals usually stabi-

dized at a quadratic fit. Some plots have very scattered
data points and the residual of fitting is very large.
This implies that there is hardly any trend. In order
to compare results from different tests, we normalized
them into the range $[0,1]$. 

**Experimental Results:** The results on the different
data sets are displayed in Figure 4 and Figure 5. In
Figure 4, we display the change of overall accuracy
with the increase of four metrics in all four data sets.
In Figure 5, we show the results on overall accuracy
improvement. In each plot, we display 100 data points
(First Union plots have 50 points), both linear and
quadratic fit if there is any trend. Each figure contains
4 columns, each displaying results of four metrics on
one data set. Starting from the left column, there are
BOOLEAN5678, Chase, First Union and BOOLEAN45.
Each row is the result of one metric on all four data
sets.

We first observe that the result on BOOLEAN45 (the
last column of both Figure 4 and Figure 5) is an out-
lier. Its trends on all metrics, if any, for both overall
accuracy and accuracy improvement tests differ from
the results on the other three data sets. All the re-
results on the accuracy improvement measure don’t con-
form to previous findings either. The data points of
many plots are scattered. We have taken a look at the
meta-level testing data. The reason is that many con-
flict types (such as type (1,0,0) depicted in Figure 1)
have a conflict ratio $\epsilon$ close to 0.5. The majority la-
bel of these types became minority labels for the test-
ing data. Due to this reason, the measures based on
meta-level training data were no longer valid for the
testing data. Therefore, we will focus our discussion
on the other three domains (which are in the first three
columns of Figure 4 and 5).

We compare the different metrics towards predicting
overall accuracy $\sigma$. **CB-accuracy estimate** is the best
performer. There is a very clear increasing trend
when the **CB-accuracy estimate** increases. This grow-
ing trend is consistent for BOOLEAN5678. Chase and
First Union data sets. The slope of the linear fit is al-
most 1.0. The residuals of fitting are relatively small.
The other metrics don’t show any consistent trend or
any trend at all for the different data sets. The data
points of these metrics are more scattered than the
plot of **CB-accuracy estimate**.

Next we compare the different metrics towards pre-
dicting accuracy improvement. Both **CB-accuracy im-
provement estimate** and **correlated error** are the best
performers. There is a clear increasing trend when the
**CB-accuracy improvement estimate** increases, while
there is a clear and decreasing trend with the increase of
**correlated error**. The linear fit slopes (absolute
value) for the **CB-accuracy improvement estimate** and
the **correlated error** are identical, which means they
have equal predictive value. For diversity, there is
an increasing trend with the increase of diversity in
2 cases. In the 4 sets of experiments, we see a decreas-
ing trend of accuracy improvement when the **specialty**
metric increases. We anticipate the accuracy improve-
ment to increase with the **specialty** metric. On close
inspection, the metric is flawed. For example, if a clas-
sifier always predicts one class, it has a high **specialty**
value. An improved **specialty**-based metric is proposed
in [15].

4 **Discussion and Conclusion**

The **CB-accuracy estimate** and **CB-accuracy improve-
ment estimate** metrics are effective in predicting the
performance of a stacked classifier, when the conflict
ratio $\epsilon$ of each type are different than 0.5. But they
are inaccurate when the ratios are close to 0.5. The
reason is that we didn’t take conflict ratio $\epsilon$ into ac-
count when defining $A$. For a stacked system in the
presence of severe conflicts, we propose **confidence** or
**CB-accuracy estimate range**. We define the **confidence
ratio** for a conflict type to be, $confidence_{(c_1,c_2,...,c_t)} = f(\alpha_{(c_1,c_2,...,c_t)})$. $f(x)$ is a function that maps $\alpha$ to
the range of $[0,1]$ with $f(0.5) = 0$ and $f(1) = 1$. Our
empirical studies show that when $\alpha$ is more than 0.6,
it is very unlikely that such a conflict type will be
flipped to its minority label during testing. There-
fore, a non-linear function with decreasing derivatives
may be preferred. We can use a sigmoid-like function,
\[ y(x) = \frac{1}{1 + e^{-\beta x}}, \quad f(x) = \frac{y(x) - y(0.5)}{y(1) - y(0.5)}. \] (t adjusts the derivatives of \( f(x) \).) We define the confidence of \( A \) to be the weighted sum of the confidence of each conflict type. The weight of each conflict is its size divided by the size of the training set. A stacked system on the same data set with both a higher \( A \) and a higher confidence ratio is very likely to have higher testing accuracy than one with both lower \( A \) and confidence ratio. An alternative approach is to define \( A \) as a range. For each conflict type whose \( \epsilon \geq \tau \), we use its minority class frequency to estimate the lower bound of overall accuracy. The upper bound is the original definitions of \( A \). In practice, we can set \( \tau \) to be 0.4. If the lower bound of one system is bigger than the upper bound of another, it is likely that the previous one will have higher overall accuracy.

The rote table approach is equivalent to an un-pruned full tree, but it may not be the same as a pruned/generalized classifier. For example, if two leaves \((1,1,1)\) (with majority label 1) and \((1,1,0)\) (with majority label 0) are pruned, then their parent node \((1,1,?)\) is a new leaf and we assume that 1 is its majority label. This means for \((1,1,0)\) pattern, its minority label 1 will be used in testing. For a pruned meta-classifier, we can still estimate its overall accuracy and accuracy improvement. We enumerate all the patterns of meta-level training data, \((0,0,...,0)\) to \((1,1,...,1)\), and send them to the meta-classifier. The predictions by the meta-learner can be used as the ‘majority label’ to calculate the accuracy of each conflict type \( \alpha \) and thus \( A \).

When designing a meta-learning system, one must choose from a set of available classifiers those whose combination will derive the best overall stacked classifier. Various metrics have been proposed as a means for choosing the best classifiers. The presence of conflicts in a stacked generalizer is an important factor affecting its accuracy. We have derived the CB-accuracy estimate and the CB-accuracy improvement estimate metrics from conflicts to measure and compare the performance of stacked systems. From our analysis and empirical studies, CB-accuracy estimate is the most accurate measure of overall testing performance and CB-accuracy improvement estimate is as good as correlated error and better than all the other metrics previously proposed.

References


[10] S. Hashem Optimal Linear Combination of Neural Networks. Phd Thesis, Purdue University, School of Industrial Engineering, Lafayette, IN, 1993


Figure 4: Different Metrics to Predict Overall Accuracy $\sigma$
Figure 5: Different Metrics to Predict Accuracy Improvement