Optimal Data Partition for Semi-Automated Labeling

Daniel Lopresti  
Lehigh University  
Bethlehem, PA, 18015 USA  
lopresti@cse.lehigh.edu

George Nagy  
Rensselaer Polytechnic Institute,  
Troy, NY, 12180 USA  
nagy@ecse.rpi.edu

Abstract

In a pattern recognition sequence consisting of alternating steps of interactive labeling, classifier training, and automated labeling (e.g., CAVIAR systems), the choice of sample size at each step affects the overall amount of human interaction necessary to label all the samples correctly. The appropriate splits depend on the error rate of the classifier as a function of the size of the training set and, perhaps surprisingly, are independent of the relative costs of interactive correction and confirmation. We model such a system and report the sequence of optimal data partitions for a representative range of parameters.

1. Introduction

The following paradigm is the common denominator of some Computer Aided Visual Interactive Classification (CAVIAR) systems developed for diverse applications (Fig 1):

1. A sequence of $N_{\text{train}}$ patterns is displayed on a graphic user interface and labeled by an operator.
2. A conventional classifier is trained on features extracted from the labeled patterns.
3. The classifier is run on a (usually larger) set of unlabeled patterns.
4. The newly labeled patterns are displayed for approval or correction by the operator.
5. Optionally, the corrected or approved labels and patterns are added to previous training set to retrain the classifier for the next batch of unknown patterns.

The cost of entering the label of a training pattern or of correcting a misclassified test pattern is higher than that of confirming a correct label (which typically requires only a single mouse click). It has been our experience that corrections take 5-6 times longer than confirmation for flowers or web table cells. The relative costs can be determined from a short interactive session. A ratio of 3 is appropriate for digit recognition due to the smaller number of classes.

We consider computational costs of interaction plus classification essentially constant and endeavor to minimize operator time. Multistage CAVIAR systems incur lower human interaction costs than the customary alternative of training on a small fixed partition and correcting every error at the end.

Given a fixed number of patterns to be labeled with a CAVIAR system, it is necessary to decide the appropriate balance between the size of the training set and the size of the interactively corrected test set. A large training set decreases the number of errors on the test set, but labeling it is expensive. A small training set may results in too many costly-to-correct errors on the test set.

We propose a model for determining the optimal size of the training set given a fixed number of patterns to be classified. We consider both two-stage (Steps 1-4 above) and $n$-stage (Steps 1-5) CAVIAR systems.

Figure 1. CAVIAR dataflow
selected automatically, has goals similar to CAVIAR's medical form entry [14].

complete verification is indispensable in financial and with residual errors [13]. An application where labeled dataset, would be timely.

on the human time required to produce an accurately [12]. A comparison between these approaches, based error-sensitive applications, discuss what is to be done

web tables [6]. Similar systems are being developed
goals was directed at error-sensitive applications, discuss what is to be done

practical pattern-recognition problems. " [10]. It was, and classification systems (IPACS) as a general approach to emergence of graphics-oriented interactive pattern analysis

and the samples are added to the previous training set errors at each stage $i$

3. A model for determining the best splits

We model a multistage labeling process where the errors at each stage $i$ are corrected by a human operator and the samples are added to the previous training set to retrain the classifier. The classifier trained on $N_{i-1}$ samples makes $M_i$ errors when it classifies the next $N_i$ samples ($i = 1, 2, ..., n$). These errors are corrected and the rest of the labels are approved.

The average duration of inspecting and confirming a correct label is taken as the unit of time or cost: each confirmation has unit cost. Either entering a new label or correcting a wrong label is assumed to take $r$ units of time and have cost $r$. The number of errors $M_i$ at the $i^{th}$ stage is $N_i f(N_{train} : \beta)$, where $\beta$ is a scalar or vector parameter of the function that specifies the dependence of the error rate on the training set size. The time required to confirm all the correct labels at the $i^{th}$ stage is $(N_i - M_i)$, and the time required to correct the errors is $r \times M_i$.

All $N_0$ objects of the initial training set must be labeled, therefore $M_0 = N_0$. The operator time at stage $i$ is $T_i$. More formally:

$M_0 = N_0, \ n > 0, f(0; \sigma) = 1, \ r \geq 1,$

and $N_{total} = \sum_{k=0}^{n} N_k$. \[M_i = N_i f \left( \sum_{k=1}^{i} N_{k-1} ; \beta \right)\]

$T_i = (N_i - M_i) + r M_i = N_i + (r - 1) M_i$

$= N_i \left( 1 + (r - 1) f \left( \sum_{k=1}^{i} N_{k-1} ; \beta \right) \right)$

$T_{total} = \sum_{h=0}^{n} T_h$

$= r N_0 + \sum_{h=1}^{n} N_h \left( 1 + (r - 1) f \left( \sum_{k=1}^{h} N_{k-1} ; \beta \right) \right)$

For $n = 2$ it is easy to show that the derivative of $(T_0 + T_1)$ with respect to $N_0$ is proportional to $(r-1)$. Therefore the optimum split is independent of $r$. This holds also for $n > 2$. The total cost and the error function $e^{\beta N_{total}}$, with $n = 1$, $N_{total} = 1000$, $r = 3$, and $\beta = 0.0046$, are graphed in Figure 2 as a function of $N_0$. The optimal $N_0$ is 311, so $N_1$ is 689. Splitting at $N_0 = 500$ would increase interaction time by 8%.

### Table I. Some CAVIAR systems

<table>
<thead>
<tr>
<th>Objects</th>
<th>Sample Size</th>
<th>Label</th>
<th>Year</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Printed documents</td>
<td>30,000</td>
<td>ASCII code</td>
<td>1971</td>
<td>[1]</td>
</tr>
<tr>
<td>Flowers</td>
<td>612</td>
<td>Species</td>
<td>2003</td>
<td>[3,4]</td>
</tr>
<tr>
<td>Calligraphy</td>
<td>13,351</td>
<td>GB-2312 code</td>
<td>2011</td>
<td>[5]</td>
</tr>
<tr>
<td>Faces</td>
<td>500</td>
<td>Name of person</td>
<td>ongoing</td>
<td>[7]</td>
</tr>
<tr>
<td>Election ballots</td>
<td>13,315</td>
<td>Candidate</td>
<td>&quot;&quot;</td>
<td>[8]</td>
</tr>
<tr>
<td>Cervigrams</td>
<td>100</td>
<td>Abnormal tissues</td>
<td>&quot;&quot;</td>
<td>[9]</td>
</tr>
</tbody>
</table>
We resorted to exhaustive search to find the optimum number of samples before switching from automatic processing to interaction for \( n = 3 \). We set the parameter \( \beta \) to reach an error rate of either \( E_{1000} = 1\% \) or \( E_{1000} = 5\% \) after training on 1000 patterns.

Published experiments indicate that the effect of the size of the training set on the error rate varies considerably depending on the data, features, and classifier. We have therefore calculated the splits for super-linear, linear, and supra-linear initial fall-off of the error rate with the size of the training set. These functions, graphed in Figure 3, are:

\[
\begin{align*}
 f(N_{\text{train}}; \beta) & = e^{-\beta N_{\text{train}}} , \\
g(N_{\text{train}}; \beta) & = 1 - \beta N_{\text{train}}, \\
h(N_{\text{train}}; \beta) & = 1 - \beta (N_{\text{train}})^2 .
\end{align*}
\]

4. Discussion

The above analysis provides guidance to optimal sizing of data sets when human labeling alternates with automated classifier training and labeling. Practical application requires an approximation of the error function. Fortunately, only an approximation of the early, high-error segment is needed, and the approximation can be improved from the results on the early stages. The relative cost of corrections vs. confirmations does not affect the experimental protocol, only the overall gain.

The gain is a slowly rising function of the number of stages, which in practice is limited by the logistics of retraining the classifier and scheduling interaction. Ideally the classifier would be retrained after every verified or corrected label.

Some aspects of the proposed protocol that deserve further consideration are:

4.1 Domain of application

The current analysis is limited to classifiers that treat each pattern as an independent entity. It excludes classifiers that use language, scene, or style context because such contexts may extend beyond batch boundaries.

In many applications, however, only-short range context is exploited. If this range is much smaller than the number of samples in the split sets, then the model may still apply.

4.2 Fundamental independence assumption

We have implicitly assumed that the classification process is stationary and that successive decisions are statistically independent. The best way to assure that these assumptions apply to any particular set of data is to reorder the entire data set with a pseudo-random permutation. That guarantees that each training set is representative of the corresponding test set, and that

<table>
<thead>
<tr>
<th>error fn</th>
<th>( E_{1000} )</th>
<th>( r )</th>
<th>( N_0 )</th>
<th>( N_1 )</th>
<th>( N_2 )</th>
<th>( S(%) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>h(*)</td>
<td>0.01</td>
<td>3</td>
<td>177</td>
<td>274</td>
<td>549</td>
<td>73</td>
</tr>
<tr>
<td>f(*)</td>
<td>0.01</td>
<td>3</td>
<td>333</td>
<td>333</td>
<td>334</td>
<td>25</td>
</tr>
<tr>
<td>g(*)</td>
<td>0.01</td>
<td>3</td>
<td>645</td>
<td>237</td>
<td>118</td>
<td>11</td>
</tr>
<tr>
<td>h(*)</td>
<td>0.05</td>
<td>3</td>
<td>214</td>
<td>300</td>
<td>486</td>
<td>54</td>
</tr>
<tr>
<td>f(*)</td>
<td>0.05</td>
<td>3</td>
<td>333</td>
<td>333</td>
<td>334</td>
<td>23</td>
</tr>
<tr>
<td>g(*)</td>
<td>0.01</td>
<td>5</td>
<td>645</td>
<td>237</td>
<td>118</td>
<td>14</td>
</tr>
<tr>
<td>h(*)</td>
<td>0.05</td>
<td>3</td>
<td>418</td>
<td>836</td>
<td>3746</td>
<td>131</td>
</tr>
</tbody>
</table>

The values of \( N_0, N_1, \) and \( N_2 \) required to minimize \( T_{\text{total}} \) are shown for several situations in Table II. \( N_{\text{total}} = 1000 \) in every simulation except the last one, where \( N_{\text{total}} = 5000 \). The human time saved by using the classifier is \( S = 1 - T_{\text{total}} / T_{\text{max}} = 1 - T_{\text{total}} / r N_{\text{total}} \).

If correction and verification cost the same \( (r = 1) \), any choice would give the same final cost \( r N_{\text{total}} \). The slower the error rate decreases initially, the more samples should be used in the earlier stages. Poor initial generalization leads to higher final cost and less savings. A linear error function splits the data uniformly. The gain over direct entry increases with the ratio of corrections to verification and with the total number of samples to be classified. With error function \( f \), increasing the total sample size from 1000 to 5000 almost doubles the time saved.
successive classifier decisions are statistically independent. Although on any particular sequence the error rate may not decrease monotonically, randomization ensures that it does so on average.

In contrast to experiments designed to demonstrate generalization, the grain of randomization should be as small as possible in order to minimize errors. In document image analysis, for example, we would randomly split each document into individual glyphs or words. In table processing or writer recognition, each source should be split into sequences of samples.

4.3 Statistical fluctuation of the error rate
The optimality of the splits applies of course only to statistical expectations rather than individual runs. Fortunately the optimal splits seldom push the classifier to its lowest error rate on the available sample size. For instance, the error rate at the optimal \(N_0\) in Fig. 2 is 24%, in contrast to 1% at \(N_0 = 1000\). The statistical fluctuation of the higher error rates resulting from medium-sized training sets tends to be much less than those generally reported for cross-validation using the largest possible sets of training samples.

4.4 Cost of simulation
For a three-stage system for 1000 samples, the simulation runs on a garden-variety laptop in less than a tenth of a second. It is, however, proportional to \(N_{\text{total}}\) for the exact number of samples in each split, if we have a million samples we can scale the error function to \(\frac{N_{\text{total}}}{n}\), where \(n\) is the number of stages, because it searches over the cross product of every stage except the last (the splits must add up to \(N_{\text{total}}\)).

There are several means of reducing run time. If the error function is monotonically decreasing, the total time has only a single minimum. Therefore any gradient-descent method will find it quickly.

Furthermore, the only critical input to the simulation is the error function. Instead of searching for the exact number of samples in each split, if we have a million samples we can scale the error function to 1000 or 10,000 samples and search for splits accurate to the nearest 1000 or nearest 100 samples. Of course interaction on such large data sets raises other problems as well.

4.5 Selective inspection
It is appealing to consider inspecting and correcting or verifying only the labels of patterns flagged by the classifier (as in [1]). This procedure, however, fails to guarantee a final error rate of zero (i.e., labels considered correct by a human operator.).

If each stage is not completely verified and only rejects or patterns flagged “critical” by the classifier are labeled by the operator, then some cost must be assigned to undetected errors relative to rejects. The simulation can be readily modified to accommodate any available error/reject or ROC curve. This would be appropriate in applications where the cost of exhaustive verification is prohibitive.

Acknowledgment
Daniel Lopresti receives support from a DARPA IPTO grant administered by Raytheon BBN Technologies.

References