Impact of Classifier Prototyping on Classification Systems Optimization

Abstract

This paper compares two wrapper approaches to optimize classification systems. The traditional direct approach uses the actual classifier on the optimization process, whereas the second prototypes the target classifier with a computationally inexpensive classifier. Our results indicate that using the actual classifier yields computationally faster classifiers. However, the accuracy in both approaches is comparable and the required optimization time may be prohibitive in certain contexts, justifying the use of a prototyping approach.

1. Introduction

Image-based pattern recognition requires that pixel information be first transformed into an abstract representation (a feature vector) suitable for recognition with classifiers, a process known as *feature extraction*. A relevant classification problem is the *intelligent character recognition* (ICR), most specifically the offline recognition of isolated handwritten symbols on documents. A methodology to extract features must select the spatial location to apply transformations on the image [5]. The choice takes into account the *domain context*, the type of symbols to classify, and the *domain knowledge*, what was previously done in similar problems. The process is usually performed by a human expert in a trial-and-error process. Associated to this burden, one classification system is tailored to a specific domain. Figure 1 details the difficulties faced by changing the handwriting style. Thus, the same classification system can not be used on another problem with the same reliability, unless the classification system is properly adapted to the new context.

This context mandates a semi-automated approach that uses the expert's domain knowledge to optimize the classification system. To minimize the human intervention in defining and adapting classification systems, this problem is modeled as an optimization problem, using the expert's domain knowledge and information from the domain context – actual images. This optimization problem optimizes classification systems as in Fig. 2. The *Intelligent Feature Extraction* (IFE) methodology extracts feature sets in RS_{IFE}, where the best performing classifier SI is selected. The IFE is further detailed in Section 2.

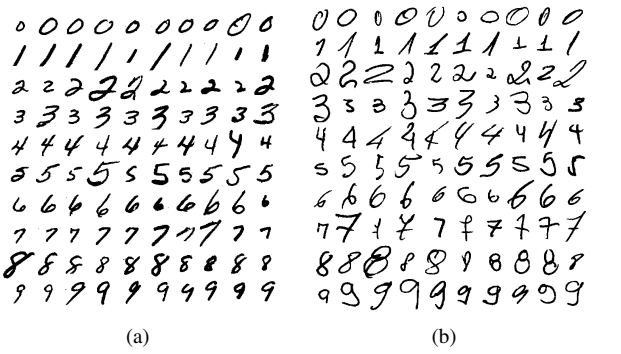

Figure 1. Handwritten digits: (a) NIST SD-19 and (b) Brazilian checks.

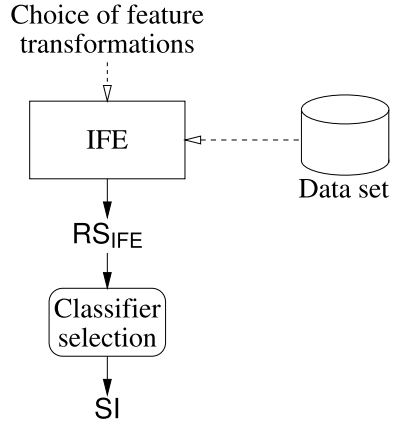

Figure 2. Classification system optimization approach.

The optimization algorithm employed also plays an important role, not only related to obtained feature set quality, but to the required processing time as well. Experi-

ments were performed in [author reference] using *multi-objective genetic algorithms* (MOGAs)[1] and in [author reference] with the *Record-to-Record Travel* (RRT) algorithm [6]. Both algorithms successfully optimized the problem, producing solutions that outperformed the human expert approach. We used a prototyping wrapper approach to optimize a *multi-layer perceptron* (MLP) neural network, owing to the long processing time required to train the MLP classifiers. The prototyping consisted in using the *projection distance* (PD) classifier [2] to obtain RS_{IFE} and select SI, which was then trained using an MLP classifier. Our hypothesis is that SI could be improved if we used the target classifier (MLP) in the optimization process, instead of prototyping feature sets with a different classifier.

The new contribution is to investigate this hypothesis of wether prototyping impacts heavily the solutions obtained, by using the RRT algorithm to optimize the IFE directly with MLP classifiers. The RRT algorithm is detailed in 3 and the expetriments performed are indicated in Section 4. Results obtained are detailed in Section 5 and discussed in Section 6.

2. Intelligent Feature Extraction

The goal of IFE is to help the human expert define representations in the context of isolated handwritten symbols, using a wrapper approach to optimize solutions. IFE models handwritten symbols as features extracted from specific *foci* of attention on images using *zoning*. Two operators are used to generate representations with IFE: a *zoning operator* to define *foci* of attention over images, and a *feature extraction* operator to apply transformations in zones. The choice of transformations for the feature extraction operator constitutes the human expert domain knowledge. The domain context is introduced as actual observations in the *optimization* data set used to evaluate and compare solutions. Hence, the zoning operator is optimized by the IFE to the domain context and domain knowledge.

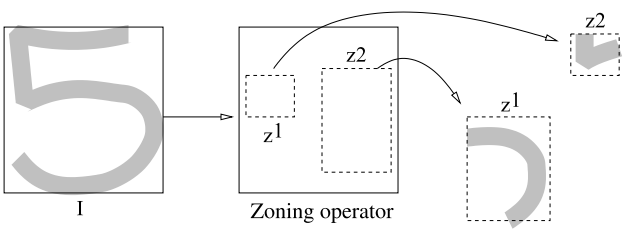

Figure 3. IFE structure.

The IFE structure is illustrated in Fig. 3. The zoning operator defines the zoning strategy $Z = \{z^1, \ldots, z^n\}$, where $z^i, 1 \leq i \leq n$ is a zone in the image I and n the total number of zones. Pixels inside the zones in Z are transformed by the feature extraction operator in the representation $F = \{f^1, \ldots, f^n\}$, where $f^i, 1 \leq i \leq n$ is the partial feature vector extracted from z^i. At the end of the optimization process, the optimization algorithm has explored the representation set $RS_{IFE} = \{F^1, \ldots, F^p\}$. For MOGAs, RS_{IFE} is the optimal set at the last generation, and for the RRT algorithm, it is the explored solution set. Whereas the later may seem larger at first, their sizes are actually comparable.

The result set RS_{IFE} is used to train the classifier set $K = \{K^1, \ldots, K^p\}$, where K^i is the classifier trained with representation F^i. The first hypothesis is to select the most accurate classifier $SI, SI \in K$ for a single classifier system. The second hypothesis, which is not subject of this work, is to use K to optimize an an *ensemble of classifiers* (EoC) [4, 3] for higher accuracy. The remainder of this section discusses the IFE operators chosen for experimentation with isolated handwritten digits and the candidate solution evaluation.

2.1. Zoning Operator

To compare performance to the traditional human aproach, a *baseline* representation with a high degree of accuracy on handwritten digits with a *multi-layer Perceptron* (MLP) classifier [author reference] is considered. This baseline representation was defined on a traditional trial and error basis. Its zoning strategy, detailed in Fig. 4.b, is defined as a set of three image dividers, producing 6 zones. The *divider zoning operator* expands the baseline zoning concept into a set of 5 horizontal and 5 vertical dividers that can be either *active* or *inactive*, producing zoning strategies with 1 to 36 zones. Figure 4.a details the operator template, encoded by a 10-bit binary string. Each bit is associated with a divider's state (1 for active, 0 for inactive).

2.2. Feature Extraction Operator

The classification system in [author reference] used and detailed a mixture of concavities, contour directions and black pixel surface transformations, extracting 22 features per zone (13 for concavities, 8 for contour directions and 1 for surface). To allow a direct comparison between IFE and the baseline representation, the same feature transformations (the domain knowledge) are used to assess the IFE.

2.3. Candidate Solution Evaluation

Candidate solutions are evaluated with respect to their classification accuracy. Thus, the objective is to minimize

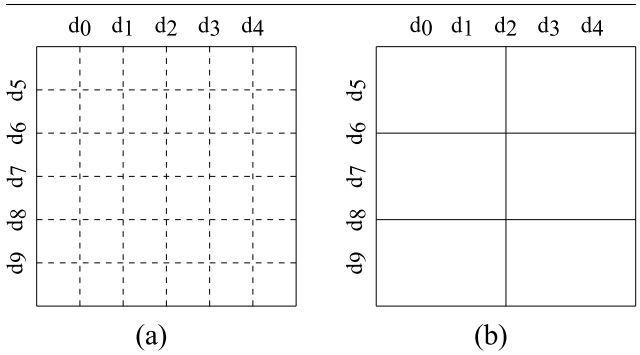

Figure 4. Divider zoning operator (a). The baseline representation in (b) is obtained by setting only d_2, d_6 and d_8 as active.

the classification error rate on the *optimization* data set (the domain context). To compare wrapper methods, candidate solutions are optimized with both the PD and MLP classifiers. The PD classifier is used as an inexpensive prototyping classifier, and obtained feature sets are used to train the SI MLP classifier. The optimization process is also performed directly with the MLP classifier. The goal is to compare classifier performance when using prototyped feature sets against feature sets obtained directly with the target classifier.

3. Optimization Algorithm

The *Record-to-Record Travel* (RRT) algorithm [6], is an annealing based heuristic. It is said to be a local search algorithm, searching for new solutions in the vicinity of the current solution. The RRT algorithm improves an initial solution i by searching in its neighborhood for better solutions based on their evaluation (classification error rate). The RRT algorithm, detailed in Algorithm 1, produces after a number of iterations the record solution r. The algorithm is similar to a hill climbing approach, but avoids local optimum solutions by allowing the search towards non-optimal solutions with a fixed deviation D. Earlier experiments indicated that the RRT algorithm over-fitted solutions during the optimization process. The global validation strategy discussed in [author reference] is used to avoid this effect, and Algorithm 1 includes support for this strategy.

Given the initial solution i, the algorithm will copy it to the record solution r and store its evaluation value in $RECORD$. It also copies i as the current solution p. Next it will repeat the following process during a number of iterations, until the current solution is worse than the record solution plus the allowed deviation. First it will find the set P, solutions neighbor to p, and select the best neighbor $p', p' \in P$. To avoid cyclic optimization, solutions already evaluated are not considered for p'. If evaluating p' yields results within the allowed deviation, it is copied as p for the next iteration. Solution p' replaces the record solution r only if it yields better results. If p' is worse than the allowed deviation, the optimization process stops. The explored solution set S is responsible to store solutions tested by the RRT algorithm for the global validation strategy. At each iteration, the algorithm inserts into S the solutions in the neighbor set P. At the end of the optimization process, solutions in S are validated and the most accurate solution is selected. For the IFE process, S is the result set RS_{IFE} used to create the classifier set K.

Data: Initial solution i
Data: Deviation D
Result: Record solution r
Result: Explored solution set S
$r = i$;
$RECORD = eval(r)$;
$p = i$;
$S = \emptyset$;
repeat
  Create the solution set P, neighbor to p;
  $S = S \cup P$;
  Select the best solution $p' \in P$ such as that p' has not yet been evaluated;
  if $eval(p') < RECORD + RECORD \times D$ **then**
    $p = p'$;
    if $eval(p') < RECORD$ **then**
      $RECORD = eval(p')$;
      $r = p'$;
    end
  end
until $eval(p) <= RECORD + RECORD \times D$;

Algorithm 1: Modified record to record travel (RRT) algorithm used to optimize classification systems with global validation.

Neighbors to solution X^i are created by swapping bits in the binary string with their complement. For a binary string E with p bits, a set of p neighbors is created by complementing each bit $i, 1 \leq i \leq p$ on solution E^i. For the IFE, solution in Fig. 5.a has solutions in Figs. 5.b and 5.c as two possible neighbors.

4. Experimental Protocol

The tests are performed as in Fig. 2. The IFE methodology is solved to obtain the representation set RS_{IFE}, which is used to train the classifier set K and the most accurate classifier $SI, SI \in K$ is selected. When using the prototyping approach, the RRT algorithm produces RS_{IFE} using the PD classifier, and only when training the selected

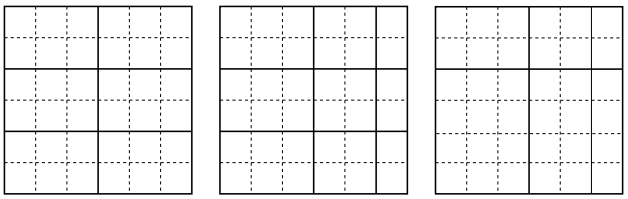

Figure 5. Zoning strategy (a) and two neighbors (b and c) using the proposed divider zoning operator.

SI that the MLP classifier is used. The direct wrapper approach uses the MLP during the optimization process. To select the most accurate solution in the result set S, we use the global validation approach detailed in [author reference]. Solutions obtained are compared to the baseline representation defined in [author reference] by an human expert. The RRT algorithm is deterministic, thus no replications are required to evaluate mean values.

The data sets in Table 1 are used in the experiments – isolated handwritten digits from NIST SD-19. MLP classifier training is performed with the *training* data set, whereas PD classifier training is performed with the $training'$ data set. The *validation* data set is used to adjust the classifier parameters (PD hyper planes, MLP weights). The optimization process is performed with the *optimization* data set, and the *selection* data set is used with the global validation strategy to select solutions. The $test_a$ and $test_b$ databases are used to compare solutions, where the later is known to require a robust classifier for higher accuracies. The number of MLP hidden layers is selected as the most performing in val in one of the five factors of feature set cardinality in $hn = \{0.4, 0.45, 0.50, 0.55, 0.6\}$.

Data set	Size	Origin	Sample range
$training'$	50000	hsf_0123	1 to 50000
training	150000	hsf_0123	1 to 150000
validation	15000	hsf_0123	150001 to 165000
optimization	15000	hsf_0123	165001 to 180000
selection	15000	hsf_0123	180001 to 195000
$test_a$	60089	hsf_7	1 to 60089
$test_b$	58646	hsf_4	1 to 58646

Table 1. Handwritten digits data sets extracted from NIST SD-19.

Both experiments have the IFE initial solution associated to an empty string (all bits set to 0). Thus, there are no active dividers in the initial IFE solution. The deviation D is set empirically to $D = 5\%$. The RRT is a deterministic algorithm, hence a single run is performed with both processes. The baseline representation is compared directly with solutions SI_p, obtained with the prototyping approach (PD classifier), and SI, obtained directly with the MLP classifier, regarding both accuracy, feature set cardinality and IFE optimization required processing time.

5. Results

Experimental results are detailed in Table 2, where Z is the solution zone number, $|S|$ is the solution cardinality (feature number), hn is the MLP hidden layer size, e_{test_a} and e_{test_b} are classification error rates on $test_a$ and $test_b$. Solution SI_p is obtained through classifier prototyping with PD during the optimization process, whereas SI is obtained using the MLP during the optimization process. The baseline representation trained with an MLP classifier is included for comparison purposes.

Solution	Z	$\|S\|$	hn	e_{test_a}	e_{test_b}
Baseline	6	132	60	0.91%	2.89 %
SI_p	15	330	132	0.82%	2.51 %
SI	10	220	132	0.80%	2.48 %

Table 2. Digits optimization results.

Regardless the wrapper approach used during the optimization process, the IFE ouperformed the baseline representation defined by the human expert. When comparing SI and SI_p, we find that both solutions are comparable. Whereas SI is numerically better than SI_p, it represents only a 2.43% error rate decrease on $test_a$ (0.02% difference), and an 1.2% error rate decrease on $test_b$ (0.03% difference). Thus, we observe no significant accuracy improvement on SI when compared to SI_p. On the other hand, SI has a much smaller feature set size $|SI| = 220$, if compared to $|SI_p| = 330$, a 33% feature set size decrease. Classification system performance is directly related to the feature set size it uses, thus, using SI is preferable, instead of SI_p.

The obtained zoning strategies are shown in Fig. 6, along with the baseline solution. We observe that the baseline solution, defined by the human expert, shares the same basic structure as SI and SI_p. All divider bits that are set on the baseline solution are also set on SI and SI_p. We also observe that most divider bits are similar on SI and SI_p, and that differences observed demonstrate that different classi-

fiers require feature sets better adapted for higher performance.

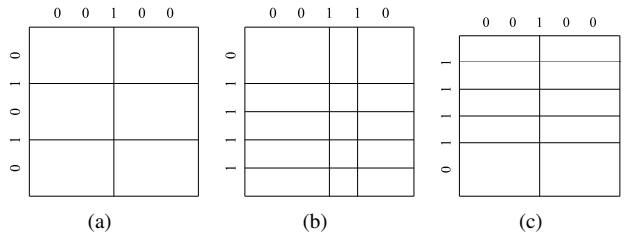

Figure 6. Baseline zoning strategy (a), SI_p (b) and SI (c).

As different classifiers were used on both tests, the number of different solutions evaluated is also different. To optimize SI_p using the PD classifier, a total of 76 different candidate solutions was tested. To optimize SI using the MLP classifier, the RRT algorithm evaluated 53 different candidate solutions. Finally, considering the required processing time, the prototyping approach is faster. On a Core2DUO 3GHz processor with 4GB memory, optmizing SI directly with the MLP classifier took 4 days using both processor cores (one MLP training process per core), whereas optimizing SI_p took 10 hours using only one processor core, which can be further optimized if more processor cores are used. For comparison purposes, using a MOGA to optimize SI_p with the IFE required the evaluation of 450 different candidate solutions. Thus, optmizing IFE using a MOGA and the direct wrapper approach will require further optmizations so it can be processed in reasonable time.

6. Discussion

We tested two different wrapper approaches to optimize classification systems. One direct approach, which uses the target classifier, and a prototyping approach, which uses a different classifier to prototype feature sets to reduce processing time. Whereas both approaches produced solutions that have comparable accuracy, we observed that their cardinality was different, favoring the direct approach to use the target classifier. We also observed that the optmization process took about 10 times longer when using the direct approach, in comparison to the prototyping approach. As the classification system computational cost is related to the feature set used, we can say that it is preferable to optimize classification systems with the direct approach when the time required to optimize the classification system is not an issue, or the training data sets are smaller. Otherwise, the prototyping approach will provide a good trade-off between classification accuracy and required optimization process time.

Current processors used for these experiments indicate that the direct approach (MLP classifier) is limited by the processing time required. To avoid long optimization times, we should investigate GPU based parallelism to accelerate the MLP and PD training procedures, or to use a cluster to increase the computational power for MLPs. Future works will also investigate the use of RS_{IFE} obtained with the direct approach to optimize an EoC in order to obtain a classification system with higher accuracy.

Using the IFE with a MOGA as the optimization algorithm might also be possible after accelerating the MLP training stage. This test would be interesting to confirm the RRT as the computational efficient choice for the IFE, as in [author reference] we observed that the IFE produced comparable results with both a MOGA and the RRT algorithm.

References

[1] K. Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, LTD, Baffins Lane, Chichester, West Sussex, PO19 1UD ,England, 2001.

[2] F. Kimura, S. Inoue, T. Wakabayashi, S. Tsuruoka, and Y. Miyake. Handwritten Numeral Recognition using Autoassociative Neural Networks. In *Proceedings of the International Conference on Pattern Recognition*, pages 152–155, 1998.

[3] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[4] L. I. Kuncheva and L. C. Jain. Design classifier fusion systems by genetic algorithms. *IEEE Transactions on Evolutionary Computation*, 4(4):327–336, 2000.

[5] Z.-C. Li and C. Y. Suen. The partition-combination method for recognition of handwritten characters. *Pattern Recognition Letters*, 21(8):701–720, 2000.

[6] J. W. Pepper, B. L. Golden, and E. A. Wasil. Solving the traveling salesman problem with annealing-based heuristics: A computational study. *IEEE Trans. on Systems, Mand and Cybernetics – Part A: Systems and Humans*, 32(1):72–77, 2002.