Multiple Classifier Systems for Multisensor Data Fusion

Robi Polikar*, Devi Parikh[§] and Shreekanth Mandayam
Electrical and Computer Engineering, Rowan University, Glassboro, NJ 08028
E-Mail: polikar@rowan.edu, dparikh@andrew.cmu.edu, shreek@rowan.edu
*Contributing author: R. Polikar, 136 Rowan Hall, 201 Mullica Hill Road, Glassboro, NJ 08028
http://users.rowan.edu/~polikar/RESEARCH

Abstract – *We have previously introduced Learn^{++}, an ensemble of classifiers based algorithm capable of incremental learning from additional data, and pointed to its feasibility in data fusion applications. In this contribution, we provide additional details, updated results and insight on how such a system can be used in integrating complementary knowledge provided by different data sources obtained from different sensors. Essentially, the algorithm generates an ensemble of classifiers using data from each source, and combines these classifiers using a weighted voting procedure. The weights are determined based on the individual classifier's training performance as well as the observed or predicted reliability of each data source.*

Keywords - *Fusion, combining classifiers, ensemble systems, incremental learning, Learn^{++}*

I. INTRODUCTION

In many applications of pattern recognition and automated identification, it is not uncommon for data obtained from different sensors monitoring a physical phenomenon to provide complimentary information. A suitable combination of such information is usually referred to as *data* or *information fusion*, and can lead to improved accuracy and confidence of the classification decision compared to a decision based on any of the individual data sources alone.

We have previously introduced Learn^{++}, an ensemble of classifiers based approach, as an effective automated classification algorithm that is capable of learning incrementally. The algorithm is capable of acquiring novel information from additional data that later become available after the classification system has already been designed. To achieve incremental learning, Learn^{++} generates an ensemble of classifiers (experts), where each classifier is trained on the currently available database. Recognizing the conceptual similarity between data fusion and incremental learning, we discuss a similar approach for data fusion: employ an ensemble of experts, each trained on data provided by one of the sources, and then strategically combine their outputs. We have observed that the performance of such a system in decision making applications is significantly and consistently better than that of a decision based on a single data source across several benchmark and real world databases.

The applications for such a system are numerous, where data available from multiple sources (or multiple sensors) generated by the same application may contain complementary information. For instance, in non-destructive evaluation of pipelines, defect information may be obtained from eddy current, magnetic flux leakage images, ultrasonic scans, thermal imaging; or different pieces of diagnostic information may be obtained from several different medical tests, such as blood analysis, electrocardiography or electroencephalography, medical imaging devices, such as ultrasonic, magnetic resonance or positron emission scans, etc. Intuitively, if such information from multiple sources can be appropriately combined, the performance of a classification system (in detecting whether there is a defect, or whether a diagnostic decision can be made) can be improved. Consequently, both incremental learning and data fusion involve learning from different sets of data. In incremental learning *supplementary information* must be extracted from new datasets, which may include instances from new classes. In data fusion, complementary information must be extracted from new datasets, which may represent the data using different features.

Traditional methods are generally based on probability theory (Bayes theorem, Kalman filtering),or decision theory such as the Dempster-Schafer (DS) and its variations, which were primarily developed for military applications, such as notably target detection and tracking [1-3]. Ensemble of classifiers based approaches seek to provide a fresh and a more general solution for a broader spectrum of applications. It should also be noted that in several applications, such as the nondestructive testing and medical diagnostics mentioned above, the data obtained from different sources may have been generated by different physical modalities, and therefore the features obtained may be heterogeneous. While using probability or decision theory based approaches

[§] D. Parikh is now with the Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15123

0-7803-9581-6/06/$20.00 ©2006 IEEE

become more complicated in such cases, heterogeneous features can easily be accommodated by an ensemble based system, as discussed below.

An ensemble system combines the outputs of several diverse classifiers or experts. The diversity in the classifiers allows different decision boundaries to be generated by using slightly different training parameters, such as different training datasets. The intuition is that each expert will make a different error, and strategically combining these classifiers can reduce total error [4-6]. Ensemble systems have attracted a great deal of attention over the last decade due to their reported superiority over single classifier systems on a variety of applications [7-10].

Recognizing the potential of this approach for incremental learning applications, we have recently developed Learn^{++}, and shown that Learn^{++} is indeed capable of incrementally learning from new data. Furthermore, the algorithm does not require access to previously used data, does not forget previously acquired knowledge and is able to accommodate instances from classes previously unseen in earlier training [11]. The general approach in Learn^{++}, much like those in other ensemble algorithms, such as AdaBoost [12], is to create an ensemble of classifiers, where each classifier learns a subset of the dataset. The classifiers are then combined using weighted majority voting [13].

In this contribution, we review the Learn++ algorithm suitably modified for data fusion applications [14]. In essence, for each dataset generated from a different source and/or using different features, Learn++ generates new ensemble of classifiers, which are then combined using weighted majority voting.

II. LEARN^{++}

The pseudocode of the Learn^{++} algorithm, as applied to the data fusion problem, is provided in Figure 1, and is described in detail in the following paragraphs.

For each database, FS$_k$, $k=1,\dots,K$, that consists of a different set of features that is submitted to Learn^{++}, the inputs to the algorithm are (i) a sequence S_k of m_k training data instances $\boldsymbol{x_i}$ along with their correct labels y_i ; (ii) a supervised classification algorithm BaseClassifier, generating individual classifiers (henceforth, hypotheses); and (iii) an integer T_k, the number of classifiers to be generated for the k^{th} database.

Each hypothesis h_t, generated during the t^{th} iteration of the algorithm, is trained on a different subset of the training data. This is achieved by initializing a set of weights for the training data, $\boldsymbol{w_t}$, and a distribution $\boldsymbol{D_t}$ obtained from $\boldsymbol{w_t}$ (step1). According to this distribution a training subset $\boldsymbol{TR_t}$ is drawn from the training data $\boldsymbol{S_k}$ (step 2). The distribution $\boldsymbol{D_t}$ determines which instances of the training data are more likely to be selected into the training subset $\boldsymbol{TR_t}$. The Base classifier is trained on $\boldsymbol{TR_t}$ in step 3, which returns the tth hypothesis h_t. The error of this hypothesis, ε_t, is computed on the current database $\boldsymbol{S_k}$ as the sum of the distribution weights of the misclassified instances (step 4). This error is required to be less than ½ to ensure that a minimum reasonable performance can be expected from h_t. If this is the case, the hypothesis h_t is accepted and the error is normalized to obtain the normalized error (step 5).

```
Algorithm Learn$^{++}$ for Data Fusion
Input: For each feature set $FS_k$, $k=1,2,\dots,K$
  • Training data $S_k = [(\mathbf{x}_i, y_i)]$, i=1,…,$m_k$
  • Supervised algorithm BaseClassifier.
  • Integer $T_k$, specifying the number of classifiers.
 Do for each $k=1,2,\dots,K$:
   Initialize $w_1(i) = \boldsymbol{D_1}(i) = 1/m_k, \ \forall i, \ i = 1,2,\cdots,m_k$
   Do for $t = 1,2,\dots,T_k$:
     1. Set $\boldsymbol{D_t} = \boldsymbol{w}_t \Big/ \sum_{i=1}^{m_k} w_t(i)$
     2. Draw training $\boldsymbol{TR_t}$ subset from $\boldsymbol{D_t}$.
     3. Obtain $h_t$ by training with data $\boldsymbol{TR_t}$
     4. Calculate the error of $h_t$
          $\varepsilon_t = \sum_{i:h_t(\mathbf{x}_i) \neq y_i} \boldsymbol{D_t}(i)$
        on $S_k$. If $\varepsilon_t > 1/2$, discard $h_t$ and →Step 2.
     5. Set $\beta_t = \varepsilon_t / (1-\varepsilon_t)$. Obtain the composite hypothesis through weighted majority voting
          $H_t = \arg\max_{y \in \Omega} \sum_{t:h_t(\mathbf{x})=y} \log(1/\beta_t)$
     6. Compute error of $H_t$ : $E_t = \sum_{i:H_t(\mathbf{x}_i) \neq y_i} \boldsymbol{D_t}(i)$
     7. Set $B_t = E_t/(1-E_t)$, and update the weights:
          $w_{t+1}(i) = w_t(i) \times \begin{cases} B_t, & \text{if } H_t(\mathbf{x}_i) = y_i \\ 1, & \text{otherwise} \end{cases}$
  Compute voting weights adjustment factor
          $\alpha_k = \left( \sum_{i=1}^{m_k} \left[ \left| H_{T_k}(\mathbf{x}_i) \neq y_i \right| \right] \right) \Big/ m_k$
 Output the final hypothesis:
   $H_{final}(\mathbf{x}) = \arg\max_{y \in \Omega} \sum_{k=1}^{K} \sum_{t:h_t(\mathbf{x})=y} \log\left( \frac{1}{\beta_t \alpha_k} \right)$
```

Fig. 1. The Learn^{++} algorithm for data fusion

If $\varepsilon_t \geq$ ½, the current hypothesis is discarded, and a new training subset is selected by returning to step 2. All t hypotheses generated thus far are then combined using weighted majority voting (WMV) to obtain a composite hypothesis H_t. In WMV, each hypothesis is assigned a weight that is inversely proportional to its error, giving a higher weight to classifiers with smaller training error. The error of the composite hypothesis H_t is then computed in a similar fashion as the sum of the distribution weights of the instances that are misclassified by H_t (step 6).

The normalized composite error B_t is obtained which is then used for updating the distribution weights assigned to individual instances in Step 7. The distribution weights of the instances correctly classified by the composite hypothesis H_t are reduced by a factor of B_t; hence when the distribution is re-normalized in step 1 of the next iteration, the weights of the misclassified instances are effectively increased. We note that this weight update rule, based on the performance of the current ensemble, facilitates learning from new data. This is because, when a new dataset is introduced (particularly with new classes or features), the existing ensemble (H_t) is likely to misclassify the instances that have not yet been properly learned, and hence the weights of these instances are increased, forcing the algorithm to focus on the new data.

An additional set of weights are introduced in data fusion applications for each ensemble. These weights represent the importance and reliability of the particular data source and can be assigned based on former experience, (e.g., for diagnosing a neurological disorder we may know that magnetic resonance imaging (MRI) is more reliable then electroencephalogram, and we may therefore choose to give a higher weight to the classifiers trained with MRI data), or they can be based on the performance of the ensemble trained on the particular feature set on its own training data. We have calculated a set of such weights α_k for the k^{th} dataset based on the training performance of the ensemble trained on the k^{th} dataset, and adjusted the voting weights using α_k. The adjusted weight of each classifier is then used during the weighted majority voting for the final hypothesis H_{final}

The schematic representation of the algorithm is provided in Figure 2. Simulation results of Learn^{++} on incremental learning using several datasets, as well as comparisons to the other methods of incremental learning such as Fuzzy ARTMAP can be found in [11]. The simulation results of Learn^{++} on two applications of data fusion are presented below, which primarily include additional detail, updated results and further insight than those presented in [14,15]. Both of these applications are real world applications, one involving the combination of ultrasonic and magnetic flux leakage data for identification of pipeline defects, and the other involving the combination of chemical sensor data from several sensors for identification of volatile organic compounds.

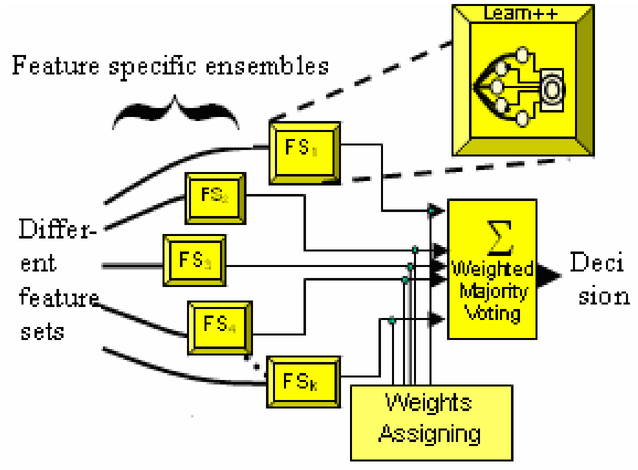

Fig. 2. Schematic representation of algorithm

III. RESULTS

A. Nondestructive Evaluation Database

Nondestructive evaluation is primarily concerned with detection and identification of flaws in various types of materials. Data fusion methods for NDE data have been developed, which usually employ one or more of the methods mentioned above [16]. In this work, two datasets that consist of heterogeneous features were fused: magnetic flux leakage (MFL) images, ultrasonic testing (UT) images. Illustrations of these images and the type of defect they represent are shown in Figure 3.

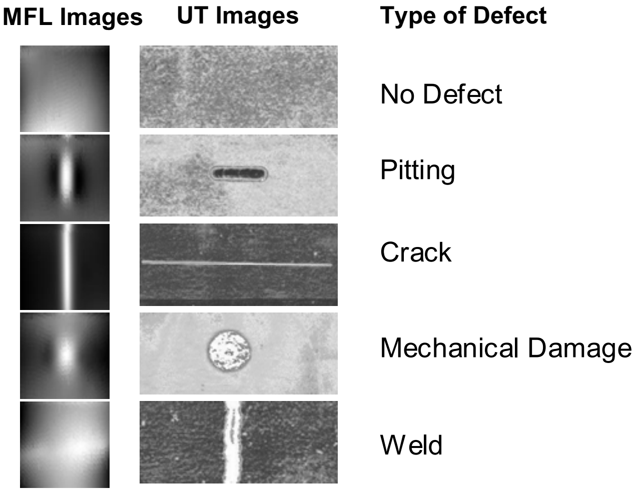

Fig. 3. Sample MFL and UT images of defect types

Two-dimensional discrete Fourier transform based features were extracted from each imaging modality with 15 features for MFL and 72 features for the UT. The database consisted of 21 images from to a total of 5 classes: (i) No defect: 4 images; (ii) Pitting: 9 images; (iii) Crack: 4 images; (iv) Mechanical Damage: 4 images; (v) Weld: 4 images. Ten images (2 from each class) were randomly selected for training and the remaining 11 were used for validation.

The base classifier was a single hidden layer MLP with 0.05 error goal, 30 hidden layer nodes and 30 classifiers trained with each dataset. These parameters were selected based on large number of (several thousand) statistical tests to determine the optimum parameters. Once the optimum parameters were selected, training and testing were repeated 40 times, the average results of which are given in Table 1. These numbers suggest that the data fusion performance is significantly better than either of the individual MFL and UT performances, even when the classifier parameters are optimized for each feature set.

Table 1: Generalization performances - 95% CI

Dataset	Average generalization performance
MFL	81.60 ± 3.62 %
UT	79.87 ± 2.69 %
Fusion	95.02 ± 2.00 %

B. Volatile Organic Compounds Database

This database was generated from responses of twelve quartz crystal microbalances (12 features) to twelve volatile organic compounds including acetone, acetonitrile, toluene, xylene, hexane, octane, methanol, ethanol, methyethylketone, tricholoroethylene, tricholoroethane, and dicholoroethane. The sensors themselves were each coated with a different polymer, each carefully selected to be sensitive to at least some of the volatile organic compounds (VOCs) listed above, or more specifically to at least one of the functional groups represented by the volatile organic compounds (such as alcohols, benzenes, etc.). There were seven responses for each VOC, which represent seven different concentration levels, in the 70~700 ppm range for a total of 84 instances (7 from each class). Four instances were randomly picked from each class (VOC) and were used for training while the remaining 36 instances were used for testing. The available 12 sensors were randomly divided into three feature sets (with four sensors each) to simulate a data fusion scenario.

Similar to the nondestructive evaluation dataset, the optimum set of parameters were determined through a large number of tests, where the performance of hundreds of different combinations of error goal, number of hidden layers, and number of classifiers were evaluated. Based on these analyses, the following parameters were deemed to be optimal: error goal of 0.0005, 15 hidden layer nodes and 5 classifiers in each ensemble. We should note that the purpose of seeking the optimal parameters is to find out whether the data fusion system can provide an additional performance improvement to over any of the individual data sources, if the classifiers trained with these individual sources were optimized.

All possible combinations of data fusion were performed: sensor set 1, 2 and 3 were combined with each other, as well all together. Cross validation was performed with over 100 random partitions of training and test data. Corresponding generalization performance results are summarized in Table 2, along with their 95% confidence intervals, obtained from the averages of the 100 independent trials.

Table 2: Generalization performances - 95% CI

Feature Set	Average generalization performance
1	84.6 ± 1.1 %
2	86.1 ± 1.3 %
3	86.1 ± 1.1 %
1 & 2	90.2±1.0%
1 & 3	89.5±1.0%
2& 3	90.0±1.0%
1, 2 & 2	91.3 ± 0.9 %

Not only the data fusion performance of any two sensor set was better than any of the individual sensor set performances, but also the performance of all sensors fused was better than any of the individual or two way combinations, even with optimum parameters, indicating that the algorithm was able to extract complementary information from the three different sets of sensors.

We should also point however that, while the performance improvement obtained by combining two data sources was always statistically significant compared to the performance of classifiers trained on single data source, the same claim cannot be made for combining all three datasets. In fact, the performance improvement obtained by combining all three datasets is not statistically significant over the performance obtained by two-way combination of the datasets. This may indicate that, while there is significantly complementary information to be obtained by combining data from any two sources, there is nothing additional to be learned from a third database, above and beyond what is already learned from the previous two data sources.

IV. CONCLUSIONS

Recognizing the conceptual similarities between incremental learning and data fusion, the incremental learning algorithm Learn^{++} has been evaluated in a data fusion setting. The algorithm sequentially learns from data comprised of different sets of features by generating an ensemble of classifiers for each dataset, and then combining them through a modified weighted majority voting scheme. We have evaluated the algorithm on two real world data fusion applications: identifying defect types from UT and MFL images and identifying the type of a VOC present in the environment from chemical sensor responses.

The results indicate that the Learn^{++} algorithm, when used to combine information from two or more datasets, consistently performed significantly better than each of the testing modalities, even if the classifiers trained on individual datasets are optimized. Therefore, the advantage of Learn^{++} is that data from different measurement modalities can be sequentially added without having to retrain the entire system, with an added advantage of improved classification performance. The ability of the algorithm to learn incrementally as well as to fuse different datasets to extract additional information makes Learn^{++} a versatile algorithm.

Tests are also currently being conducted to observe the sensitivity of the data fusion performance to varying parameters (for instance, using randomly selected moderate parameters for the individual feature sets, as opposed to optimized parameters). It is expected that fusing feature sets that have been optimized yields a smaller margin of increase in performance using data fusion and can be used as a fine tuning step. On the contrary, fusing features that are not optimally obtained would provide a larger margin of improvement on the fusion performance, and thus can be used as an alternative for the expensive optimization process.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grant No. ECS-0239090, "CAREER: An Ensemble of Classifiers Approach for Incremental Learning."

REFERENCES

[1] D. Hall and J. Llinas, "An introduction to multisensor data fusion," *IEEE Proceedings*, vol. 85, no. 1, 1997.

[2] D. Hall and J. Llinas (editors), *Handbook of multisensor data fusion*, CRC Press: Boca Raton, FL, 2001.

[3] L. A. Klein, Sensor *and Data Fusion Concepts and Applications*, SPIE Press, vol. TT35: Belingham, WA, 1999.

[4] L.K. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 10, pp. 993-1001, 1990.

[5] T.G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization," *Machine Learning*, vol. 40, no. 2, pp. 1-19, 2000

[6] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123-140, 1996.

[7] T.G. Dietterich, "Ensemble methods in machine learning," *Proc. 1st Int. Workshop on Multiple Classifier Systems (MCS 2000)*, LNCS vol. 1857, pp. 1 – 15, Springer: New York, NY, 2000.

[8] T. Windeatt and F. Roli (eds), *Proc. 3rd Int. Workshop on Multiple Classifier Systems (MCS 2002),* LNCS vol. 2364, p. 1-15, Springer: New York, NY, 2002

[9] T. Windeatt and F. Roli (eds), *Proc. 4th Int. Workshop on Multiple Classifier Systems (MCS2003)*, LNCS, vol. 2709, Springer: New York, NY, 2003.

[10] L.I. Kuncheva, *Combining Pattern Classifiers –Methods and Algorithms*, Hoboken, NJ: Wiley Interscience, 2004.

[11]R. Polikar, L. Udpa, S. Udpa, V. Honavar, "Learn++: an incremental learning algorithm for supervised neural networks," *IEEE Trans. Sys., Man, Cyber.*, vol.31, no.4, pp.497-508, 2001.

[12]Y. Freund and R. Schapire, "A decision theoretic generalization of online learning and an application to boosting," *Computer and System Sciences*,vol. 57, no. 1, pp. 119-139, 1997.

[13] N. Littlestone and M. Warmuth, "Weighted majority algorithm," *Information and Computation*, vol. 108, pp. 212-261, 1994.

[14] D. Parikh, A. Gangardiwala, M. Kim, J. Oagaro, S. Mandayam and R. Polikar, "Combining classifiers for multisensor data fusion," *IEEE Int. Conf. on Systems, Man and Cybernetics*, pp. 1232-1237, The Hague, The Netherlands, October 2004

[15] D. Parikh and R. Polikar, "A multiple classifier approach for multisensor data fusion," Information Fusion 2005, July 2005, Philadelphia (unpublished as of this publication)

[16] X. E. Gros, *NDT Data Fusion*, Arnold Publishers, 1997.